Using Position Manager EnDat22 Library on IDDK Hardware

User's Guide

![Texas Instruments logo]

Contents

List of Figures

List of Tables

![Texas Instruments logo]

Using Position Manager EnDat22 Library on IDDK Hardware

1 Introduction to IDDK

The DesignDRIVE Kit (IDDK) is a single platform that makes it easy to develop and evaluate design solutions for many industrial drive and servo topologies. The IDDK offers support for a wide variety of motor types, sensing technologies, encoder standards and communications networks, as well as easy expansion to develop with real-time Ethernet communications and functional safety topologies, enabling more comprehensive, integrated system solutions. Based on the real-time control architecture of TI's C2000™ microcontrollers (MCUs), the kit is ideal for the development of industrial inverter and the servo drives used in robotics, computer numerical control machinery (CNC), elevators, materials conveyance, and other industrial manufacturing applications.

The IDDK offers an integrated drive design with a full-power stage to drive a three-phase motor, easing the evaluation of a diverse range of feedback sensing and control topologies. The kit includes a 180-pin HSEC controlCARD based on the TMS320F28379D C2000 Delfino™ MCU, which integrates dual C28x real-time processing cores and dual CLA real-time co-processors, providing 800 MIPS of floating-point performance with integrated trigonometric and FFT acceleration.

The sophisticated sensing peripherals on the TMS320F28379D MCU, including sigma-delta filter modules with up to eight input channels, four high-performance 16-bit ADCs and eight windowed comparators, enable the IDDK to support shunt, flux gate/ HALL, and sigma-delta current sensing simultaneously. For position feedback, the IDDK leverages integrated MCU support for resolver and incremental encoder interfaces. In addition, customers can also explore configuration options that allow the MCU to be placed on either side of the high voltage isolation barrier.

The kit is designed to plug into 110V/220V AC mains, delivers up to 8 Amps, and is rated to drive motors up to one horsepower.

This document covers the kit contents and hardware details, and explains the functions and locations of various connectors present on the board. This document supersedes all the documents available for the kit.

For more details, see the *DesignDRIVE Development Kit IDDK Hardware Reference Guide* (SPRUI23).

2 Hardware Configuration

To experiment with IDDK for the Position Manager EnDat22 Library evaluation, the following components are needed:
- An IDDK EVM
- TMDXCNCD28379D
- USB-B to A cable
- Encoder/connectors NOT included in the kit – required for evaluation
  - EnDat22 Encoder from Heidenhain (Ex: ROC425/ROC437)
  - EnDat22 M12 8-pin cable from Heidenhain – length as needed by the application (max 100m)

Figure 1. EnDat22 M12 8-Pin Cable

– Adapter to connect Heidenhain cable to IDDK

  Circular 8 position female to wire leads – M12 receptacle

Figure 2. Circular 8 Position Female to Wire Leads

– An external isolated 15 V power supply needed for MCU code development (preferably with a barrel connector commonly referred to as DC jack). Power supply EMSA150120-P5P-SZ is used for testing.

– PC with Code Composer Studio™ (CCSv6 or greater) installed

• IDDK Hardware Reference Guide located at:

  controlSUITE\development_kits\TMDSIDDK_v2.0\~Docs

• For the schematic details of the IDDK EVM, refer to the schematic file located at:

  controlSUITE\development_kits\TMDSIDDK_v2.0\IDDK_HwDevPkg\IDDK_HwDevPkg_v2.2.1
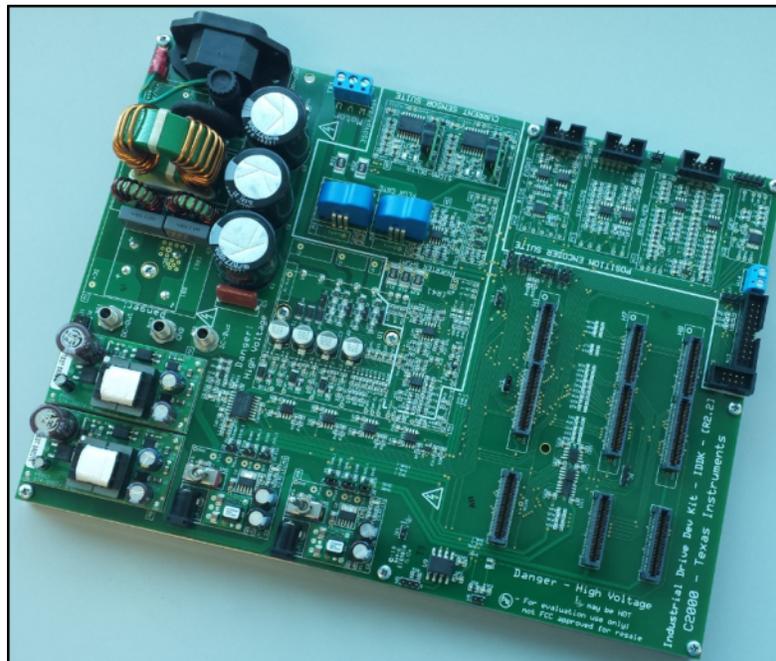
Figure 3. IDDK EVM Kit

3 Hardware Overview

This section describes the subset of the components required for the Position Manager EnDat22 Library evaluation only. Complete hardware overview of the kit can be obtained from *DesignDRIVE Development Kit IDDK Hardware Reference Guide* (SPRUI23) available on controlSUITE.

