Application Note

MCU Control Center: Clinic

Naren Siddharth M, Nima Eskandari

ABSTRACT

The MCU Control Center Clinic is a GUI-based diagnostic tool designed to diagnose and troubleshoot device and peripheral configuration issues at runtime. The tool helps users avoid common pitfalls such as overclocking, underclocking, and incorrect register configurations, which can lead to instability, unknown errors, and potential hardware damage. The clinic tool also helps check the device errata conditions and provides a visual report of errors, along with direct links to the TI.com documentation. The Clinic Diagnostic Tool can be used as a standalone application and is also seamlessly integrated into Code Composer Studio (CCS), making the tool accessible in every project.

Table of Contents

1 Introduction....2
2 Clinic Tool Overview....3
3 Getting Started with the Clinic Tool....5
3.1 Setting up the Clinic Diagnostic Tool Within CCS for a New Project....5
3.2 Setting up the Clinic Diagnostic Tool within CCS for an Existing Project....11
4 Using the Clinic Diagnostic Tool....15
4.1 Diagnosing Clocking Issues....15
4.2 Diagnosing Errata Conditions....16
4.3 Diagnosing Peripheral Errors....17
5 Supported Devices....19
6 Summary....19
7 References....19

Trademarks

All trademarks are the property of their respective owners.

Copyright © 2025 Texas Instruments Incorporated

1 Introduction

In most applications, it is common for users to make configuration mistakes when developing embedded software. Clocking issues (overclocking or underclocking), incorrectly configuring peripherals and system control settings are among the commonly seen issues when evaluating embedded software for a given device. Encountering an errata condition can also be the cause of device exhibiting unpredictable behavior. The MCU Control Center Clinic is a GUI-Based diagnostic tool designed to diagnose and troubleshoot device and peripheral configuration issues at runtime. This means that the tool automatically reads the relevant device registers out of the device memory, performs the necessary checks and updates a graphical user interface to notify the user of errors. The tool also provides a direct link to specific location in device documentation (data sheet, technical reference manual and errata).

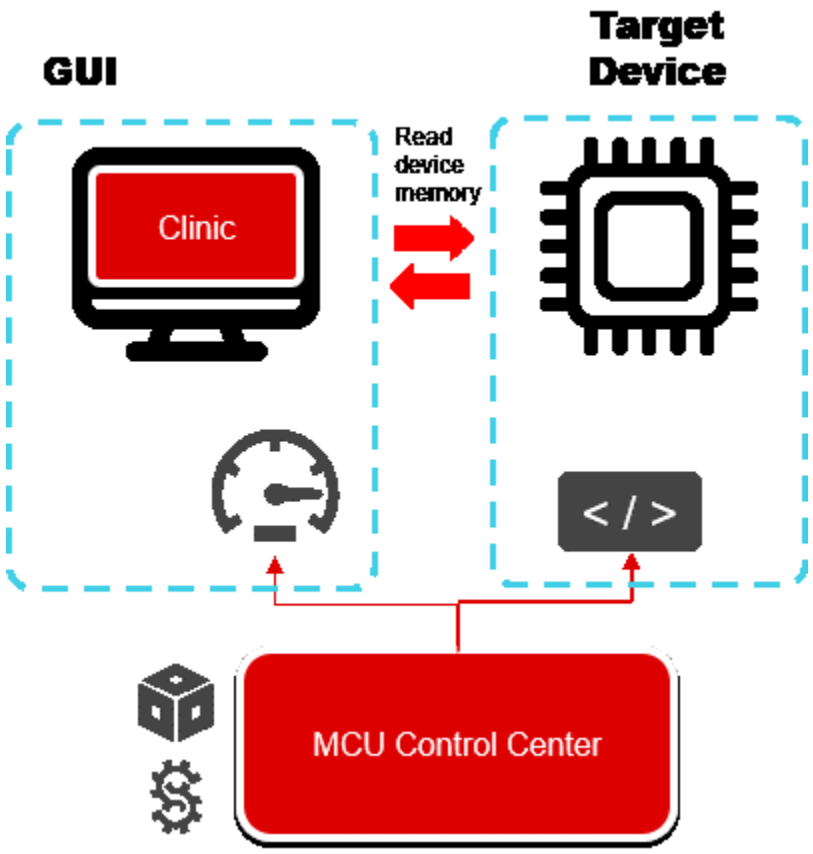

Figure 1-1. Clinic Overview

Copyright © 2025 Texas Instruments Incorporated

2 Clinic Tool Overview

The clinic tool supports reading the device memory through JTAG communication (debugger). When using the tool in Code Composer Studio (CCS), the same debugger connection can be used to both debug the device in CCS and update the MCU Control Center GUI. Launching the GUI once the device is connected in CCS establishes the communication and allows for the tool to access device memory. An example of the MCU Control Center GUI with Clinic is shown in Figure 2-1.

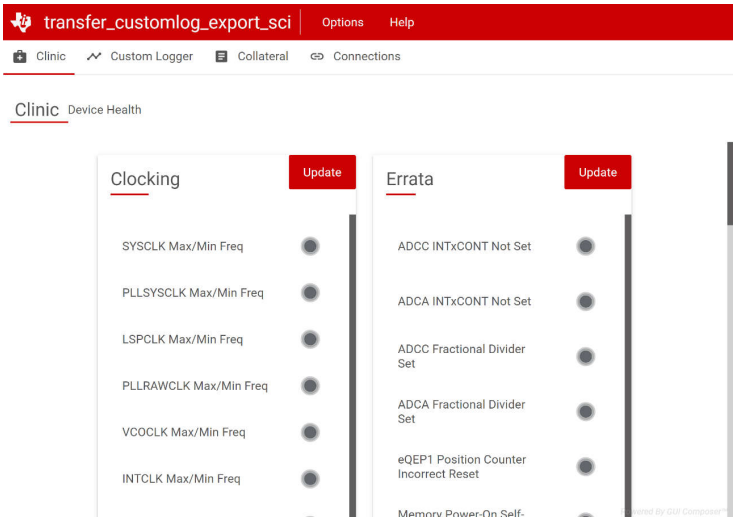

Figure 2-1. MCU Control Center Clinic Example

Clicking the *Update* button allows the GUI to read the relevant memory locations of the device and perform the required checks. Once completed, if any violation is detected, the LED next to each entry turns red.

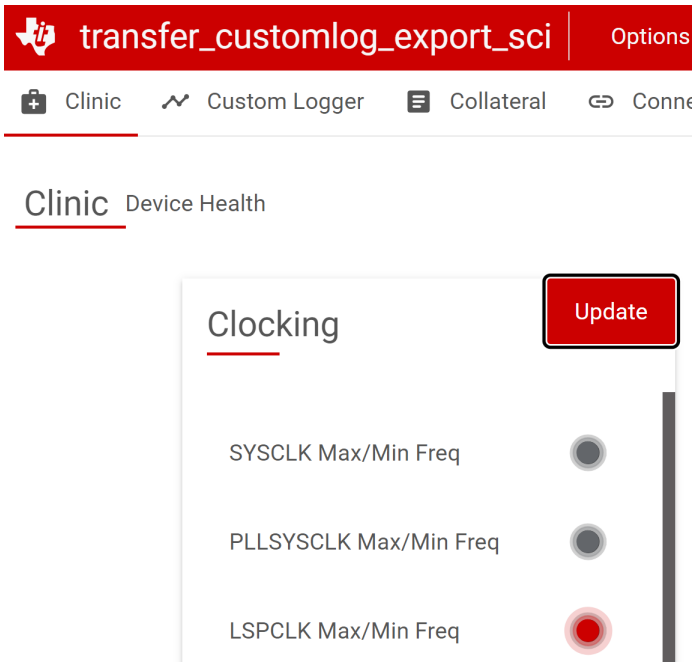

Figure 2-2. Clinic Error Detected

When running in CCS, the user can run the Clinic checks as many times as required. The user can also stop the code at any given location in the software to check the status of device and the peripherals. This allows the user to understand when and where the incorrect configuration has occurred.

Copyright © 2025 Texas Instruments Incorporated

The tool also showcases the status registers for most peripherals. For example, if a communication peripheral encounters an error, this is not known to the user unless the specific status register of that peripheral is checked.

Figure 2-3 is an example of the SCI peripheral (serial communication interface in C2000 devices) and the status registers. In case of an error, the LEDs next to each item turns red.

