Application Note

C2000 IDE Assist Tool Features Guide

Nima Eskandari, Aishwarya Rajesh, Delaney Woodward

ABSTRACT

C2000™ IDE Assist is a first-of-its-kind tool integrated within Visual Studio Code™ and Code Composer Studio™, designed to enhance development for Texas Instruments C2000 MCUs. C2000 IDE provides an easier way to code, debug, and migrate C2000 application code with features already available in your development environment. This all-in-one tool speeds up the various steps of the software production process from code initialization to runtime events to debug profiling which includes ease of migration, targeted collateral find-ability, software template support, and more. Integrated into Code Composer Studio, C2000 IDE offers easy access to various engineering tools, making the extension a comprehensive solution for C2000 application implementation.

Table of Contents

Trademarks

1 Introduction

The Texas Instruments C2000 software development kits (SDK) provide a cohesive suite of software and documentation designed to accelerate development of real-time control applications. These resources include device-specific drivers, libraries, and peripheral examples, helping developers streamline projects. The SDKs are built to accommodate users of all experience levels, with beginner-friendly features to guide first-time users. The C2000 SysConfig tool, included in the SDKs, enhances the setup process by simplifying device and peripheral complexities through an intuitive visual interface. To further improve the initialization, run-time, and debug user experience, the C2000 IDEA centralizes tools and collateral within a single environment for a more efficient workflow.

This features guide discusses how to get started with and utilize the C2000 IDE Assist to simplify the overall software implementation when used with the C2000 SDKs.

2 Getting Started

This section introduces the basics of the C2000 IDEA tool such as how to setup the development environment and basic commands to utilize the extension.

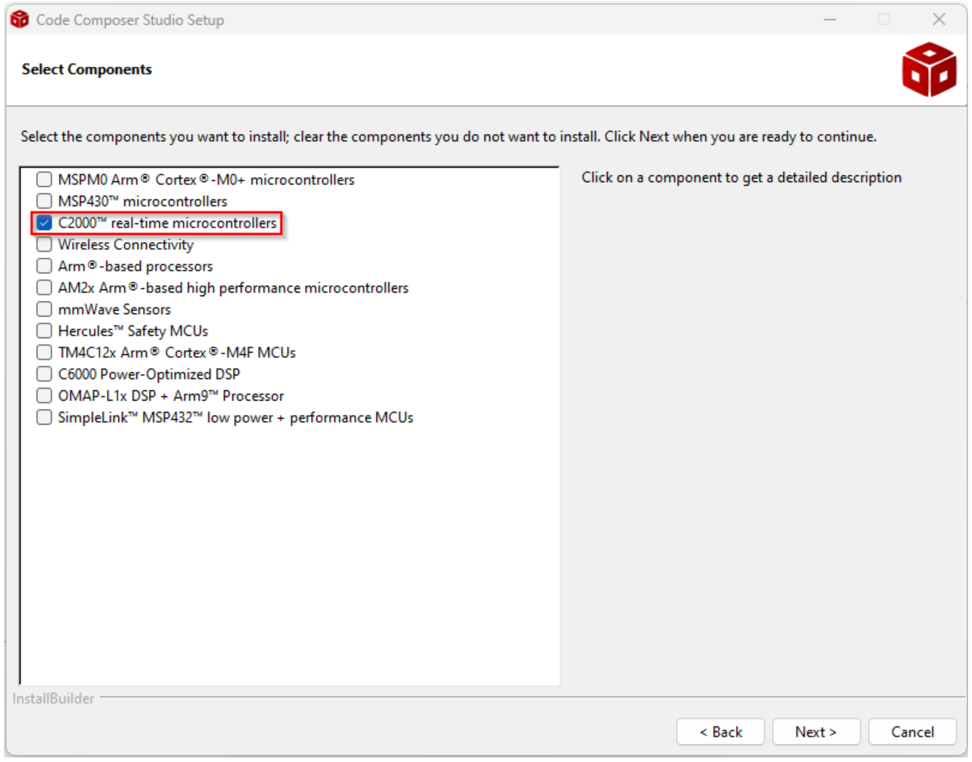
2.1 Software Installation

This section details the software installation procedure to enable the extension in your development environment.

2.1.1 Install Code Composer Studio IDE

Follow these steps to install TI's integrated development environment (IDE) for microcontrollers and processors:

1. Download the latest offline installer for Code Composer Studio (CCS):
   a. Default Recommended Path - C:/ti/ccs200X
2. Once *Select Components* is reached, select *C2000 real-time MCUs*.
3. Continue through the steps until installation is complete.
4. Launch CCS.


Code Composer Studio Setup

Select Components

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [ ] MSPM0 Arm® Cortex®-M0+ microcontrollers
- [ ] MSP430™ microcontrollers
- [✓] C2000™ real-time microcontrollers
- [ ] Wireless Connectivity
- [ ] Arm®-based processors
- [ ] AM2x Arm®-based high performance microcontrollers
- [ ] mmWave Sensors
- [ ] Hercules™ Safety MCUs
- [ ] TM4C12x Arm® Cortex®-M4F MCUs
- [ ] C6000 Power-Optimized DSP
- [ ] OMAP-L1x DSP + Arm9™ Processor
- [ ] SimpleLink™ MSP432™ low power + performance MCUs

Click on a component to get a detailed description

Figure 2-1. CCS Setup

Note

This extension is only supported for Code Composer Studio v20.0.0+.

2.1.2 Install C2000 IDEA Extension

Follow these instructions to install the extension to CCS:

1. Navigate to the *Extensions* tab in the left side bar panel.
2. Search for *C2000 IDEA* and select install.

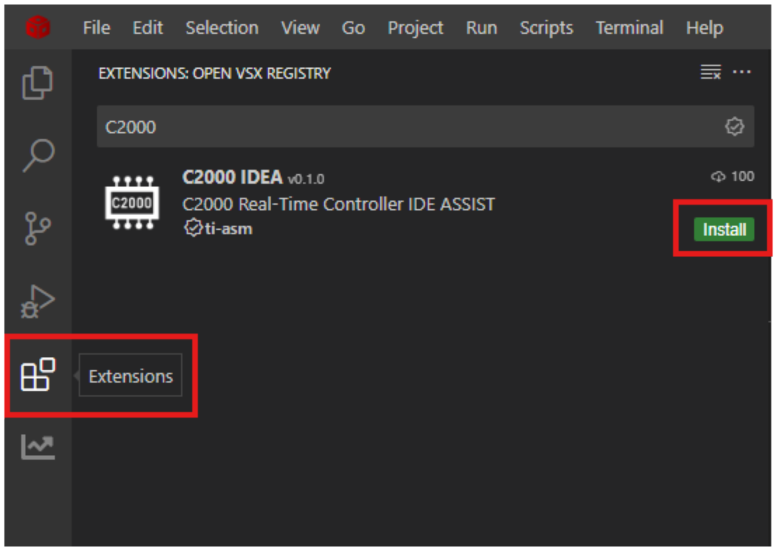
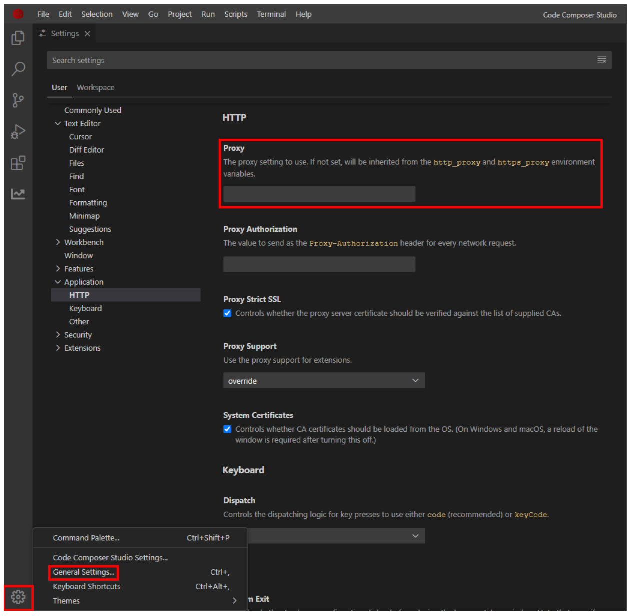


Figure 2-2. C2000 IDEA Setup in CCS

a. If no results are displayed in the Extension Marketplace, your connection may be configured to use a proxy server. To set up the proxy settings:
   i. Click the gear (Manage) icon in the left side bar panel and select *Settings*.
   ii. Navigate to Application > Proxy and input proxy information.