Evaluation of the Position Manager EnDat22 Library requires usage of:

• Processor (CPU) block for control
• Position encoder suite
• On-board power supplies

3.1 Functional Blocks

Table 1 illustrates the subset of functional blocks on the DesignDRIVE Development Kit IDDK Hardware, along with the macro names used for the Position Manager EnDat22 Library evaluation.

Table 1. Hardware Macros in IDDK Used for PM EnDat22 Evaualtion

Functional Block	Macro Reference	Macro Function
Power Supplies	M9	DC Power Supply – Linear Reg 15 V-5 V/3.3 V
Position Encoder Suite	M12	EnDat Encoder Interface
Processor/ControlCard	H1	All other functions

Each functional block, and the macros that make them, are presented in brief detail in the following sections. The layout of various macros in the board is given in Figure 4. Schematic details of the individual macros are available at:

controlSUITE\development_kits\ TMDSIDDK_v2.0\IDDK_HwDevPkg\IDDK_HwDevPkg_v2.2.1

Figure 4. Layout of IDDK EVM With its Functional Macros

3.2 *Processor Section (Control Processor Slot – H1)*

The IDDK is designed around the main control processor card in slot H1. This is designed to plug in a C2000 Delfino (TMS320F28379D) MCU control card TMDXCNCD28379D designed with a HSEC180-pin edge connector (see Figure 5).
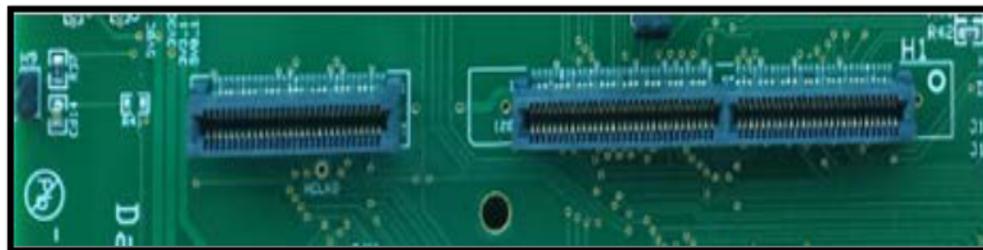
Figure 5. Processor Block

3.3 *Position Encoder Suite*

This block provides a range of position encoder and sensing interfaces (see Figure 6) such as:

- QEP
- Resolver
- Sin-Cos

Copyright © 2015, Texas Instruments Incorporated

- EnDat/BiSS
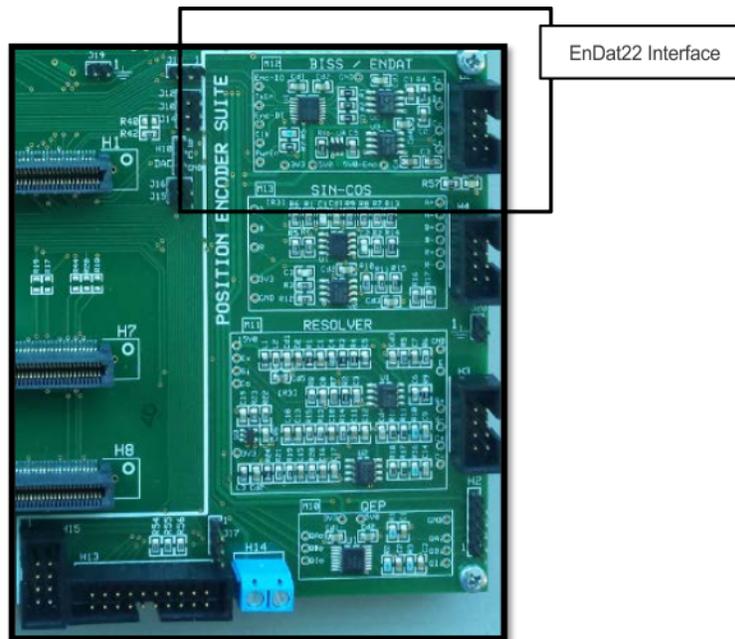
Figure 6. Position Sensor Suite

The PM EnDat22 Library evaluation uses this interface for EnDat communication.

3.3.1 EnDat Encoder

This is designed as a macro, referred to as M12, which is a common interface for EnDat encoders. H6 is the external interface header and it only interfaces digital signals as shown in Figure 7. GPIO32 is used to control on/off of the 5 V supply details in Table 2.
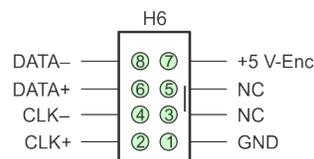
Figure 7. EnDat/BiSS Interface Header

PM EnDat22 Library evaluation uses this interface for EnDat communication.