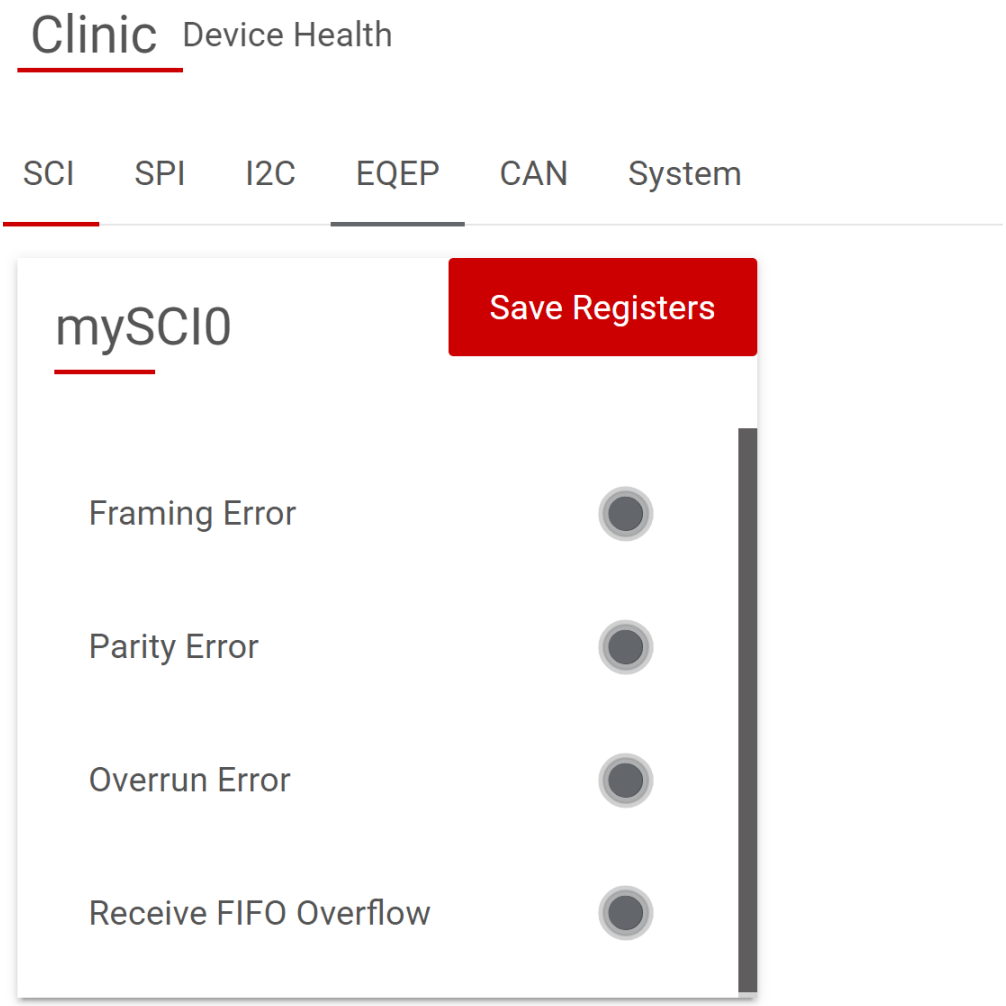

Figure 2-3. Peripheral Status Registers

Copyright © 2025 Texas Instruments Incorporated

3 Getting Started with the Clinic Tool

This section is intended to help new users to get started quickly with the clinic diagnostic tool.

The user can setup the clinic tool with an existing project or with a new project. Both scenarios are discussed in the following sections.

3.1 Setting up the Clinic Diagnostic Tool Within CCS for a New Project

First, the user must import the project into the CCS workspace.

3.1.1 Importing the Project into CCS

To get started, open the transfer example project from the Driverlib based examples available in <C2000WARE_INSTALL>/driverlib/<device>/examples.

1. In the CCS menu, click *Project* > *Import CCS Projects*
2. Enter the path to the example project in the *Select search-directory* and click *Refresh* or select the *Browse* button to choose the proper folder directly.
3. Select the project required.
4. Click *Finish*.

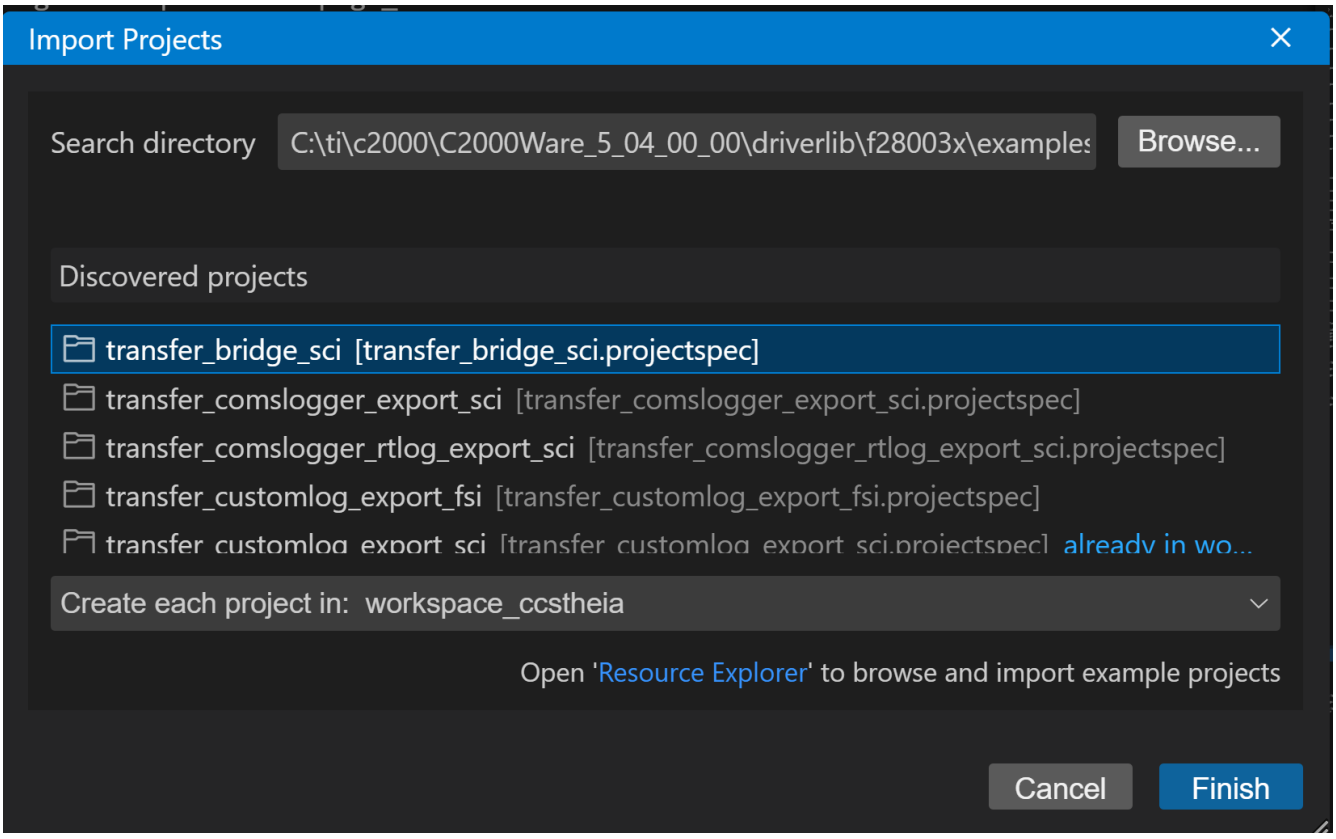

Figure 3-1. Import Project

Next, the project properties must be updated.

Copyright © 2025 Texas Instruments Incorporated

3.1.2 Updating GUI Variable

Follow the steps below to change the project properties.

1. Right click on the project and select *Project Properties*.
2. Under *General*, select *Variables*.
3. Click on *GUI_SUPPORT* variable to edit the variable.
4. Change the *Value* to *1* and click *OK*.
5. Click Save and Close.