HTTP

Proxy
The proxy setting to use. If not set, will be inherited from the `http_proxy` and `https_proxy` environment variables.

Proxy Authorization
The value to send as the `Proxy-Authorization` header for every network request.

Proxy Strict SSL
✓ Controls whether the proxy server certificate should be verified against the list of supplied CAs.

Proxy Support
Use the proxy support for extensions.
override

System Certificates
✓ Controls whether CA certificates should be loaded from the OS. (On Windows and macOS, a reload of the window is required after turning this off.)

Keyboard

Dispatch
Controls the dispatching logic for key presses to use either `code` (recommended) or `keyCode`.

Figure 2-3. Proxy Setup

Follow these instructions to install the extension to VS Code:

1. Download the VSIX from OPEN-VSX.
   a. NOTE: Make sure to always download the latest version. For more information regarding the tool, refer to the C2000-IDEA GitHub repository.
2. Navigate to the *Extensions* tab in the left side bar panel.
3. Click on the three dots and select the *Install from VSIX*. Select the downloaded VSIX in the file explorer.

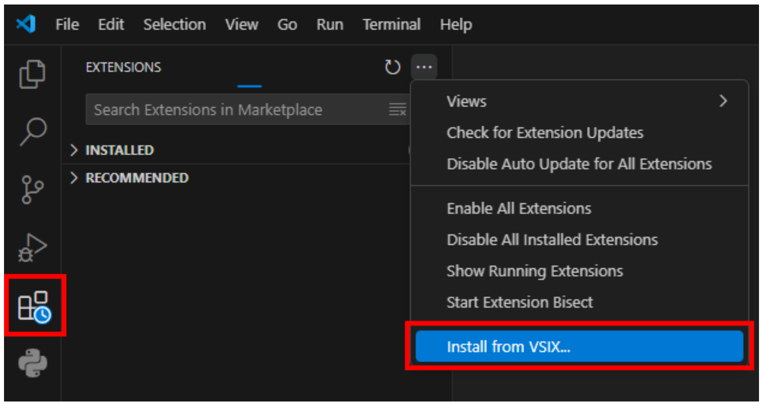


Figure 2-4. C2000 IDEA Setup in VS Code

Refresh the IDE. The extension has been successfully installed when the *C2000 IDE Assist* icon is present in the left side bar panel on the left side of the screen.

2.2 Import Project

Once the code editor platform is set up with the extension, the tool is ready to be used with CCS projects.

Follow these steps to import a project:

1. Open the CCS or VS Code IDE.
2. Open a new or existing workspace as normal.
3. Navigate to the CCS Extension by selecting the icon on the left side bar panel.

2.3 Detect Project

All projects using C2000 devices can be detected with a single click of the *Get Projects* element. This step allows the extension to recognize the device for each project and keep track of which project each file belongs to. Project Detection unlocks the complete capabilities of the extension in real-time such as Driverlib Migration across devices. Some extension features are only available with project detection (or default device setup, described in Section 2.5.1), whereas other features can be run on single files by requiring input of the current device. See the Section 4.4 for a full list of features and commands available.

Follow these steps to detect a project:

1. To detect all the projects in the workspace, use one of the two options below:
   a. Enter CTRL+SHIFT+P and type and click *C2000: Get Projects*.
   b. Click *Project Detection* > *Get Projects* in the ***C2000 IDEA - Features*** pane of the Extension tree.
2. Once run, all the projects in the workspace detected by the extension are present in the ***C2000 IDEA - Projects*** pane of the Extension tree. View your project and verify the device variant and current device details are as expected.

Note

C2000 IDEA is recommended to be used specifically within CCS to utilize all the features of the extension along with the complete debugging capabilities provided in CCS concurrently.

2.4 Basic Commands

CCS and VS Code provide a powerful palette of commands that allow users to perform most, if not all, tasks using solely the keyboard. These commands can be activated using CTRL+SHIFT+P. Refer to the table below for some easy-to-use commands enabled by the C2000 IDEA extension. Alternatively, select features have UI buttons available in the C2000 IDEA Features pane that automatically run the associated commands to activate the feature.

Note

See Section 4.4 in this document for the full list of commands or search "C2000" in the command pallette.

Table 2-1. C2000-IDEA Basic Commands

Purpose	Commands	Devices	Project Detection
Open Tool Walkthrough	C2000: Help Walkthrough	GEN2, GEN3, GEN4	NOT REQUIRED
Run Driverlib Register Vision	C2000: Run Driverlib Register Vision on Current File	GEN3, GEN4	NOT REQUIRED
Run Bitfield Register Vision	C2000: Run Bitfield RegisterVision on Current File	GEN3, GEN4	NOT REQUIRED
Enable Register Coder	C2000: Enable Register Code Write/Read	GEN3, GEN4	REQUIRED
Enable Interrupt Coder	C2000: Enable Interrupt Code Templates	GEN3, GEN4	REQUIRED
Enable Continuous Migration Check	C2000: Enable Continuous Migration Check on Current File	GEN3, GEN4	REQUIRED

2.5 Global Settings

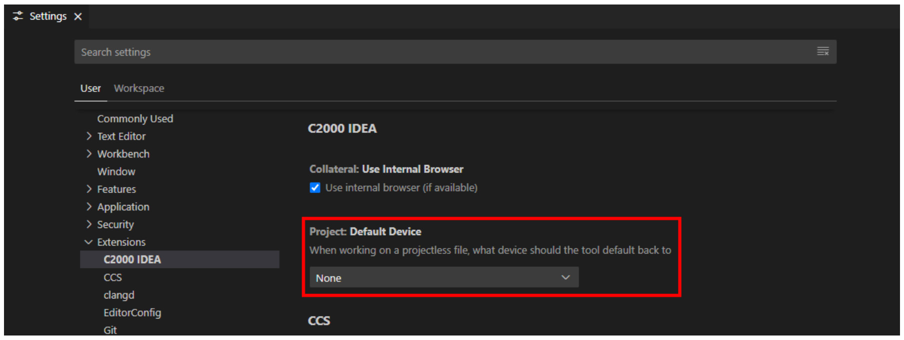
Global settings can be utilized depending on the user's project needs. In the Extension > C2000 IDEA tab, within the specific IDE Settings, there are several configuration options.

2.5.1 Project-less Detection

Users can manually select a default device to be used when a device is not automatically detected or when project detection has not been run. By setting a default device, most features of the tool can be utilized even if a file is not part of a detected project. In other words, when no device is automatically detected in a project-less file, the tool uses the user configured default device.

Follow these steps to set up the default C2000 device in CCS and VS Code:

1. Click the gear (Manage) icon in the left side bar panel and select the *Settings* option.
2. Navigate to User > Extensions > C2000 IDEA.
3. In *Project: Default Device*, select the desired device.