3.4 Power Supply

The kit only needs via M9 that has a power supply jack JP1 and a toggle switch SW1. External (15 V) power supply can be fed in through JP1 while SW1 should be turned on to feed the linear regulators of M9. External power supply is preferred to power the controller during code development so that the board operates in low voltage. This ensures a safe environment as there is no high-voltage node present on the board.
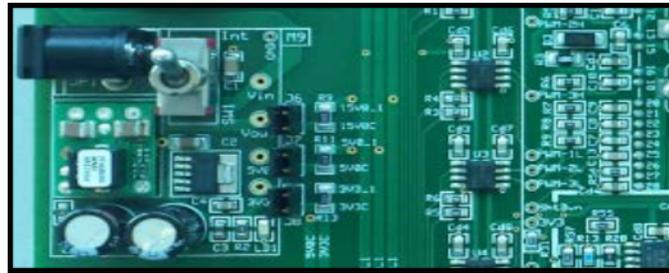
Figure 8. Power Supply Block

4   Hardware Resource Mapping

Section 4.1 through Section 4.3 shows hardware resource requirements and signal mapping for the PM EnDat22 Library evaluation on IDDK.

4.1   Digital Signal Mapping

Table 2 shows the functional mapping of signals connected to the control processor on H1 that are relevant to PM EnDat22 Library evaluation.

Table 2. Digital Signal Mapping on Control Card H1

| IDDK Signal Name	MCU GPIO	MCU Peripheral Associated With GPIO	Function
En-Clk	6	ABS ENC-CLK	Absolute Encoder Clock out
C1-SPIB-CLK	7	PEM CLK	Encoder SPI Clock out
En-DO	24	SPIB	SPI I/f for Abs Encoder
En-DI	25	SPIB	SPI I/f for Abs Encoder
C1-SPIB-CLK	26	SPIB	SPI I/f for Abs Encoder
C1-SPIB-STE	27	SPIB	SPI I/f for Abs Encoder
En-PwrEn	32	GPIO output	Power Enable for Abs Encoder
En-TxEn	34	(GPIO) PEM TX En	Tx Enable for Abs Encoder

4.2   Jumpers and Switches

The jumpers shown in Table 3 need to be populated for the PM EnDat22 Library evaluation.

Table 3. Purpose of Jumpers and Switches

Jumpers and Switches	Configuration	Description
[Main] - J6- J8	Populate	Jumpers to bring out linear regulator block M9 voltages
[Main] - J10	Populate	Connects GPIO to EnDat Clock
[Main] - J12	Don't Populate	
[Main] - J13	Populate	Connect SPISTE to ground
[Main] - J14-17	Don't populate	
[Main] - J18	Populate (1-2)	Connects Encoder data out (En-Do-1 to the processor)
[Main] - J19-J21		GND headers for probe access
[M9] – SW1	Turn ON	Select 15 V supply from JP1 or onboard from M8 to generate 5 V and 3.3 V

4.3 Headers and Connectors

Table 4 shows the headers and connectors available on board.

Table 4. Headers and Connectors

Headers/ Connectors	Description
[M9] – JP1	15 V DC power supply jack adaptor
[Main] - H1	180 pin HSEC connector slot for control processor card
[Main] – H6	4x2 header for EnDat

5 Hardware Setup Instructions for PM EnDat22 Library Evaluation

1. Ensure default configuration. Make sure that jumpers [Main]-J6, [Main]-J7 and [Main]-J8 (in front of macro M9) and resistors [Main] R8-R13 are populated, and that GND plane resistors R14 and R15 are mounted as shown in the *Various GND Planes on the Bottom of Board* and *Default Connection of Various GND Planes* figures of the *Power Supplies* section in the *DesignDRIVE Development Kit IDDK v2.2 - Hardware Reference Guide* (SPRUI23).

2. Unpack the TMDXCNCD28379D control card and slide it in the connector slot of [Main]-H1. Push down vertically using even pressure on both ends of the card until it cannot slide down further. Spread open the retaining clips, if present, to remove the card and pull the card out by applying even force at the far edges.

3. Connect a USB cable to connector J1 on the control card. The control card isolates the JTAG signals between the C2000 device and the computer. LED D2 on the control card should light.

4. Connection to the encoder:

   (a) Prepare an adapter to connect the Heidenhain cable to the IDDK EnDat interface ([Main] – H6 4x2 header) using the circular 8 position female to wire leads – M12 receptacle (see Figure 9).
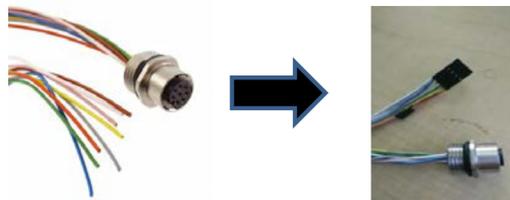
Figure 9. Adapter to Connect the Heidenhain Cable to IDDK

   For more information, see the Heidenhain and hardware schematic in Section 3.3.