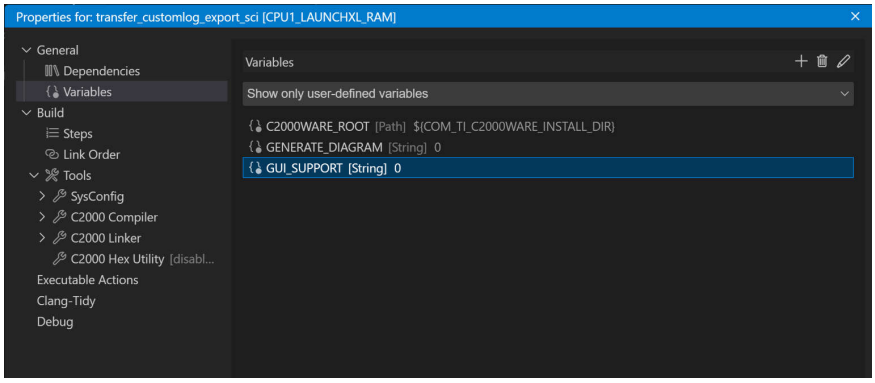

Figure 3-2. Project Properties Before GUI Variable Update

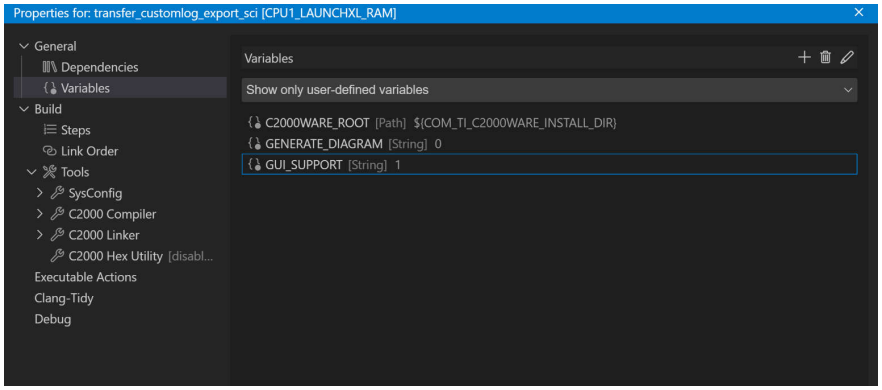

Figure 3-3. Project Properties After GUI Variable Update

3.1.3 Configuring the Clinic Diagnostic Tool in SysConfig

To open the SysConfig tool, double-click on the *.syscfg* file to be edited in the CCS Project Explorer window.

Note

MCU Mission Control has been renamed to MCU Control Center.

1. Under *MCU MISSION CONTROL AND TRANSFER*, select *MCU MISSION CONTROL* to enable the program.
2. Select *Use debugger for device diagnostic export* option as the *Clinic - Device Diagnostic Export Mode*.
3. Enable *Custom Export Logger*.
4. Change the name of the Clinic Tool, by modifying the *GUI Project Name* value.
5. Save the .syscfg file.

MCU Mission Control and Custom Export Logger are enabled by default in a transfer example project.

Copyright © 2025 Texas Instruments Incorporated

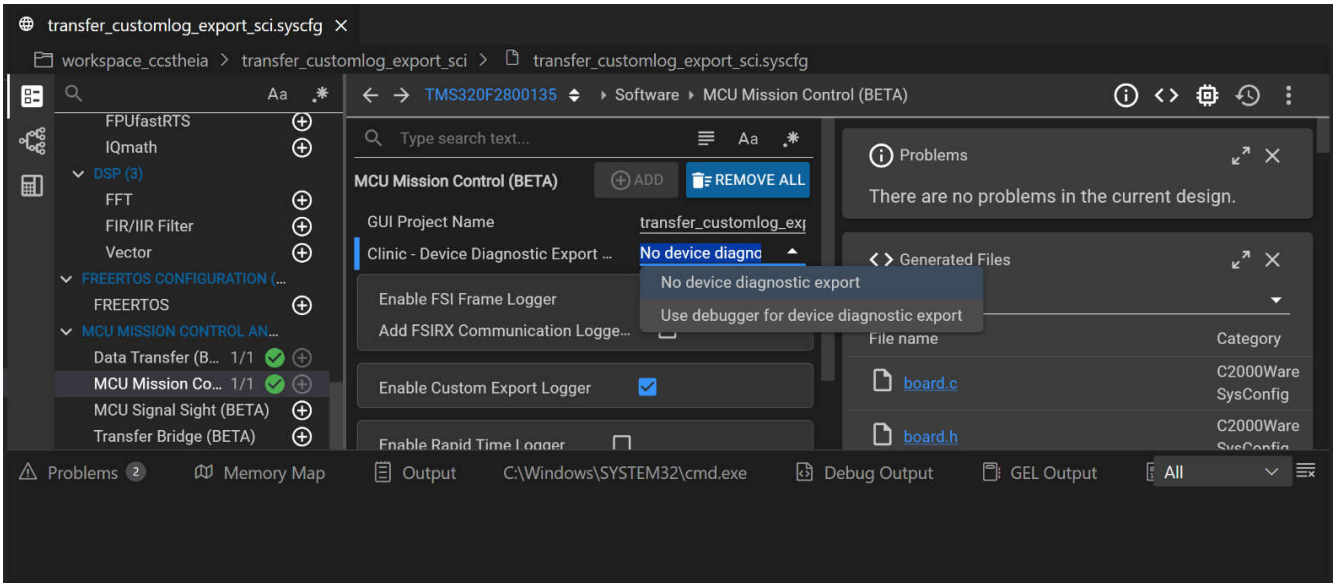

Figure 3-4. Enabling Clinic Tool in SysConfig

3.1.4 Opening the Clinic Diagnostic Tool

Once all the steps above are successfully completed, save the project. Now, to launch the Clinic Tool,

1. Rebuild the project.
2. Open the *Command Palette*.
3. Run the *Refresh Plugins* command.
4. Then, click *View* and under *Plugins*, select the Clinic Tool to launch the tool (available with the name configured in the .syscfg file).