C2000 IDEA

Collateral: **Use Internal Browser**

✓ Use internal browser (if available)

Project: **Default Device**

When working on a projectless file, what device should the tool default back to

None

CCS

Figure 2-5. Default Device Selection

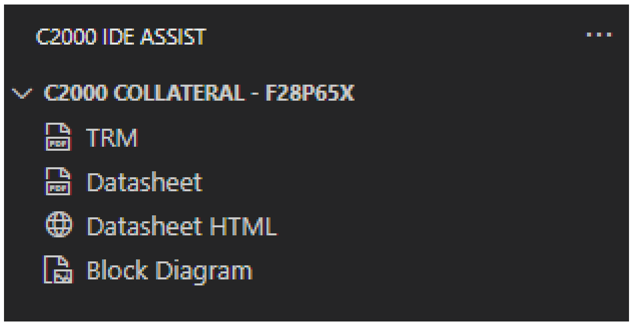
Note

TI recommends to always use the latest SDKs when developing with this extension as the C2000 IDEA correlates with the latest available SDKs.

3 Overview

This section discusses an overview of the C2000 IDEA interface and how to easily navigate through the tool. The features of the tool are split into four panels on the left hand side of the screen. The following images showcase an overview of the tool when working with a sample C2000WARE example.

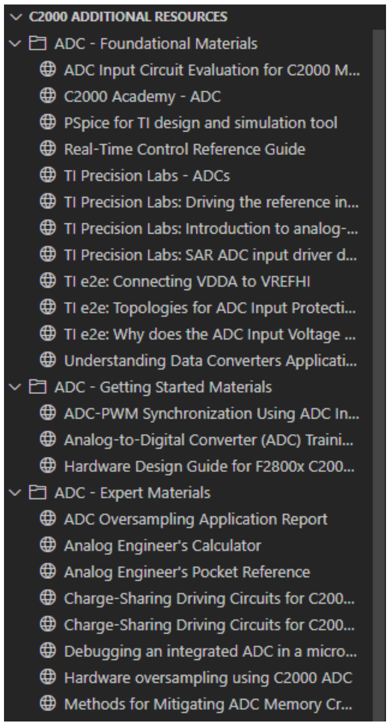
1. ***C2000 COLLATERALS:*** This panel showcases key collateral, including the Technical Reference Manual and Data Sheet, corresponding with the device associated with the project and/or file that is currently opened.



C2000 IDE ASSIST
C2000 COLLATERAL - F28P65X
TRM
Datasheet
Datasheet HTML
Block Diagram

Figure 3-1. C2000 Collaterals

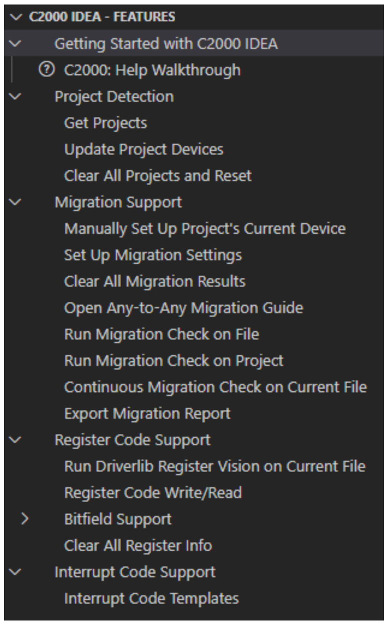
2. ***C2000 ADDITIONAL RESOURCES:*** This panel lists all the associated collateral with the project specific to the device and peripherals. The collateral is organized based on foundational, getting started, and expert material.



C2000 ADDITIONAL RESOURCES
ADC - Foundational Materials
ADC Input Circuit Evaluation for C2000 M...
C2000 Academy - ADC
PSpice for TI design and simulation tool
Real-Time Control Reference Guide
TI Precision Labs - ADCs
TI Precision Labs: Driving the reference in...
TI Precision Labs: Introduction to analog-...
TI Precision Labs: SAR ADC input driver d...
TI e2e: Connecting VDDA to VREFHI
TI e2e: Topologies for ADC Input Protecti...
TI e2e: Why does the ADC Input Voltage ...
Understanding Data Converters Applicati...
ADC - Getting Started Materials
ADC-PWM Synchronization Using ADC In...
Analog-to-Digital Converter (ADC) Traini...
Hardware Design Guide for F2800x C200...
ADC - Expert Materials
ADC Oversampling Application Report
Analog Engineer's Calculator
Analog Engineer's Pocket Reference
Charge-Sharing Driving Circuits for C200...
Charge-Sharing Driving Circuits for C200...
Debugging an integrated ADC in a micro...
Hardware oversampling using C2000 ADC
Methods for Mitigating ADC Memory Cr...

Figure 3-2. C2000 Additional Resources

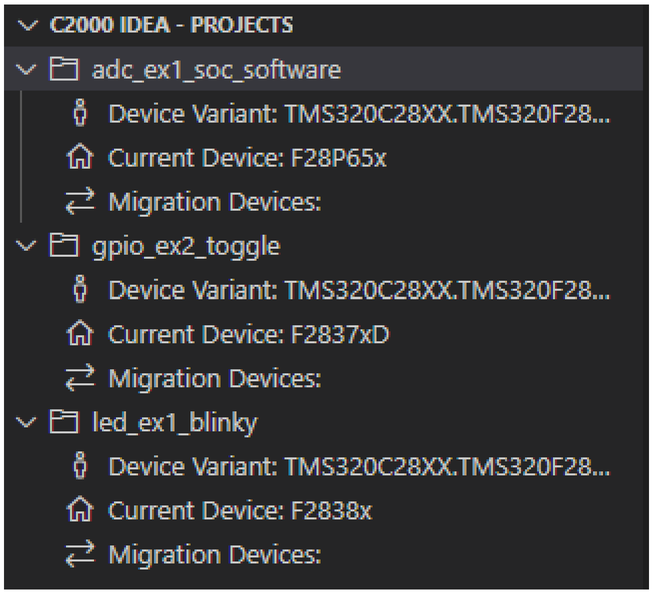
3. ***C2000 IDEA - FEATURES:*** This panel is primarily used when working with C2000 IDEA as all the software development tools including walkthroughs of the tool are hosted here.



C2000 IDEA - FEATURES
Getting Started with C2000 IDEA
C2000: Help Walkthrough
Project Detection
Get Projects
Update Project Devices
Clear All Projects and Reset
Migration Support
Manually Set Up Project's Current Device
Set Up Migration Settings
Clear All Migration Results
Open Any-to-Any Migration Guide
Run Migration Check on File
Run Migration Check on Project
Continuous Migration Check on Current File
Export Migration Report
Register Code Support
Run Driverlib Register Vision on Current File
Register Code Write/Read
Bitfield Support
Clear All Register Info
Interrupt Code Support
Interrupt Code Templates

Figure 3-3. C2000 IDEA - Features

4. ***C2000 IDEA - PROJECTS:*** This panel displays all the projects currently opened in a workspace. Each project has an associated *Device Variant*, the *Current Device*, and *Migration Devices* to indicate which device to migrate from the current device.



C2000 IDEA - PROJECTS
adc_ex1_soc_software
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F28P65x
Migration Devices:
gpio_ex2_toggle
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F2837xD
Migration Devices:
led_ex1_blinky
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F2838x
Migration Devices:

Figure 3-4. C2000 IDEA - Projects

4 Features

This chapter details the full capabilities of features discussed in the previous section and how to optimize them while developing software.

4.1 Targeted Collateral Delivery

The C2000 IDEA tool offers a highly efficient method of accessing a wealth of collateral at your fingertips. With targeted collateral delivery, users can reach the most relevant information with a single click, eliminating the need to search through various documents. The tool automatically detects the topic at hand and provides immediate suggestions, including direct links to critical resources like device documentation and register definitions.

4.1.1 Key Collateral

When a detected project is edited, links to key collateral for the project's device display automatically in the C2000 Collateral pane of the Extension. This feature generates top level links to the device data sheet and the Technical Reference Manual (TRM), as well as links to key sections inside this documentation, like the device Functional Block Diagram.