   (b) Cable connections between the Encoder and IDDK – insert the header created using the adapter above into ([Main] – H6 4x2 header of IDDK); the other end of the adapter connects to the male side of the Heidenhain M12 cable. The female end of the M12 Heidenhain cable connects to the encoder as shown in Figure 10.

Figure 10. M12 Heidenhain Cable Connects to the Encoder

5. Ensure that toggle switch [M9]-SW1 is in "Int" position. Connect an isolated 15 V DC power supply to [M9]-JP1.

6. Turn on the toggle switch [M9]-SW1, now [M9]-LD1 should turn on. Note that more LEDs on the control card should light up indicating that the control card is receiving power from the board.

6   Software Setup for PM EnDat22 Evaluation Example Projects

6.1   *Installing Code Composer and controlSUITE*

1. Install Code Composer v6.x or later from http://www.ti.com/tool/CCSTUDIO, if not already installed.

2. Go to http://www.ti.com/controlsuite and run the controlSUITE installer. Allow the installer to download and update any automatically checked software for C2000.

3. EnDat22 Library is available at:

    <base> install director is

    C:\ti\controlSUITE\libs\app_libs\position_manager\vX.X

    The following sub-directory structure is used:

    <base>\Doc - Documentation

    <base>\Float - Contains implementation of the library and corresponding include file

    <base>\GUI - Host side GUI for displaying the position information, and so forth.

    <base>\examples - Example using EnDat22 library

6.2 Software Flow Diagram

The software flow used in the example project PM_endat22_BasicPosAcc_DelComp is depicted in Figure 11.
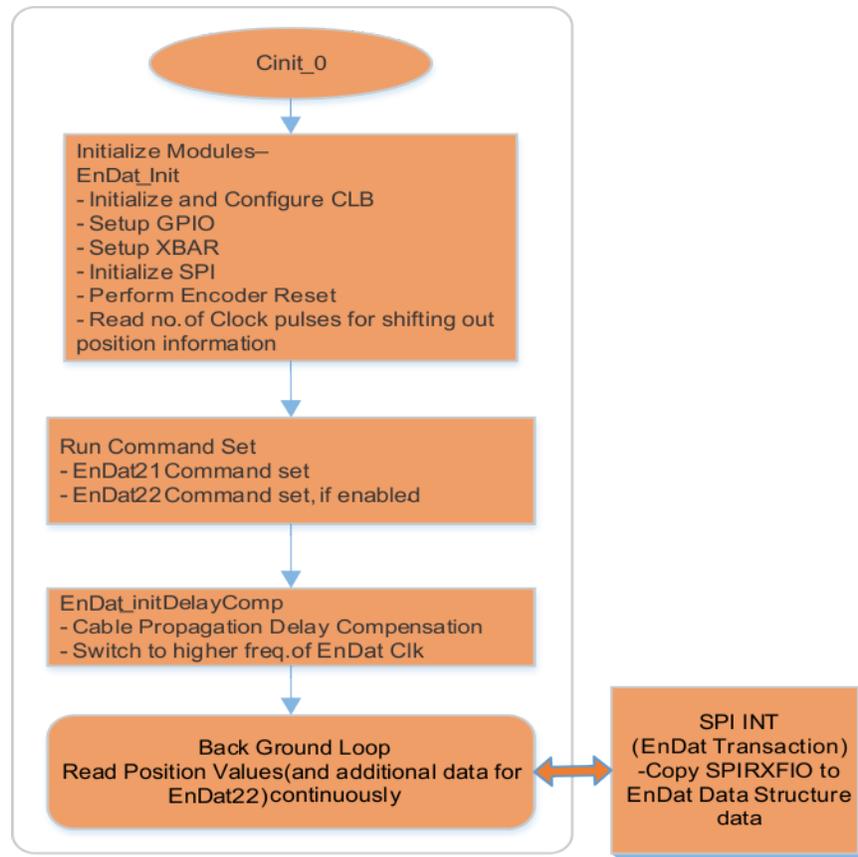
Figure 11. Software Flow Diagram for the Example Project PM_endat22_BasicPosAcc_DelComp

7 Setup Code Composer Studio to Work With TMDXIDDK379D

1. Open "Code Composer Studio". Note that this document assumes that version 6 or later is used.
2. Once Code Composer Studio opens, the workspace launcher may appear that would ask to select a workspace location. Note that workspace is a location on the hard drive where all the user settings for the IDE (which projects are open), what configuration is selected, and so forth are saved. This can be anywhere on the disk, the location mentioned below is just for reference. Note that if this is not your first-time running Code Composer the dialog below may not appear).
   - Click the "Browse…" button
   - Create the following path by making new folders as necessary:
     C:\c2000 projects\CCSv6_workspaces\PM_endat22_eval_workspace
   - Uncheck the box that says "Use this as the default and do not ask again".
   - Click "OK"
3. This will open a 'Getting Started' tab with links to various tasks from creating a new project, importing an existing project, to watching a tutorial on CCS. You can click the 'Import Project' icon that will skip the procedure to step 2 in Section 7.1. Or, you can close the 'Getting Started' Tab and go to next step.
4. Code Composer is configured in order to know which MCU it will be connecting to. This is done by setting up the 'Target Configuration'. All of these are already set up and configured in "xds100v2_F2837x.ccxml" (provided as part of the files in the project), therefore, the user can skip to step 1 in Section 7.1. However, for general information regarding setting up this configuration file, steps 6, 7 and 8 can be used.