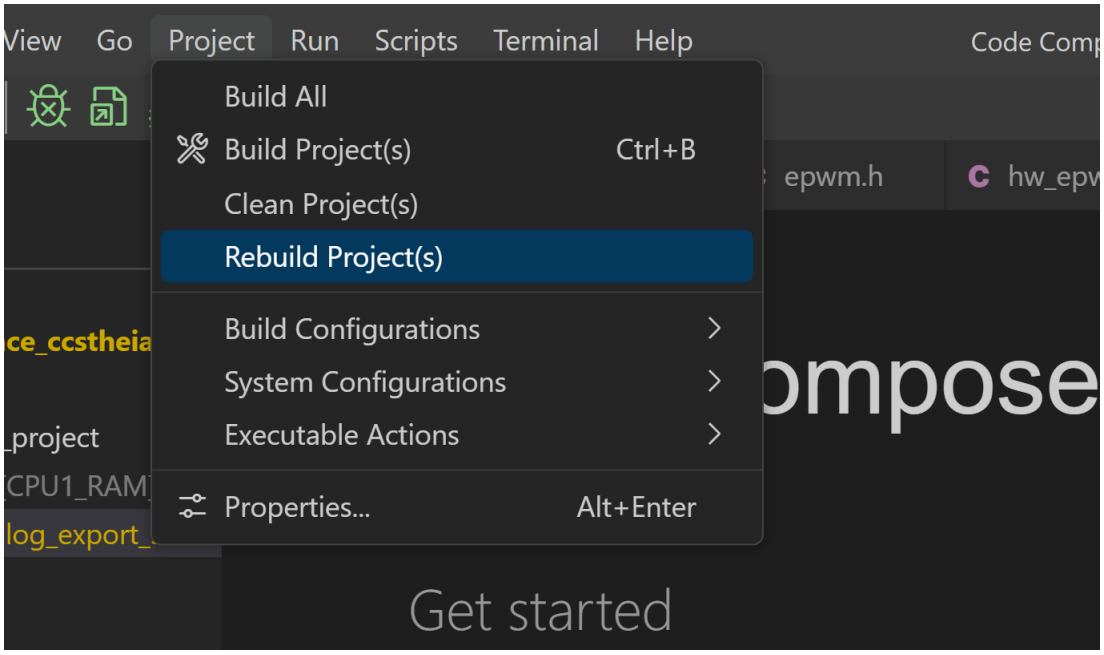

Figure 3-5. Build Project

Copyright © 2025 Texas Instruments Incorporated

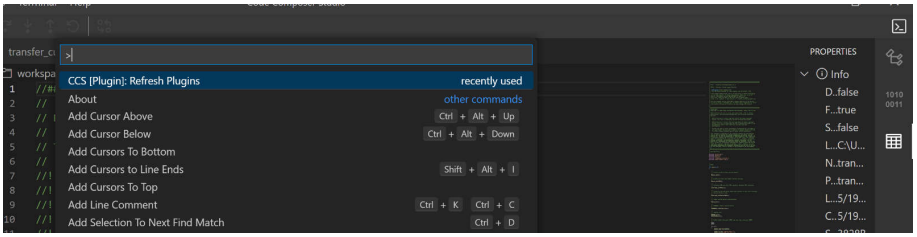

Figure 3-6. Refresh Plugins

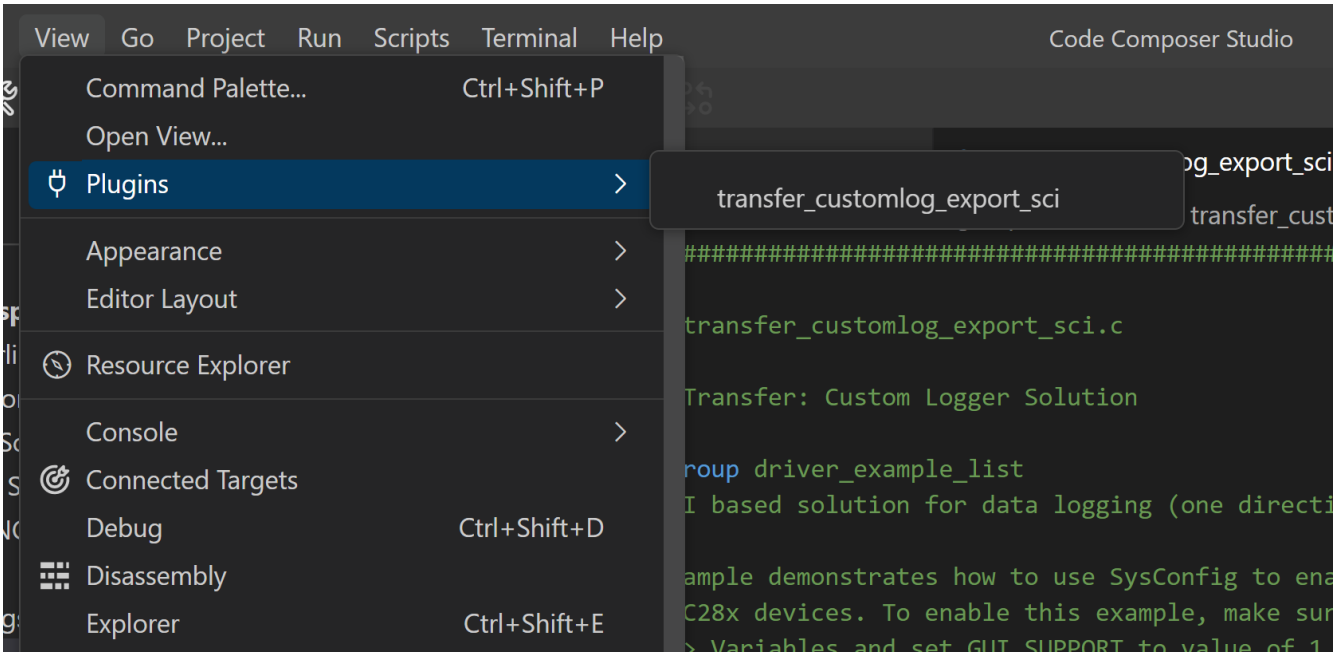

Figure 3-7. Run Plugin

Copyright © 2025 Texas Instruments Incorporated

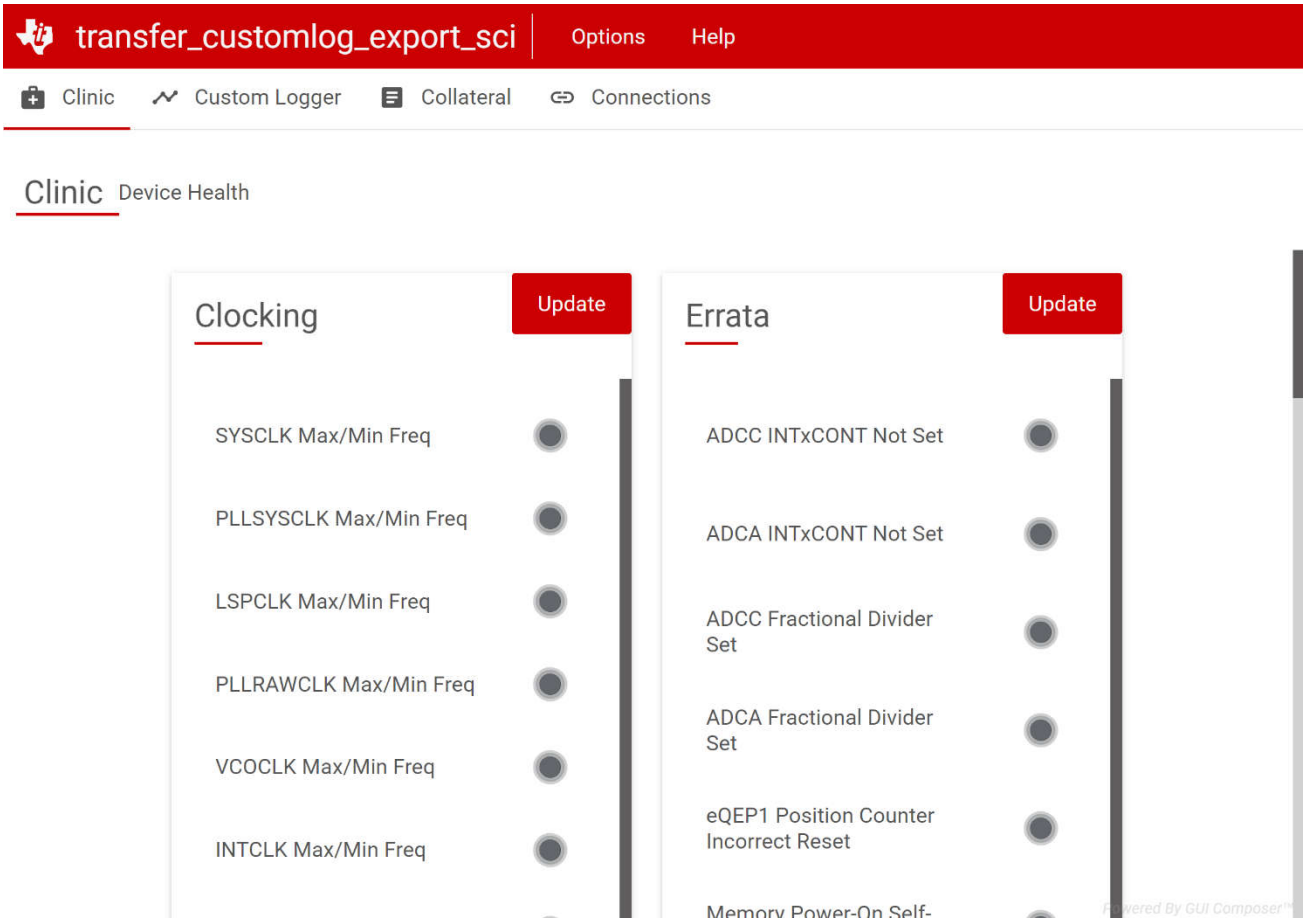

Figure 3-8. MCU Control Center Example Overview

Copyright © 2025 Texas Instruments Incorporated

3.1.5 Verifying Connection to the Target Device

To make sure that the clinic diagnostic tool can successfully connect to the target device, follow these steps:

1. Open the *Connections* section, where a user can view the current connection status to the target device.
2. A successful connection to the XDS Debug Probe on the target device is indicated by the LED lighting up, accompanied by a corresponding log message stating *Hardware connected*.
3. In the event of a connection issue, the log messages provide error reports to assist in troubleshooting and resolving the problem.

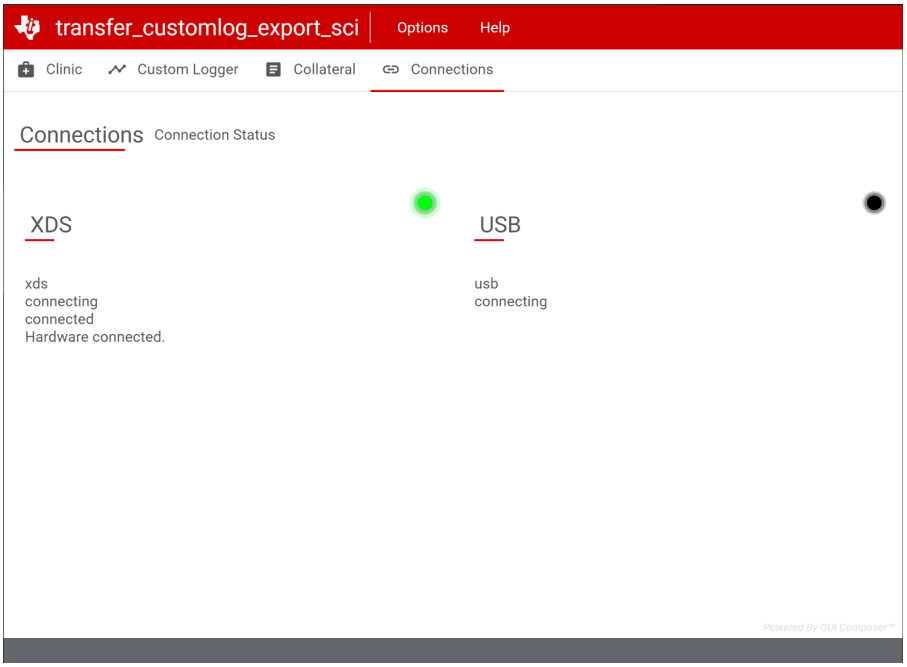

Figure 3-9. Connection Status

Copyright © 2025 Texas Instruments Incorporated

3.2 Setting up the Clinic Diagnostic Tool within CCS for an Existing Project

The clinic tool can be set up for an existing project by following the steps in this section.