4.1.2 Additional Resources

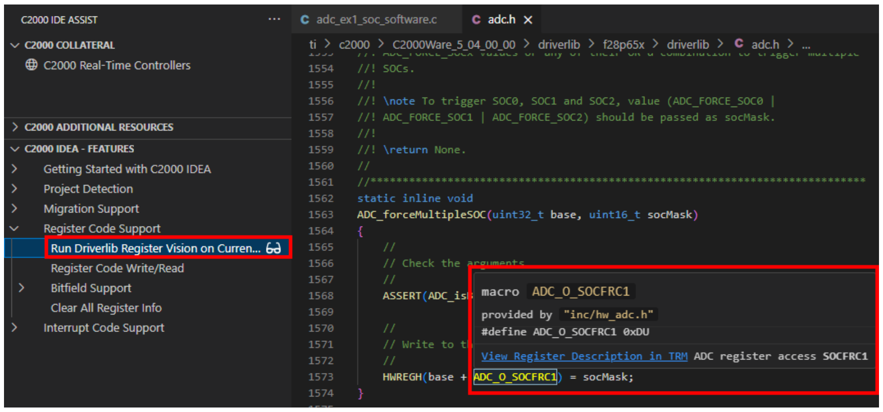
As the user cursor moves in an active file, the collateral recommendations change automatically to highlight additional helpful collateral beyond key documentation in the *C2000 Collateral* pane. This includes Application Reports, C2000 Academy pages, labs, E2E™ forum answers, and videos related to the word or phrase (ex: C2000 peripheral names) the user is currently editing. Links to these resources can be accessed from the *C2000 Additional Resources* pane in the extension tree.

4.1.3 Register Vision

The Register Vision feature provides easy access to TRM register description for all the detected registers in a file. Once the command is run, all detected registers for the specific C2000 device are highlighted and direct links to the register description in the device HTML TRM are provided.

Follow these steps to enable the Driverlib Register Vision:

1. Open a C2000 application C-Code file.
2. Run the Driverlib Register Vision by either:
    a. Press CTRL+SHIFT+P, type and select *C2000: Run Driverlib Register Vision on Current File*.
    b. Click on *Register Code Support* > *Run Driverlib Register Vision on Current File* in the ***C2000 IDEA - Features*** pane of the Extension tree.
3. If no project has been detected (or if the file doesn't belong to a project), the extension prompts to input the C2000 device corresponding to the file.
4. Hover over an identified register that is highlighted in yellow.
5. Click on the *View Register Description in TRM* link to access the register description in the online HTML TRM.



```
//! SOCs.
//!
//! \note To trigger SOC0, SOC1 and SOC2, value (ADC_FORCE_SOC0 |
//! ADC_FORCE_SOC1 | ADC_FORCE_SOC2) should be passed as socMask.
//!
//! \return None.
//
//*****************************************************************************
static inline void
ADC_forceMultipleSOC(uint32_t base, uint16_t socMask)
{
    //
    // Check the arguments
    //
    ASSERT(ADC_is

    //
    // Write to t
    //
    HWREGH(base + ADC_O_SOCFRC1) = socMask;
}
```

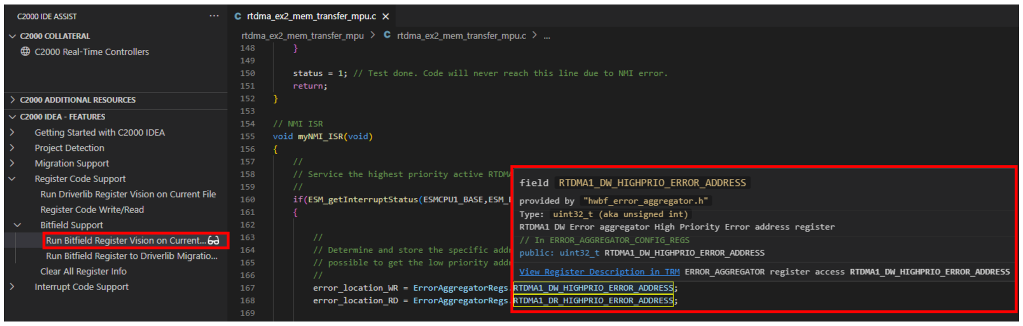
```
macro ADC_O_SOCFRC1
provided by "inc/hw_adc.h"
#define ADC_O_SOCFRC1 0xDU
View Register Description in TRM ADC register access SOCFRC1
```

Figure 4-1. Driverlib Register Vision with F28P65x

C2000 IDEA also supports register accesses for bitfield-style software development.

Follow these steps to enable the Bitfield Register Vision:

1. Open a C2000 application C-Code file.
2. Run the Bitfield Register Vision by either:
   a. Press CTRL+SHIFT+P, type and select *C2000: Run Bitfield Register Vision on Current File*.
   b. Click on *Register Code Support* > *Bitfield Support* > *Run Bitfield Register Vision on Current File* in the ***C2000 IDEA - Features*** pane of the Extension tree.
3. If no project has been detected (or if the file doesn't belong to a project), the extension prompts to input the C2000 device corresponding to the file.
4. Hover over an identified register that is highlighted in yellow.
5. Click on the *View Register Description in TRM* link to access the register description in the online HTML TRM.


C2000 IDE ASSIST
C2000 COLLATERAL
C2000 Real-Time Controllers
C2000 ADDITIONAL RESOURCES
C2000 IDEA - FEATURES
Getting Started with C2000 IDEA
Project Detection
Migration Support
Register Code Support
Run Driverlib Register Vision on Current File
Register Code Write/Read
Bitfield Support
Run Bitfield Register Vision on Current...
Run Bitfield Register to Driverlib Migratio...
Clear All Register Info
Interrupt Code Support
rtdma_ex2_mem_transfer_mpu.c
rtdma_ex2_mem_transfer_mpu > rtdma_ex2_mem_transfer_mpu.c > ...

```
    }

    status = 1; // Test done. Code will never reach this line due to NMI error.
    return;
}

// NMI ISR
void myNMI_ISR(void)
{
    //
    // Service the highest priority active RTDMA
    //
    if(ESM_getInterruptStatus(ESMCPU1_BASE,ESM_I
    {

        //
        // Determine and store the specific addr
        // possible to get the low priority addr
        //
        error_location_WR = ErrorAggregatorRegs.RTDMA1_DW_HIGHPRIO_ERROR_ADDRESS;
        error_location_RD = ErrorAggregatorRegs.RTDMA1_DR_HIGHPRIO_ERROR_ADDRESS;
```

```
field RTDMA1_DW_HIGHPRIO_ERROR_ADDRESS
provided by "hwbf_error_aggregator.h"
Type: uint32_t (aka unsigned int)
RTDMA1 DW Error aggregator High Priority Error address register
// In ERROR_AGGREGATOR_CONFIG_REGS
public: uint32_t RTDMA1_DW_HIGHPRIO_ERROR_ADDRESS
View Register Description in TRM ERROR_AGGREGATOR register access RTDMA1_DW_HIGHPRIO_ERROR_ADDRESS
```

Figure 4-2. Bitfield Register Vision with F29H85x

4.2 Developer Efficiency Tools

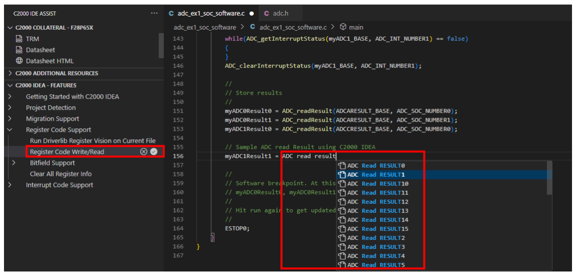
C2000 IDEA enhances developer efficiency by providing software template support, allowing users to easily insert code such as interrupt definitions and register accesses. This feature not only accelerates development but also guides users on how to write efficient C2000-based software.

4.2.1 Register Coder

The Register Code feature automatically generates code to write or read from a specific register or register field. The auto-generated code provides a template for writing to or reading from a register.