5. A new configuration file can be set by clicking "View → Target Configuration. This will open the Target Configuration window. In this window, click on 🗒. Give a name to the new configuration file depending on the target device. If "Use shared location" checkbox is checked, then this configuration file can be stored in a common location by CCS for use by other projects as well. Then, click Finish.

6. This should open up a new tab as shown in Figure 12. Select and enter the following options:

    (a) Connection – Texas Instruments XDS100v2 USB Emulator (or)
        Texas Instruments XDS100v2 USB Debug Probe

    (b) Device – the C2000 MCU on the control card, TMS320F28379D, for example

    (c) Click Save and close
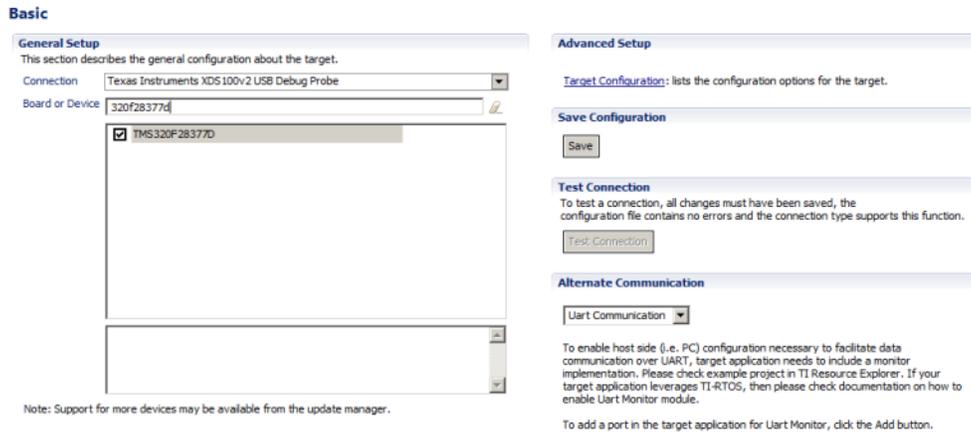
Figure 12. Configuring a New Target

7. Click "View → Target Configurations". In the "User Defined" section, find the file that was created in steps 6 and 7. Right click on this file and select "Set as Default". To use the configuration file supplied with the project, click "View → Target Configurations, expand "Projects → PM_endat22_BasicPosAcc_DelComp, and right-click on the "xds100v2_F2837x.ccxml" and "Set as Default" files. This tab also allows you to reuse the existing target configurations and links them to specific projects.

8. Add the PM EnDat22 evaluation example project into the current workspace by clicking "Project → Import CCS Project".

    (a) Select the project by browsing to:

        C:\ti\controlSUITE\libs\app_libs\position_manager\v1_00_00_00\endat22\examples\PM_endat22_BasicPosAcc_DelComp
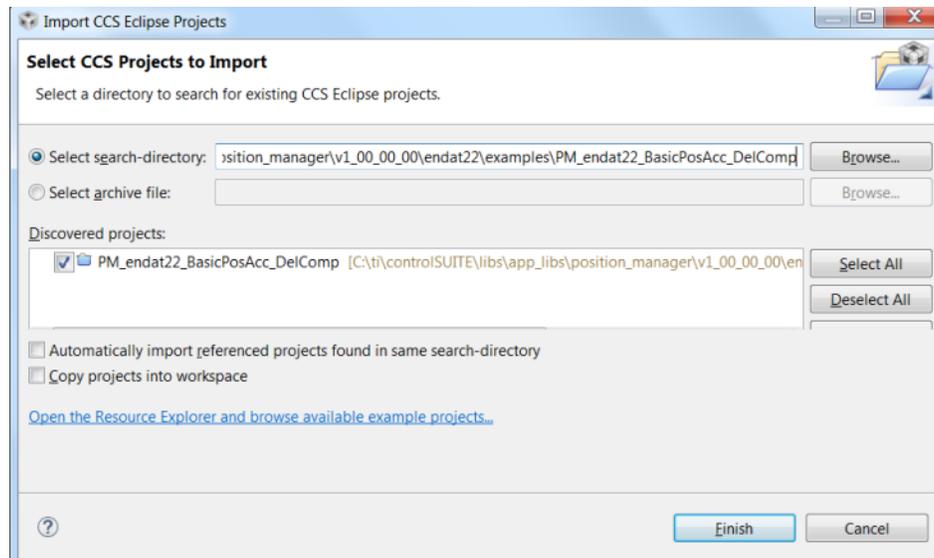
Figure 13. Adding PM EnDat22 Example Project to Workspace

(b) If there are multiple projects in this directory, choose the projects to import and then click Finish. This copies all of the selected projects into the workspace. There is only one project in Figure 13, select it and click Finish.