3.2.1 Updating GUI Variable

1. Right click on the project and select *Project Properties*.
2. Under *General*, select *Variables*.
3. Click on *GUI_SUPPORT* variable to edit it.
4. Change the *Value* to *1* and click *OK*.
5. Click *Save and Close*.

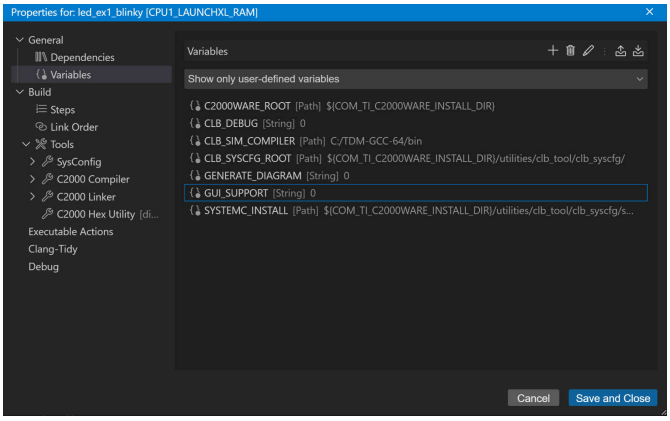

Figure 3-10. Change GUI_SUPPORT Variable

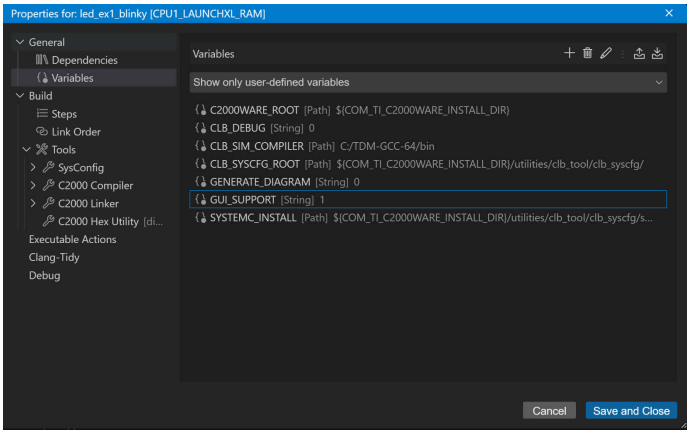

Figure 3-11. GUI_SUPPORT variable set to 1

3.2.2 Configuring the Clinic Diagnostic Tool in SysConfig

To open the SysConfig tool, double-click on the *syscfg* file to be edited in the CCS Project Explorer window.

Note

MCU Mission Control was renamed to MCU Control Center.

1. Under *MCU CONTROL CENTER AND TRANSFER*, select *MCU Control Center* to enable the MCU Control Center.
2. Select *Use debugger for device diagnostic export* option as the *Clinic - Device Diagnostic Export Mode*.
3. Enable *Custom Export Logger.*
4. Change the name of the Clinic Tool, by modifying the *GUI Project Name* value.
5. Save the .syscfg file.

Copyright © 2025 Texas Instruments Incorporated

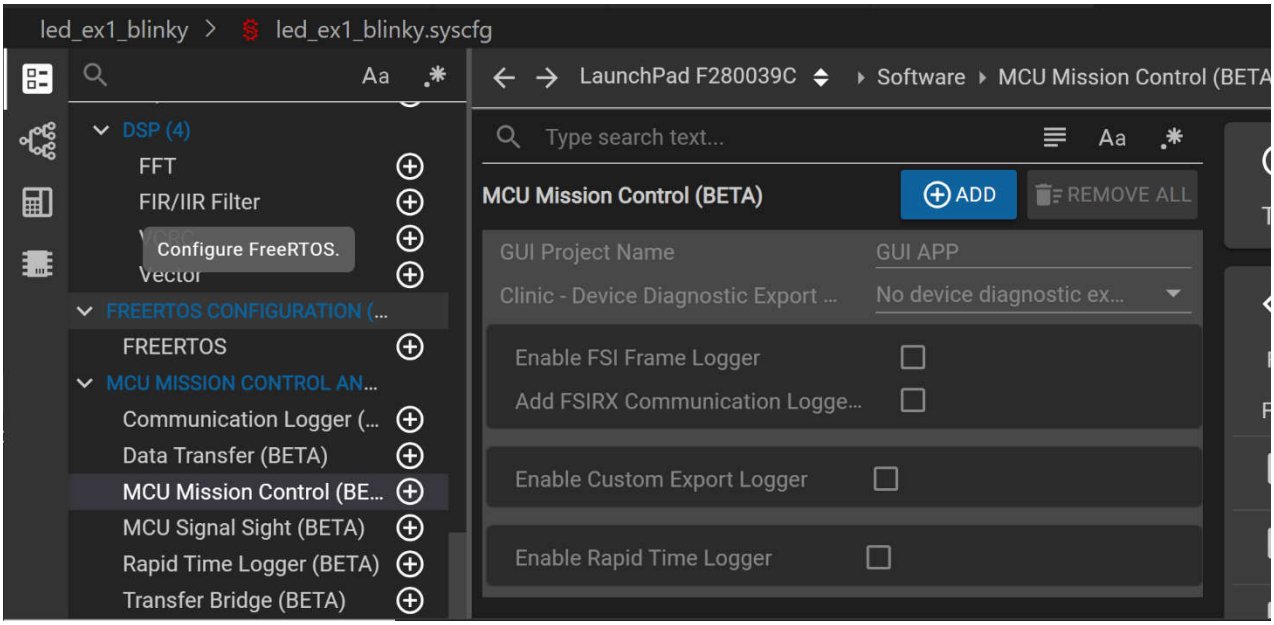

Figure 3-12. Enable and Add MCU Control Center

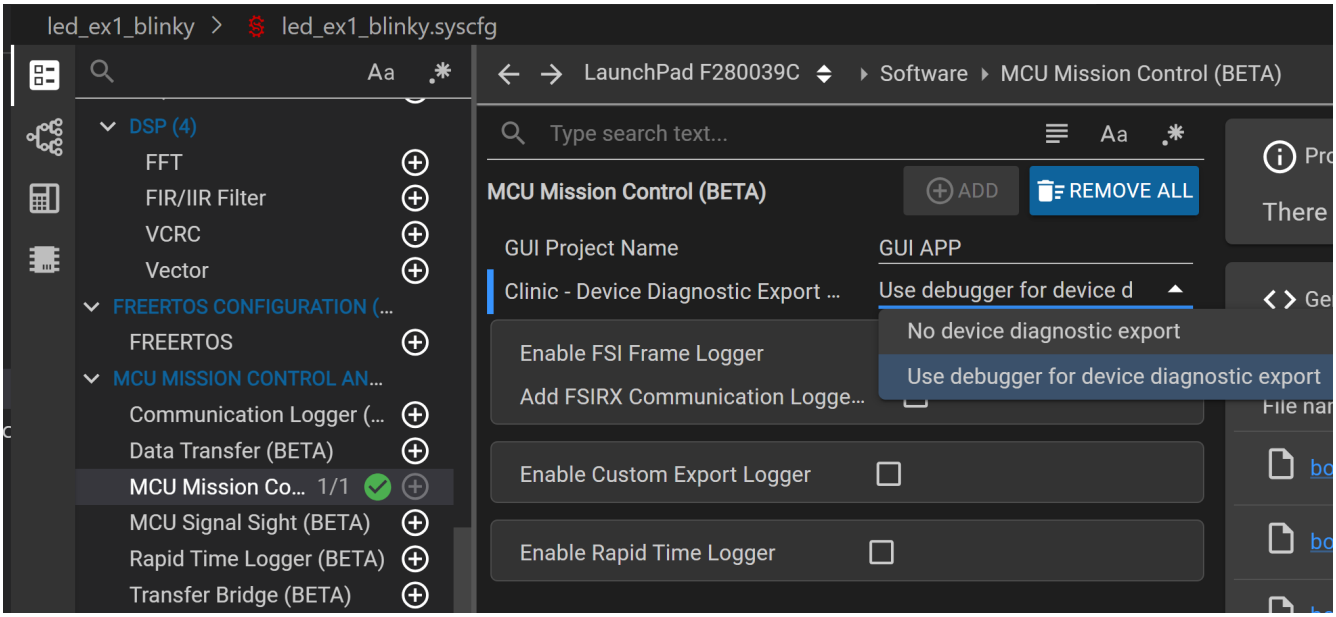

Figure 3-13. Enable Clinic Tool in SysConfig

3.2.3 Opening the Clinic Diagnostic Tool

Once all the steps above are successfully completed, save the project. Now, to launch the Clinic Tool.