Follow these steps to enable the Register Coder:

1. Run the Register Coder by either:
   a. Press CTRL+SHIFT+P, type and select *C2000: Enable Register Code Write/Read*.
   b. Click the check toggle in *Register Code Support* > *Register Code Write/Read* in the ***C2000 IDEA - Features*** pane of the Extension tree.
2. Once enabled, open a C2000 application C-Code file from a detected project.
3. Begin typing the peripheral, register, or field name. The Register Coder feature displays an alphabetical list of all available registers and register fields to read or write for the current device.
4. Select a choice to autogenerate the template code. Press *TAB* on the keyboard to edit the base address field, variable to read into, and/or value to write, as needed.



```
        while(ADC_getInterruptStatus(myADC1_BASE, ADC_INT_NUMBER1) == false)
        {
        }
        ADC_clearInterruptStatus(myADC1_BASE, ADC_INT_NUMBER1);

        //
        // Store results
        //
        myADC0Result0 = ADC_readResult(ADCARESULT_BASE, ADC_SOC_NUMBER0);
        myADC0Result1 = ADC_readResult(ADCARESULT_BASE, ADC_SOC_NUMBER1);
        myADC1Result0 = ADC_readResult(ADCCRESULT_BASE, ADC_SOC_NUMBER0);

        // Sample ADC read Result using C2000 IDEA
        myADC1Result1 = ADC read result

        //
        // Software breakpoint. At this
        // myADC0Result0, myADC0Result1
        //
        // Hit run again to get updated
        //
        ESTOP0;
    }
}
```

```
ADC Read RESULT0
ADC Read RESULT1
ADC Read RESULT10
ADC Read RESULT11
ADC Read RESULT12
ADC Read RESULT13
ADC Read RESULT14
ADC Read RESULT15
ADC Read RESULT2
ADC Read RESULT3
ADC Read RESULT4
ADC Read RESULT5
```

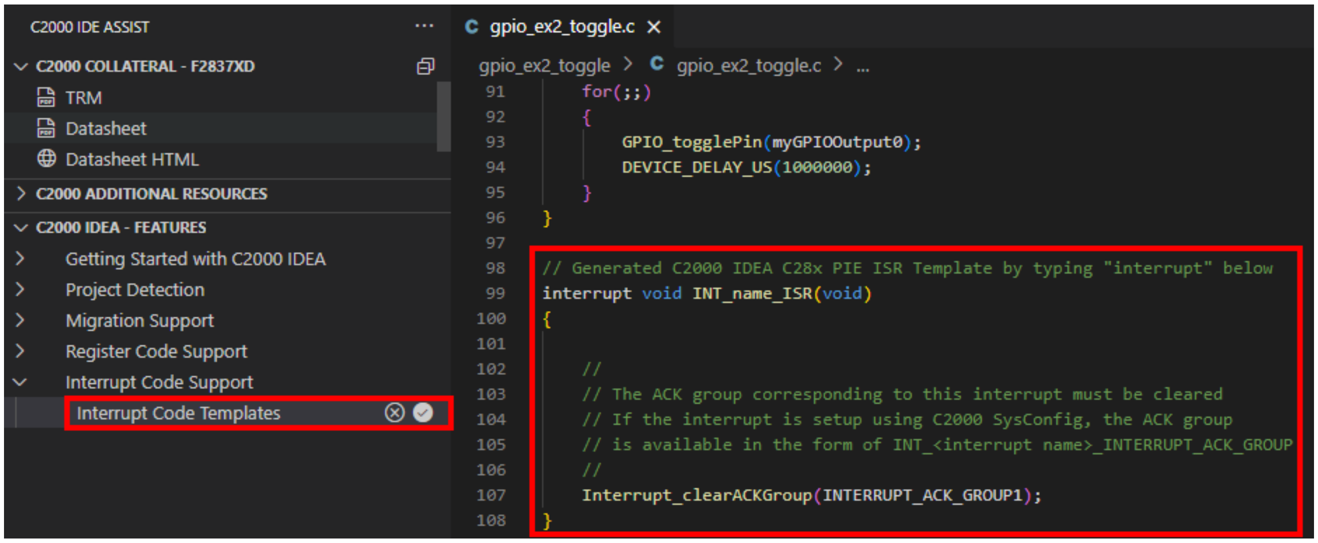
Figure 4-3. Register Coder Read with F28P65x

4.2.2 Interrupt Coder

The Interrupt Code Template feature automatically generates a template Interrupt Service Routine (ISR) function for any interrupt available on the device.

Follow these steps to enable the Interrupt Coder:

1. Run the Interrupt Coder by either:
   a. Press CTRL+SHIFT+P, type and select *C2000: Enable Register Code Write/Read*.
   b. Click on *Interrupt Code Support* > *Interrupt Code Templates* in the ***C2000 IDEA - Features*** pane of the Extension tree.
2. Once enabled, open a C2000 application C-Code file to add an Interrupt Service Routine (ISR).
3. Begin typing the phrase *interrupt handler [interrupt name]* in the code editor. The Interrupt Coder feature displays an alphabetical list of all available interrupts for the current device.
4. Select a choice to autogenerate the template interrupt code. Fill in the rest of the code needed for the specific application.

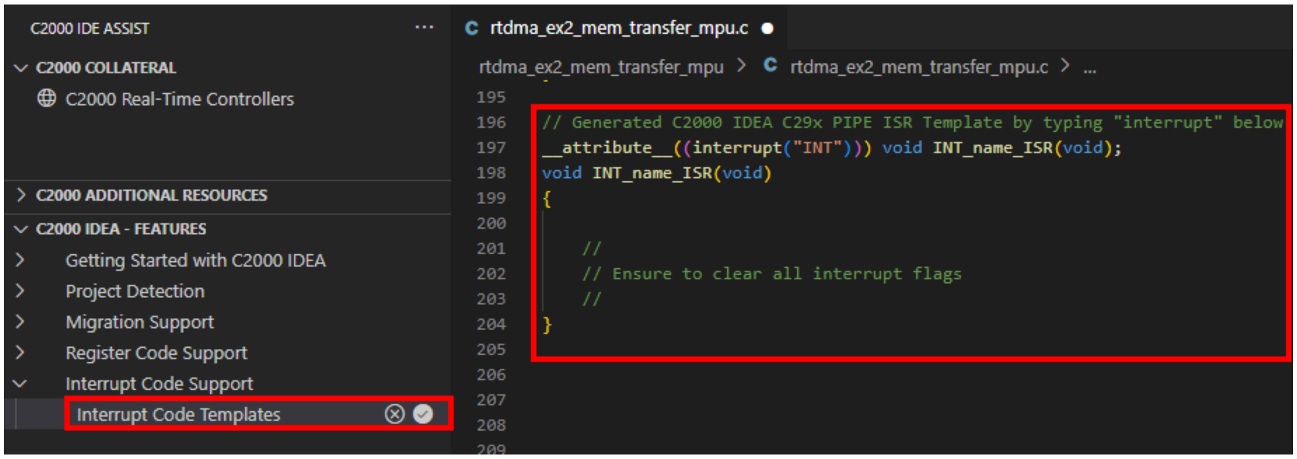


```
    for(;;)
    {
        GPIO_togglePin(myGPIOOutput0);
        DEVICE_DELAY_US(1000000);
    }
}

// Generated C2000 IDEA C28x PIE ISR Template by typing "interrupt" below
interrupt void INT_name_ISR(void)
{

    //
    // The ACK group corresponding to this interrupt must be cleared
    // If the interrupt is setup using C2000 SysConfig, the ACK group
    // is available in the form of INT_<interrupt name>_INTERRUPT_ACK_GROUP
    //
    Interrupt_clearACKGroup(INTERRUPT_ACK_GROUP1);
}
```

Figure 4-4. Interrupt Coder with C28x

Note

For C28x devices, the correct ACK group for the given peripheral interrupt is automatically filled in. Make sure to add any application-specific code before this line.