7.1 Configuring a Project

1. Assuming this is your first time using Code Composer, the xds100v2-F2837x should have been set as the default target configuration. Verify this by viewing the xds100v2-f2837x.ccxml file in the expanded project structure and an [Active/Default] status written next to it. By going to "View → Target Configurations", you can edit existing target configurations or change the default or active configuration. You can also link a target configuration to a project in the workspace by right clicking on the Target configuration name and selecting Link to Project.

2. The project can be configured to create code and run in either Flash or RAM. You can select either of the two, however, RAM configuration is used most of the time for lab experiments and Flash configuration for production. As shown in Figure 14, right-click on an individual project and select Active Build Configuration → CPU1_RAM configuration.
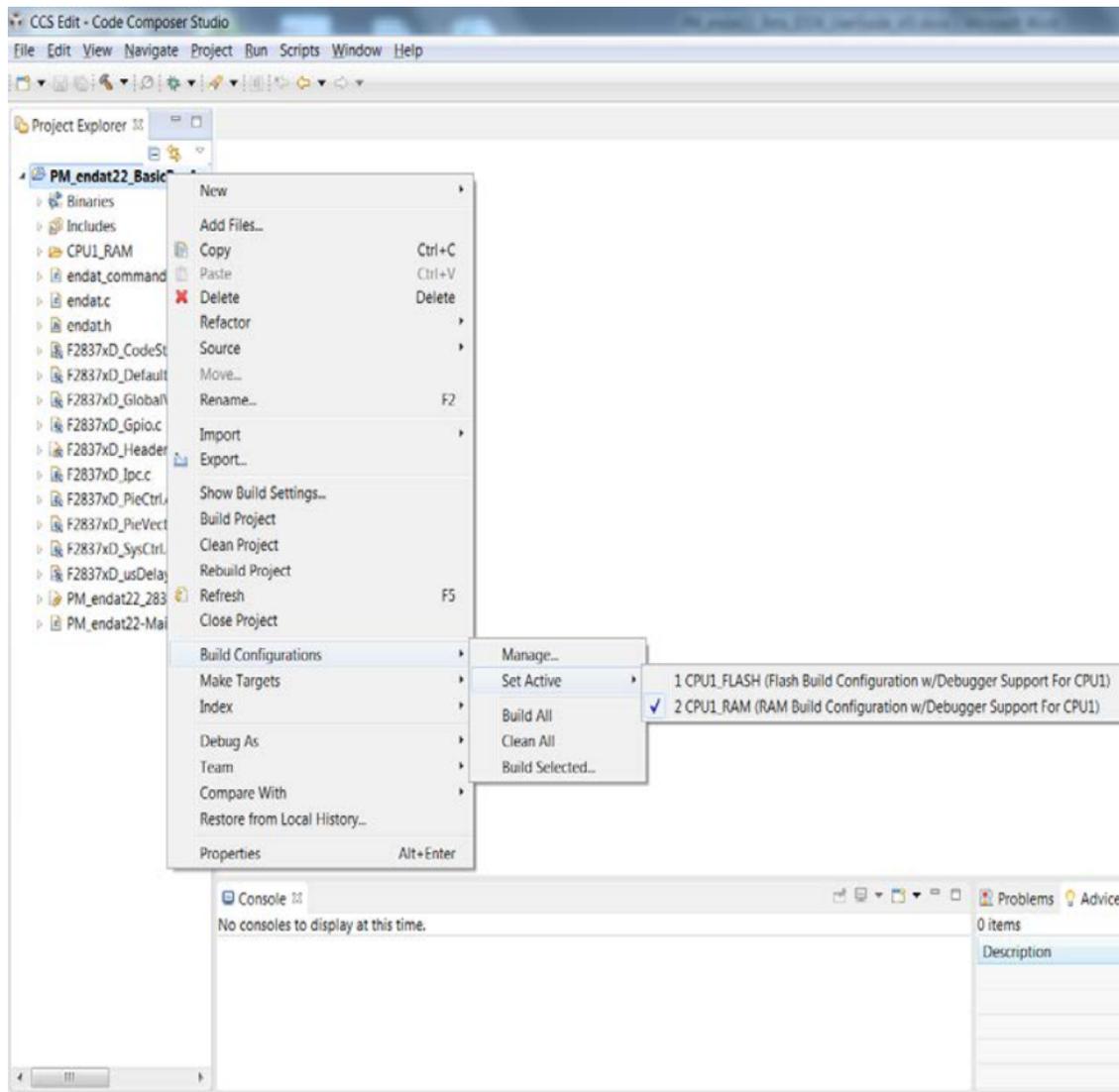
Figure 14. Selecting the F2837x_RAM Configuration

7.2 Build and Load the Project

1. Open the endat.h file and make sure that ENDAT_RUNTIME_FREQ_DIVIDER and ENDAT_INIT_FREQ_DIVIDER are set as needed; save this file. For more details, see the *C2000 Position Manager EnDat22 Library Module User's Guide* (SPRUI35).