1. Rebuild the project.
2. Open the *Command Palette*.
3. Run the *Refresh Plugins* command.
4. Then, click *View* and under *Plugins*, select the Clinic Tool to launch the plugin (available with the name configured in the .syscfg file).

Copyright © 2025 Texas Instruments Incorporated

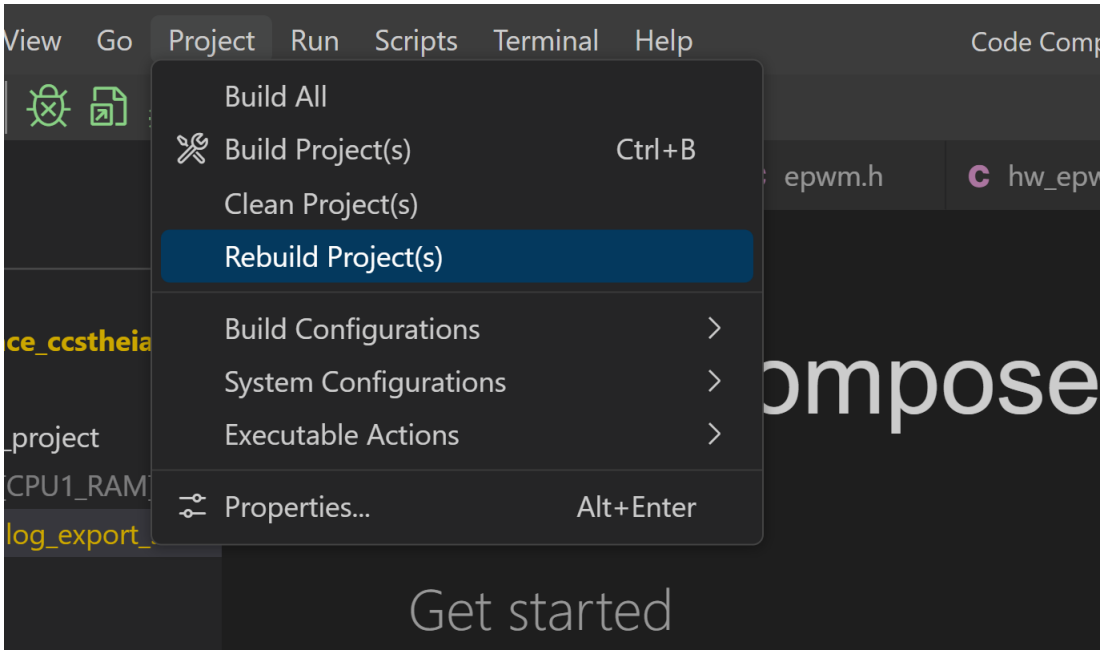

Figure 3-14. Build Project

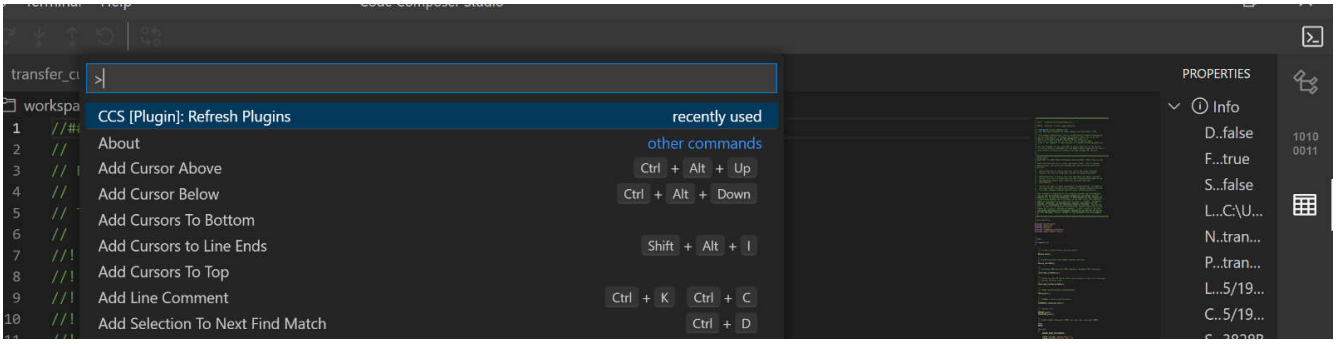

Figure 3-15. Refresh Plugins

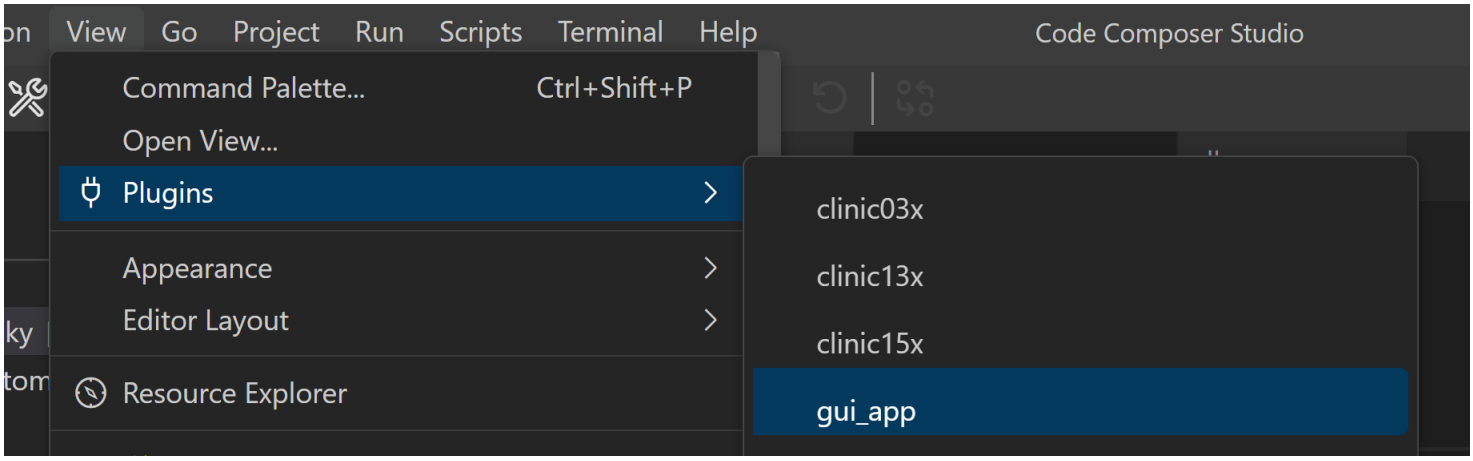

Figure 3-16. Run GUI APP Plugin

Copyright © 2025 Texas Instruments Incorporated

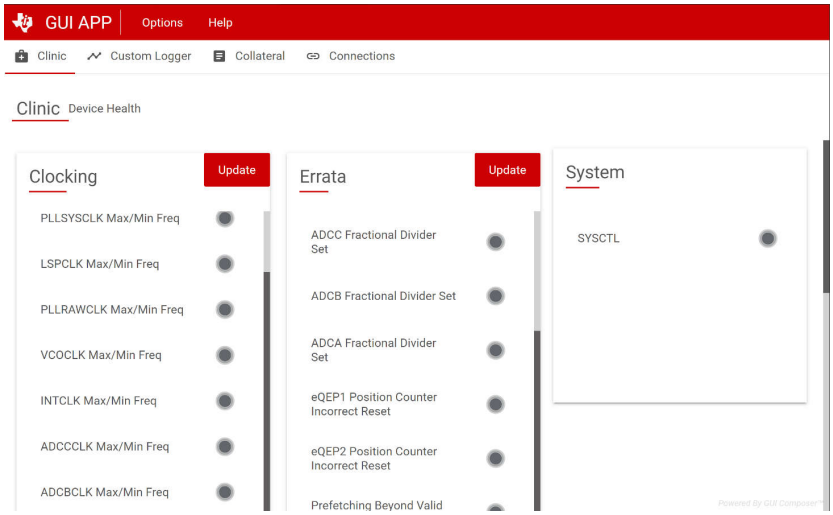

Figure 3-17. Run Clinic for Existing Project

3.2.4 Verifying Connection to the Target Device

To ensure that the clinic diagnostic tool can successfully connect to the target device, follow these steps:

1. Open the *Connections* section, where the current connection status to the target device can be viewed.
2. A successful connection to the XDS Debug Probe on the target device will be indicated by the LED lighting up, accompanied by a corresponding log message stating *Hardware connected*.
3. In the event of a connection issue, the log messages provide error reports to assist in troubleshooting and resolving the problem.