C2000 IDE ASSIST
C2000 COLLATERAL
C2000 Real-Time Controllers
C2000 ADDITIONAL RESOURCES
C2000 IDEA - FEATURES
Getting Started with C2000 IDEA
Project Detection
Migration Support
Register Code Support
Interrupt Code Support
Interrupt Code Templates
rtdma_ex2_mem_transfer_mpu.c
rtdma_ex2_mem_transfer_mpu > rtdma_ex2_mem_transfer_mpu.c > ...

```

// Generated C2000 IDEA C29x PIPE ISR Template by typing "interrupt" below
__attribute__((interrupt("INT"))) void INT_name_ISR(void);
void INT_name_ISR(void)
{

    //
    // Ensure to clear all interrupt flags
    //
}
```

Figure 4-5. Interrupt Coder with C29x

Note

For C29x devices, the interrupt type attribute is automatically included to indicate the type of interrupt being defined. By default, the extension defines interrupts as *INT* but the attribute can be modified to *RTINT*.

4.3 Migration Support

C2000 IDEA simplifies code migration between C2000 devices by automatically detecting changes — such as added, removed, or modified registers and register fields — in driverlib code, making the migration process seamless. This support extends to migration for both detected projects and standalone files for driverlib migration. For select migration paths, bitfield code migration is also available. Additionally, the tool provides migration assistance between both F28x-to-F28x and F28x-to-F29x devices, highlighting differences between the two and offering suggested solutions.

Note

For the complete list of specific devices supported by each command, refer to the Section 4.4, and for the differences between driverlib-style bitfield-style and code, refer to the Sofware Drivers documentation.

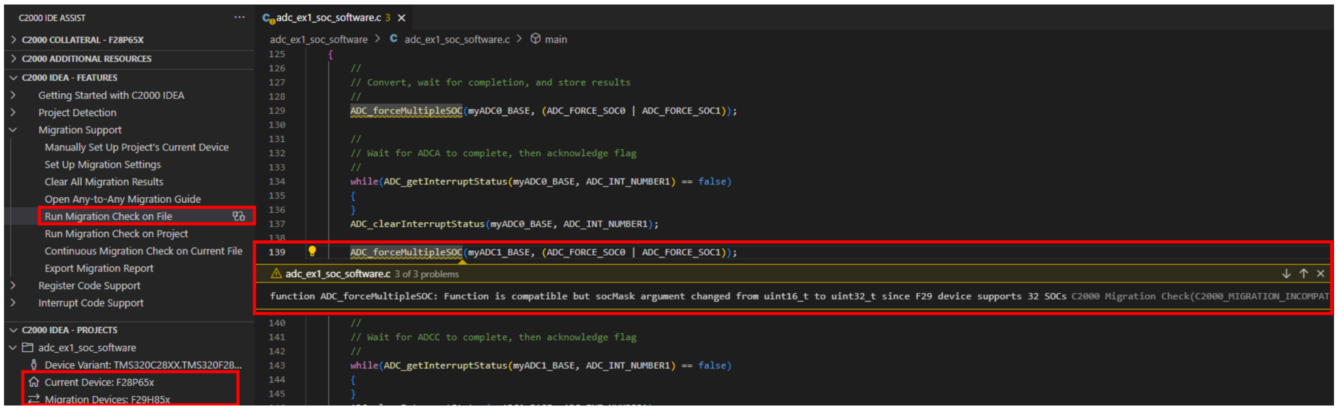
4.3.1 Driverlib Migration

The C2000 IDEA Extension can be used to run a F28x-to-F28x or F28x-to-F29x migration check on a project or file written with driverlib-style code. This code style is characterized by using calls to functions defined in the driverlib source files (ex: `Device_init()`) and/or register accesses containing the _O_ syntax.

4.3.1.1 File-Based Migration Check

Follow these steps to enable Driverlib Migration on a file:

1. Open a C2000 application C-Code file.
2. Run the migration check by either:
    a. Press CTRL+SHIFT+P, type and select *C2000: Run Migration Check on File*.
    b. Click on *Migration Support* > *Run Migration Check on Project* in the ***C2000 IDEA - Features*** pane of the Extension tree
3. Select the current C2000 device the code in the file applies to.
4. Select the C2000 device to migrate the file to.
5. The status bar at the bottom right of the screen displays *Migration check completed* when finished. All major migration concerns in the file are underlined with a red squiggly line. All other migration warnings in the file are underlined with a yellow squiggly line.
6. Review and resolve concerns throughout the file. The following options are provided when the underlined code is hovered over:
    a. Select *View Problem* to quickly loop through the detected concerns in the file.


C2000 IDE ASSIST
C2000 COLLATERAL - F28P65X
C2000 ADDITIONAL RESOURCES
C2000 IDEA - FEATURES
Getting Started with C2000 IDEA
Project Detection
Migration Support
Manually Set Up Project's Current Device
Set Up Migration Settings
Clear All Migration Results
Open Any-to-Any Migration Guide
Run Migration Check on File
Run Migration Check on Project
Continuous Migration Check on Current File
Export Migration Report
Register Code Support
Interrupt Code Support
C2000 IDEA - PROJECTS
adc_ex1_soc_software
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F28P65x
Migration Devices: F29H85x
adc_ex1_soc_software.c 3
adc_ex1_soc_software > adc_ex1_soc_software.c > main
adc_ex1_soc_software.c 3 of 3 problems

```
    {
        //
        // Convert, wait for completion, and store results
        //
        ADC_forceMultipleSOC(myADC0_BASE, (ADC_FORCE_SOC0 | ADC_FORCE_SOC1));

        //
        // Wait for ADCA to complete, then acknowledge flag
        //
        while(ADC_getInterruptStatus(myADC0_BASE, ADC_INT_NUMBER1) == false)
        {
        }
        ADC_clearInterruptStatus(myADC0_BASE, ADC_INT_NUMBER1);

        ADC_forceMultipleSOC(myADC1_BASE, (ADC_FORCE_SOC0 | ADC_FORCE_SOC1));
```

```
function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 device supports 32 SOCs C2000 Migration Check(C2000_MIGRATION_INCOMPAT
```

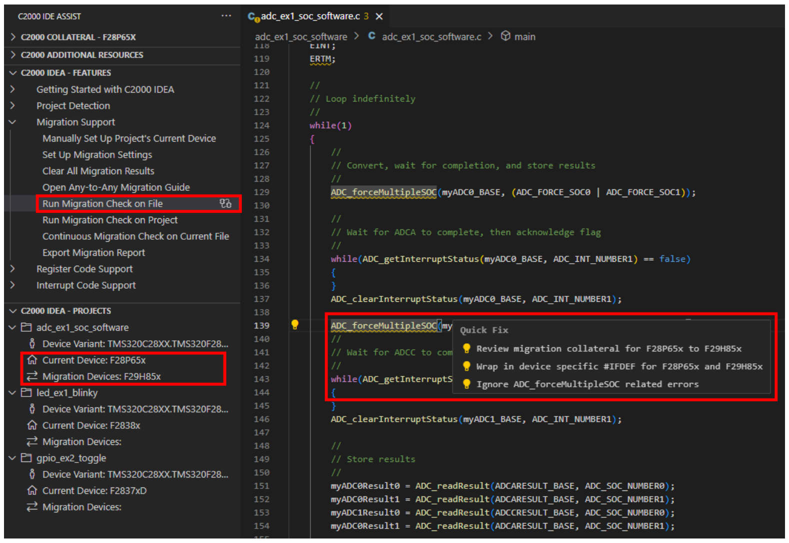
```
        //
        // Wait for ADCC to complete, then acknowledge flag
        //
        while(ADC_getInterruptStatus(myADC1_BASE, ADC_INT_NUMBER1) == false)
        {
        }
```

Figure 4-6. Migration Check - View Problem

    b. Select *Quick Fix* to mitigate the migration concern. Select one of the below options:
        i. *Review migration collateral for [current device] to [migration device]*- This option opens a link to the online migration collateral for the specific migration path using the latest version of C2000WARE.

ii. *Wrap in device specific #IFDEF for [current device] and [migration device]* - This option autogenerates pre-processor wrappers around the line of code so that an updated version of the code can be compiled for the new device. Fill in the line with the *//Enter alternate code* comment with the modified code and add a #define for the current device somewhere in the file.