   C:\ti\controlSUITE\libs\app_libs\position_manager\v1_00_00_00\endat22\Doc\PM_EnDat22_Lib.pdf

2. Right click on the Project Name and select "Rebuild Project", then watch the Console window. Any errors in the project will be displayed in the Console window.

3. On successful completion of the build, click the "Debug" button ✷ (located in the top-left side of the screen). The following window may pop up if it is the first time that it is used, requesting that you select which of the two CPUs (in F28379D) to connect to (see Figure 15). Select CPU1 by clicking the box next to CPU1.
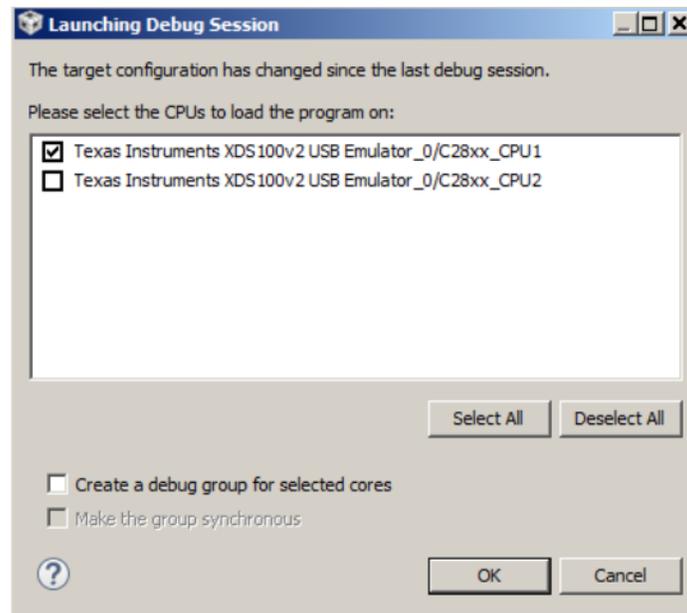
Figure 15. Selecting the CPU(s) to Connect

4. The IDE will automatically connect to the target, load the output file into the device, and change to the Debug perspective.

5. Click the Tools → Debugger Options → Program / Memory Load Options. You can enable the debugger to reset the processor each time it reloads the program by checking "Reset the target on program load or restart" and click "Remember My Settings" to make this setting permanent.

6. Click on the "Enable silicon real-time mode" button 🔲, which auto selects the "Enable polite real-time mode" button 🔲. This allows you to edit and view variables in real time. Do not reset the CPU without disabling these real-time options.

7. Select YES to enable debug events, if a message box appears. This will set bit 1 (DGBM bit) of the 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.