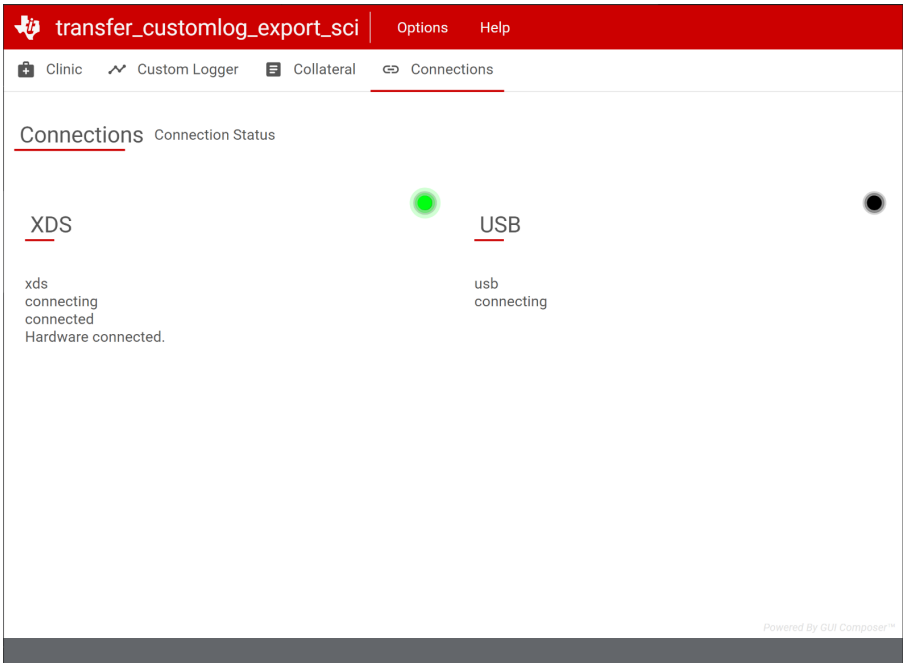

Figure 3-18. Connection Status

Copyright © 2025 Texas Instruments Incorporated

4 Using the Clinic Diagnostic Tool

This section describes how to use the clinic diagnostic tool to diagnose overclocking, possible errata and peripheral errors.

4.1 Diagnosing Clocking Issues

The Clinic Diagnostic Tool provides a convenient and efficient way to identify and troubleshoot clocking issues during runtime. To utilize this feature, follow these steps:

1. Launch the Clinic Tool to access features and functionalities.
2. Start a CCS debug session and make sure that the target device is successfully connected to the clinic tool.
3. Set breakpoints at specific points in the program where potential clocking issues must be investigated.
4. Run the program and wait for the program to stop at the designated breakpoint or halt due to an error.
5. In the clinic tool, navigate to the *Clocking* section and click the *Update* button to refresh the clocking information and highlight any issues, which are indicated by an LED beside the affected clock.

The clinic tool indicates clocks that are misconfigured by lighting up the LED beside the affected clock. This visual cue allows a user to quickly identify and address clocking issues, making sure that the system operates correctly and efficiently.

```
115     SysCtl_setPLLSysClk(divider: 10);
116     EALLOW;
117     HWREG(CLKCFG_BASE + SYSCTL_O_LOSPCP) = 7U;
118     SYSCTL_REGWRITE_DELAY;
119     EDIS;
120
121     DEVICE_DELAY_US(5000);
```

Figure 4-1. Halt at a Breakpoint

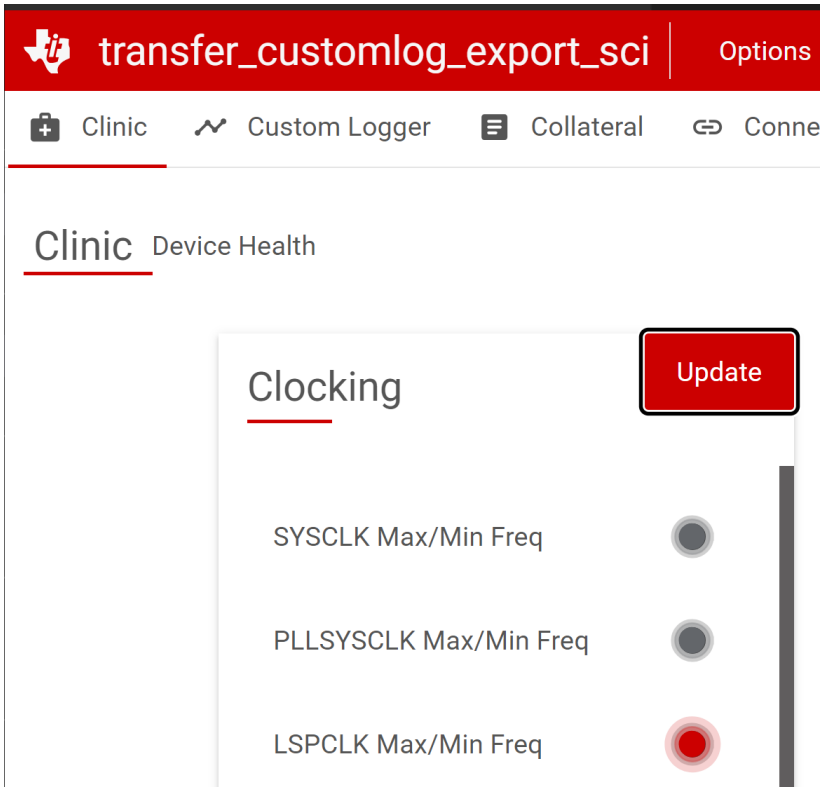

Figure 4-2. Clocking Errors Identified

Copyright © 2025 Texas Instruments Incorporated

4.2 Diagnosing Errata Conditions

The clinic diagnostic tool offers a powerful feature for detecting and diagnosing errata conditions, allowing users to quickly identify and resolve potential problems. To use this feature, follow these steps:

1. Launch the Clinic Tool to access features and functionalities.
2. Start a CCS debug session and make sure that the target device is successfully connected to the clinic tool.
3. Set breakpoints at specific points in the program where potential errata can be investigated.
4. Run the program and wait for the program to stop at the designated breakpoint or halt due to an error.
5. In the clinic tool, navigate to the *Errata* section and click the *Update* button to refresh the Errata information and highlight any potential errata configurations, which is indicated by an LED beside the errata.

Additionally, click on the corresponding errata name, which is a hyperlink that directs the user to the relevant section in the errata document of the device, providing further information and guidance on resolving the issue. This visual cue and direct access to documentation enable the user to quickly identify and address errata, making sure that the system operates correctly and efficiently.