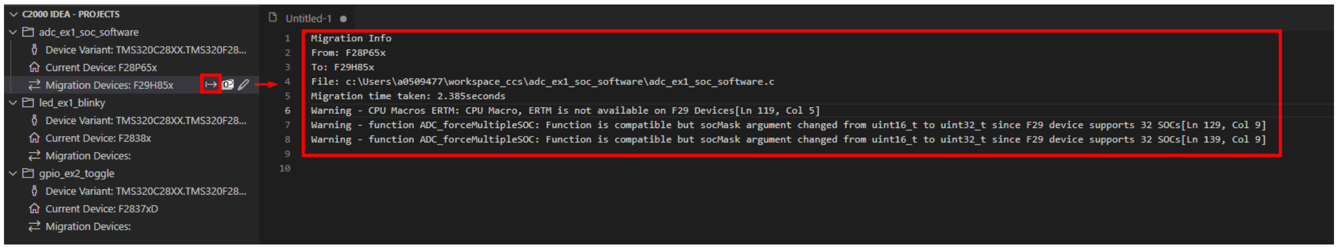
iii. *Ignore code related errors* - This option ignores this migration concern.


C2000 IDE ASSIST
C2000 COLLATERAL - F28P65X
C2000 ADDITIONAL RESOURCES
C2000 IDEA - FEATURES
Getting Started with C2000 IDEA
Project Detection
Migration Support
Manually Set Up Project's Current Device
Set Up Migration Settings
Clear All Migration Results
Open Any-to-Any Migration Guide
Run Migration Check on File
Run Migration Check on Project
Continuous Migration Check on Current File
Export Migration Report
Register Code Support
Interrupt Code Support
C2000 IDEA - PROJECTS
adc_ex1_soc_software
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F28P65x
Migration Devices: F29H85x
led_ex1_blinky
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F2838x
Migration Devices:
gpio_ex2_toggle
Device Variant: TMS320C28XX.TMS320F28...
Current Device: F2837xD
Migration Devices:
adc_ex1_soc_software.c 3
adc_ex1_soc_software > adc_ex1_soc_software.c > main
Quick Fix
Review migration collateral for F28P65x to F29H85x
Wrap in device specific #IFDEF for F28P65x and F29H85x
Ignore ADC_forceMultipleSOC related errors

```
    ERTM;

    //
    // Loop indefinitely
    //
    while(1)
    {
        //
        // Convert, wait for completion, and store results
        //
        ADC_forceMultipleSOC(myADC0_BASE, (ADC_FORCE_SOC0 | ADC_FORCE_SOC1));

        //
        // Wait for ADCA to complete, then acknowledge flag
        //
        while(ADC_getInterruptStatus(myADC0_BASE, ADC_INT_NUMBER1) == false)
        {
        }
        ADC_clearInterruptStatus(myADC0_BASE, ADC_INT_NUMBER1);

        ADC_forceMultipleSOC(my
        //
        // Wait for ADCC to com
        //
        while(ADC_getInterruptS
        {
        }
        ADC_clearInterruptStatus(myADC1_BASE, ADC_INT_NUMBER1);

        //
        // Store results
        //
        myADC0Result0 = ADC_readResult(ADCARESULT_BASE, ADC_SOC_NUMBER0);
        myADC0Result1 = ADC_readResult(ADCARESULT_BASE, ADC_SOC_NUMBER1);
        myADC1Result0 = ADC_readResult(ADCCRESULT_BASE, ADC_SOC_NUMBER0);
        myADC0Result1 = ADC_readResult(ADCARESULT_BASE, ADC_SOC_NUMBER1);
```

Figure 4-7. Migration Check - Quick Fix

7. The migration report contains a list of the detected migration concerns. To export the migration report to the desired file path, either:
    a. Enter CTRL+SHIFT+P, type and select *C2000: Export Migration Report*.
    b. Click on *Migration Support* > *Export Migration Report* in the ***C2000 IDEA - Features*** pane of the Extension tree.



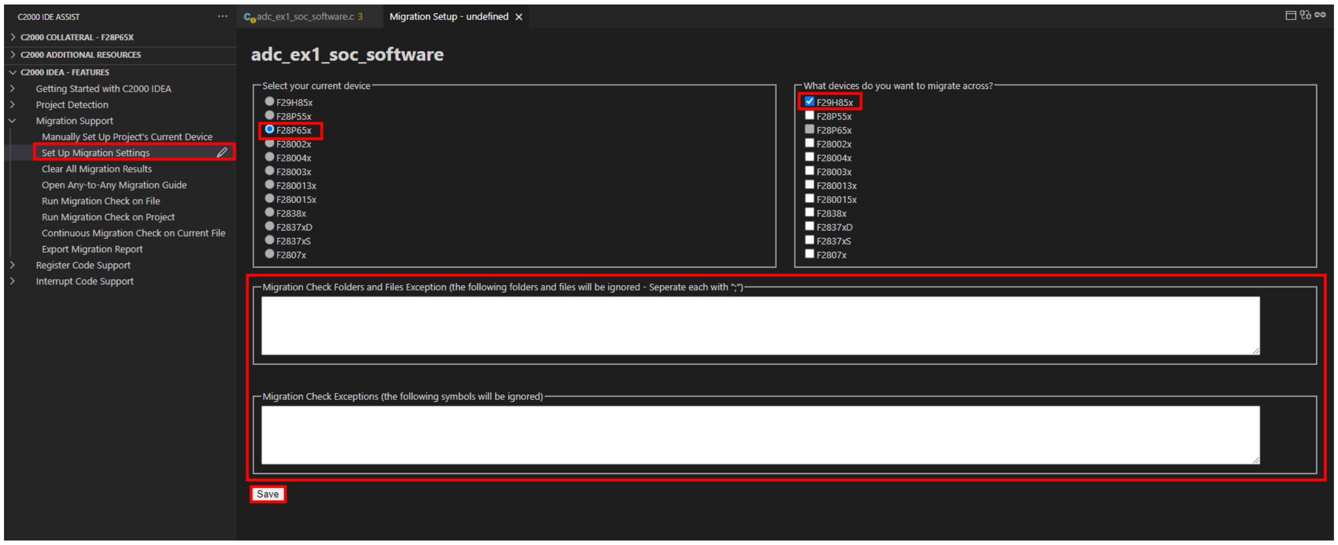
```
Migration Info
From: F28P65x
To: F29H85x
File: c:\Users\a0509477\workspace_ccs\adc_ex1_soc_software\adc_ex1_soc_software.c
Migration time taken: 2.385seconds
Warning - CPU Macros ERTM: CPU Macro, ERTM is not available on F29 Devices[Ln 119, Col 5]
Warning - function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 device supports 32 SOCs[Ln 129, Col 9]
Warning - function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 device supports 32 SOCs[Ln 139, Col 9]
```

Figure 4-8. Migration Report

4.3.1.2 Project-Based Migration Check

Follow these steps to enable Driverlib Migration on a project:

1. Run project detection (see instructions in Section 2.3).
2. Set up the C2000 device to migrate from/to as well as customize which folders/files are to be ignored for the migration check by either:
    a. Click the following icon next to *Migration Devices* under the detected project in the ***C2000 IDEA - Projects*** pane of the Extension tree.
    b. Press CTRL+SHIFT+P, type and select *C2000: Set Up Migration Settings*.
    c. Click on *Migration Support* > *Set Up Migration Settings* in the ***C2000 IDEA - Features*** pane of the Extension tree.