7.3   Setup Watch Window and Graphs

1. Click View → Expressions on the menu bar to open a *watch window* to view the variables being used in the project. Add variables to the watch window as shown in Figure 16. It uses the number format associated with variables during declaration. You can select a desired number format for the variable by right clicking on it and choosing. Figure 16 shows a typical expressions window.
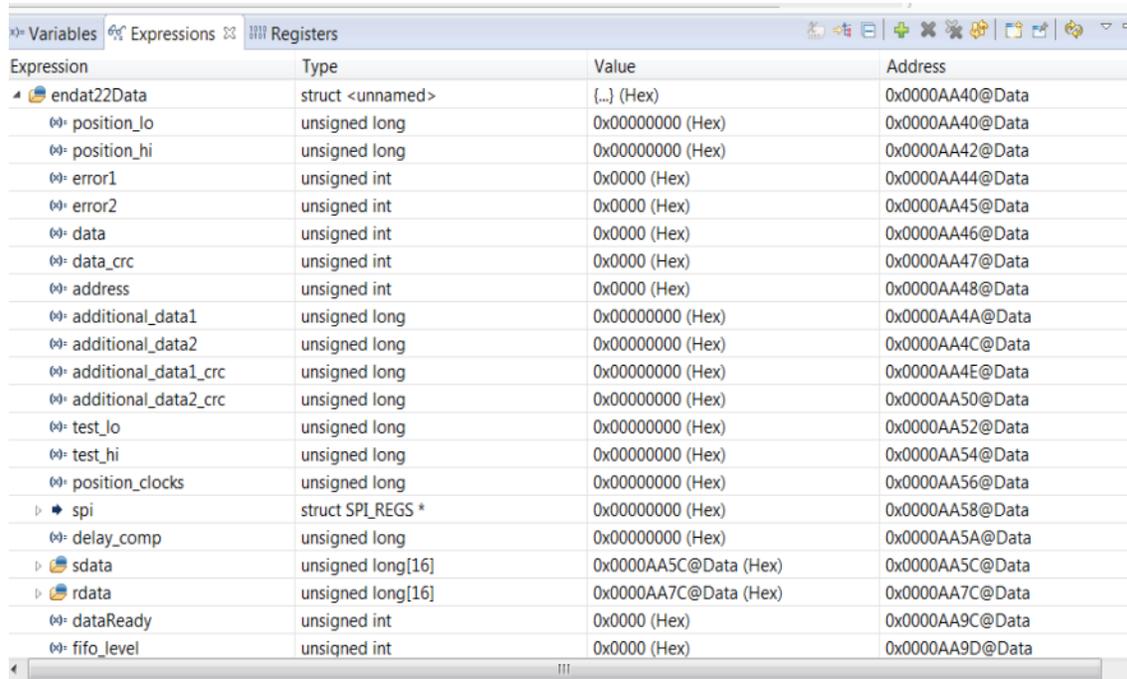
Expression	Type	Value	Address
⊿ 🔵 endat22Data	struct <unnamed>	{...} (Hex)	0x0000AA40@Data
(x)= position_lo	unsigned long	0x00000000 (Hex)	0x0000AA40@Data
(x)= position_hi	unsigned long	0x00000000 (Hex)	0x0000AA42@Data
(x)= error1	unsigned int	0x0000 (Hex)	0x0000AA44@Data
(x)= error2	unsigned int	0x0000 (Hex)	0x0000AA45@Data
(x)= data	unsigned int	0x0000 (Hex)	0x0000AA46@Data
(x)= data_crc	unsigned int	0x0000 (Hex)	0x0000AA47@Data
(x)= address	unsigned int	0x0000 (Hex)	0x0000AA48@Data
(x)= additional_data1	unsigned long	0x00000000 (Hex)	0x0000AA4A@Data
(x)= additional_data2	unsigned long	0x00000000 (Hex)	0x0000AA4C@Data
(x)= additional_data1_crc	unsigned long	0x00000000 (Hex)	0x0000AA4E@Data
(x)= additional_data2_crc	unsigned long	0x00000000 (Hex)	0x0000AA50@Data
(x)= test_lo	unsigned long	0x00000000 (Hex)	0x0000AA52@Data
(x)= test_hi	unsigned long	0x00000000 (Hex)	0x0000AA54@Data
(x)= position_clocks	unsigned long	0x00000000 (Hex)	0x0000AA56@Data
▷ ➔ spi	struct SPI_REGS *	0x00000000 (Hex)	0x0000AA58@Data
(x)= delay_comp	unsigned long	0x00000000 (Hex)	0x0000AA5A@Data
▷ 🔵 sdata	unsigned long[16]	0x0000AA5C@Data (Hex)	0x0000AA5C@Data
▷ 🔵 rdata	unsigned long[16]	0x0000AA7C@Data (Hex)	0x0000AA7C@Data
(x)= dataReady	unsigned int	0x0000 (Hex)	0x0000AA9C@Data
(x)= fifo_level	unsigned int	0x0000 (Hex)	0x0000AA9D@Data

Figure 16. Configuring the Expressions Window

Alternately, a group of variables can be imported into the Expressions window, by right clicking in the Expressions Window and clicking Import, then, browse to the .txt file containing these variables. Browse to the root directory of the project and pick 'PM_endat22_BasicPosAcc_DelComp_VAR.txt' and click OK to import the variables shown in Figure 16.

2. Click on the Continuous Refresh button 🔄 in the watch window. This enables the window to run with real-time mode. By clicking the down arrow in this watch window, you can select "Customize Continuous Refresh Interval" and edit the refresh rate of the watch window. Note that choosing too fast an interval may affect performance.

7.4   Run the Code

1. Run the code by pressing the Run Button ▶ in the Debug Tab.
2. The project should run and the values in the watch window should continuously update.
3. Once complete, reset the processor (Run → Reset → CPU Reset) 🐞 and terminate the debug session by clicking (Run → Terminate) ■. This halts the program and disconnects Code Composer from the MCU.
4. As the encoder sends the position information, observe the same in the watch window as endat22Data.position_hi, endat22Data.position_lo variables. The variable endat22Data.position_hi would only change if the encoder sends more than 32 bits of position information.
5. If the encoder is not mounted on a spinning Motor, you can manually rotate the encoder shaft and observe the position changing in the watch window.

7.5 Next Steps

It is not necessary to terminate the debug session each time the user changes or runs the code again. Instead, the following procedure can be followed:

1. After rebuilding the project, (Run → Reset → CPU Reset) 🖛, (Run → Restart) 🖛, and enable real-time options. Once complete, disable real-time options and reset the CPU. Terminate the project if the target device or the configuration is changed (Ram to Flash or Flash to Ram) and prior to shutting down CCS.

2. Customize the project to meet your encoder and application requirements. Change the encoder type in the encoder.h file. You can also change the EnDat Clock frequency as instructed in the *C2000 Position Manager EnDat22 Library Module User's Guide* (SPRUI35).

3. Modify the example code as needed and perform tests with different cable lengths.

8 References

- *DesignDRIVE Development Kit IDDK Hardware Reference Guide* (SPRUI23)
- *C2000 Position Manager EnDat22 Library Module User's Guide* (SPRUI35)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers' products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers' products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have *not* been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		Applications	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		