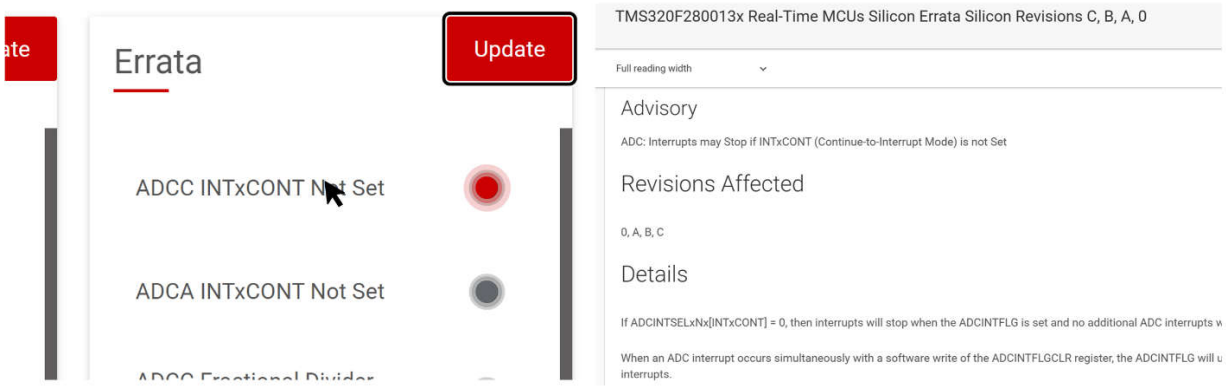

Figure 4-3. Errata Condition and Corresponding Section in Errata Document

Copyright © 2025 Texas Instruments Incorporated

4.3 Diagnosing Peripheral Errors

The Clinic Diagnostic Tool provides a comprehensive view of all peripherals that have been used or configured in the project, along with the corresponding error status registers. To access this feature, follow these steps:

1. Launch the Clinic Tool to access the features and functionalities.
2. Start a CCS debug session and make sure that the target device is successfully connected to the clinic tool.
3. Set breakpoints at specific points in the program where the user wants to investigate potential issues, if required.
4. Run the program and wait for the program to stop at the designated breakpoint or stop due to an error.
5. In the Clinic Tool, navigate to the corresponding section for the peripheral of interest, where a user can view all the error status registers associated with that peripheral.

The clinic tool displays the error status registers for each peripheral, with an LED indicator that lights up when an error is detected, making diagnosing peripheral-specific errors simpler. These error status registers are updated automatically, without requiring any manual intervention.

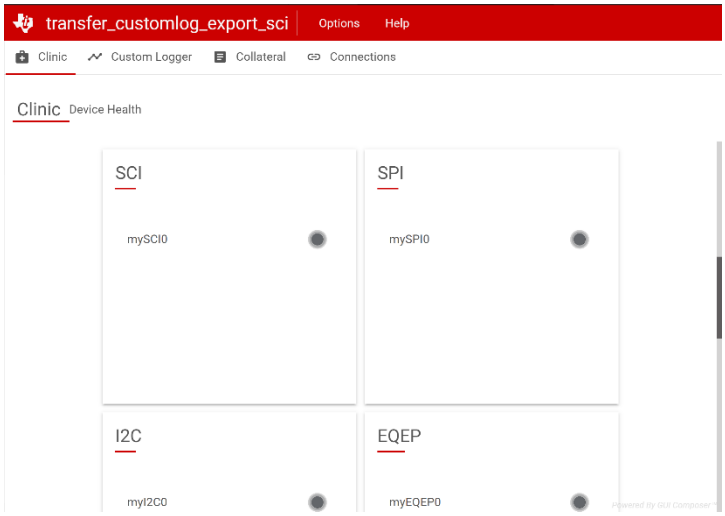

Figure 4-4. Peripherals Overview Status

Copyright © 2025 Texas Instruments Incorporated

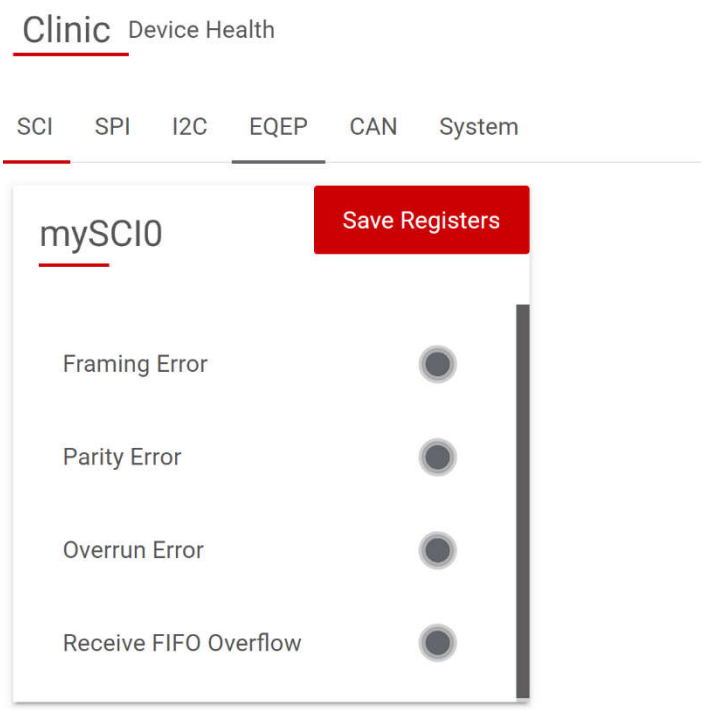

Figure 4-5. Peripheral Specific Errors

Furthermore, the clinic tool offers an additional feature that allows the user to retrieve the data of all registers associated with a peripheral. By clicking the *Save Registers* button, a user can obtain a snapshot of all register values for the peripheral at a specific point in time, providing valuable insights for debugging and troubleshooting purposes.

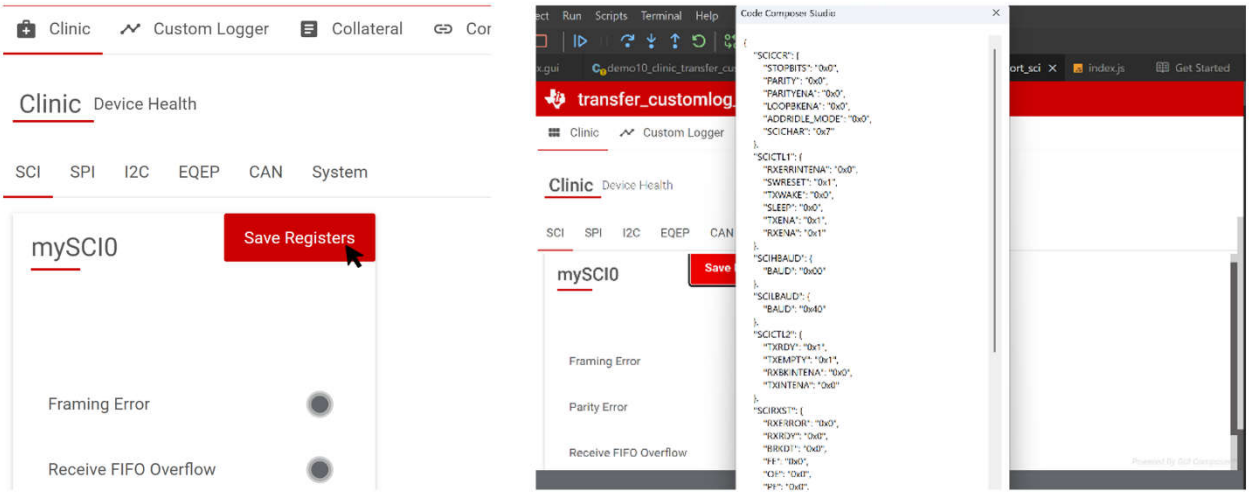

Figure 4-6. Saving a Peripheral or Module Registers

Copyright © 2025 Texas Instruments Incorporated

5 Supported Devices

The following list shows currently supported devices.

- F280013x
- F280015x
- F28003x

Additional devices are supported in the subsequent versions of the tool.

6 Summary

The MCU clinic tool allows for automatic diagnostic of application software and device configuration. The tool can be run during development to make sure that errors are not introduced during the software development. The tool can also be run on existing applications as a check to see if existing designs are violating any errata conditions. The tool is provided as integrated in CCS and SysConfig but can also be used as a standalone tool.

7 References

1. Texas Instruments, *MCU Control Center Tool Developer's Guide* application note.
2. Texas Instruments, *MCU Signal Sight Tool Developer's Guide* application note.
3. Texas Instruments, *TMS320F280013x Real-Time MCUs Silicon Errata*, errata.
4. Texas Instruments, *TMS320F280013x Real-Time Microcontrollers*, data sheet.
5. Texas Instruments, *TMS320F280015x Real-Time MCUs Silicon Errata*, errata.
6. Texas Instruments, *TMS320F280015x Real-Time Microcontrollers*, data sheet.
7. Texas Instruments, *TMS320F28003x Real-Time MCUs Silicon Errata Silicon Revision 0*, errata.
8. Texas Instruments, *TMS320F28003x Real-Time Microcontrollers*, data sheet.

Copyright © 2025 Texas Instruments Incorporated

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2025, Texas Instruments Incorporated