C2000 IDE ASSIST
C2000 COLLATERAL - F28P65X
C2000 ADDITIONAL RESOURCES
C2000 IDEA - FEATURES
Getting Started with C2000 IDEA
Project Detection
Migration Support
Manually Set Up Project's Current Device
Set Up Migration Settings
Clear All Migration Results
Open Any-to-Any Migration Guide
Run Migration Check on File
Run Migration Check on Project
Continuous Migration Check on Current File
Export Migration Report
Register Code Support
Interrupt Code Support
adc_ex1_soc_software.c 3
Migration Setup - undefined
Save

adc_ex1_soc_software

Select your current device: ○ F29H85x ○ F28P55x [● F28P65x] ○ F28002x ○ F28004x ○ F28003x ○ F280013x ○ F280015x ○ F2838x ○ F2837xD ○ F2837xS ○ F2807x

What devices do you want to migrate across?: [✓ F29H85x] ☐ F28P55x ☐ F28P65x ☐ F28002x ☐ F28004x ☐ F28003x ☐ F280013x ☐ F280015x ☐ F2838x ☐ F2837xD ☐ F2837xS ☐ F2807x

Migration Check Folders and Files Exception (the following folders and files will be ignored - Seperate each with ";")

Migration Check Exceptions (the following symbols will be ignored)

Figure 4-9. Migration Settings Setup

3. Run the migration check by either:
   a. Click the following icon next to *Migration Devices* under the detected project in the ***C2000 IDEA - Projects*** pane of the Extension tree.
   b. Press CTRL+SHIFT+P, type and select *C2000: Run Migration Check on Project*.
   c. Click on *Migration Support* > *Run Migration Check on Project* in the ***C2000 IDEA - Features*** pane of the Extension tree.

Note

An error is thrown by the extension if there is any overlap of folders or files in the migration folder and files to be ignored information in the migration setup page.

Note

Do not use these migration features while the tool is already running a migration check on the project. Wait for the *Migration check completed on [project name]* at the bottom right of the screen before enabling other migration features. The amount of time taken to run the check entirely depends on how many files, lines and code changes exist in a project. The migration report includes the time taken on each file.

4. Select the project in the workspace to run a migration check on.
5. The status bar at the bottom right of the screen displays *Migration check completed on [project name]* when finished. All major migration concerns in the file are underlined with a red squiggly line. All other migration warnings in the file are underlined with a yellow squiggly line.
6. Review and resolve concerns throughout the file. The following options are provided when the underlined code is hovered over:
   a. Select *View Problem* to quickly loop through the detected concerns in the file .
   b. Select *Quick Fix* to mitigate the migration concern. Select one of the below options:
      i. *Review migration collateral for [current device] to [migration device]*- This option opens a link to the online migration collateral for the specific migration path using the latest version of C2000WARE.
      ii. *Wrap in device specific #IFDEF for [current device] and [migration device]* - This option autogenerates pre-processor wrappers around the line of code so that an updated version of the code can be compiled for the new device. Fill in the line with the *//Enter alternate code* comment with the modified code and add a #define for the current device somewhere in the file.
      iii. *Ignore code related errors* - This option ignores this migration concern.
7. The migration report contains a list of the detected migration concerns, the time taken to complete the migration, and any settings the user customized to ignore. To export the migration report to the desired file path, either:
   a. Click the following icon next to *Migration Devices* under the detected project in the ***C2000 IDEA - Projects*** pane of the Extension tree.

b. Enter CTRL+SHIFT+P, type and select *C2000: Export Migration Report*.
c. Click on *Migration Support* > *Export Migration Report* in the ***C2000 IDEA - Features*** pane of the Extension tree.

4.3.2 Bitfield Migration

The C2000 IDEA Extension can be used to run a F28x-to-F28x or F28x-to-F29x migration check on a file written with bitfield-style code. This code style is characterized by using calls to functions defined in the bitfield source files (ex: `InitSysCtrl()`) and/or register accesses containing the `[base name].[register name].all` or `[base name].[register name].bit.[field name]` syntax.

Follow these steps to enable Bitfield Migration:

1. Open a C2000 application C-Code file.
2. Run the migration check by pressing CTRL+SHIFT+P, typing and selecting *C2000: Run Bitfield Migration Check on File*
3. Select the current C2000 device the code in the file applies to.
4. Select the C2000 device to migrate the file to.
5. The status bar at the bottom right of the screen displays *Finished Bitfield Migration from [current device] to [migration device]* when finished. All migration concerns in the file are underlined with a red squiggly line.
6. Review and resolve concerns throughout the file. The following options are provided when the underlined code is hovered over:
   a. Select *View Problem* to quickly loop through the detected concerns in the file.
   b. Select *Quick Fix* to mitigate the migration concern. Select one of the below options:
      i. *Review migration collateral for [current device] to [migration device]*- This option opens a link to the online migration collateral for the specific migration path using the latest version of C2000WARE.
      ii. *Wrap in device specific #IFDEF for [current device] and [migration device]* - This option autogenerates pre-processor wrappers around the line of code so that an updated version of the code can be compiled for the new device. Fill in the line with the *//Enter alternate code* comment with the modified code and add a #define for the current device somewhere in the file.
      iii. *Ignore code related errors* - This option ignores this migration concern.

4.4 Command List

Refer to the following table for the complete list of commands that can be utilized within the C2000 IDE Assist Tool.

Table 4-1. Table Needs a Title

Command Name	Supported Devices	Supported Software	Project Detection OR Default Device Setup
C2000: Help Walkthrough	N/A	N/A	
C2000: Help Interrupt Coder Walkthrough	N/A	N/A	
C2000: Help Register Coder Walkthrough	N/A	N/A	
C2000: Help Register Vision Walkthrough	N/A	N/A	
C2000: Open C2000-IDEA Extension View	N/A	N/A	
C2000: Manually Set Up Project's Current Device	GEN3, GEN4	N/A	REQUIRED
C2000: Set Up Migration Settings	GEN3, GEN4	N/A	REQUIRED
C2000: Get Projects	GEN3, GEN4	Bitfield + Driverlib	
C2000: Clear All Projects and Reset	GEN3, GEN4	Bitfield + Driverlib	
C2000: Update Project Devices	GEN3, GEN4	Bitfield + Driverlib	
C2000: Open Any-to-Any Migration Guide	From: GEN3, GEN4 To: GEN3, GEN4	N/A	

Table 4-1. Table Needs a Title (continued)

Command Name	Supported Devices	Supported Software	Project Detection OR Default Device Setup
C2000: Export Migration Report	From: GEN3, GEN4 To: GEN3, GEN4	Driverlib	
C2000: Run Migration Check on File	From: GEN3, GEN4 To: GEN3, GEN4	Driverlib	
C2000: Enable Continuous Migration Check on Current File	From: GEN3, GEN4 To: GEN3, GEN4	Driverlib	
C2000: Disable Continuous Migration Check on Current File	From: GEN3, GEN4 To: GEN3, GEN4	Driverlib	
C2000: Clear All Migration Results	From: GEN3, GEN4 To: GEN3, GEN4 & From: F2803x To: F280013x	Bitfield + Driverlib	
C2000: Run Migration Check on Project	From: GEN3, GEN4 To: GEN3, GEN4	Driverlib	
C2000: Run Bitfield Migration Check on File	From: GEN3, GEN4 To: GEN3, GEN4 & From: F2803x To: F280013x	Bitfield	
C2000: Run Driverlib Register Vision on Current File	GEN3, GEN4	Driverlib	RECOMMENDED
C2000: Run Bitfield Register Vision on Current File	GEN3, GEN4, F2803x	Bitfield	RECOMMENDED
C2000: Run Bitfield Register to Driverlib Migration on Current File (BETA)	GEN3, GEN4, F2803x	Bitfield	RECOMMENDED
C2000: Clear All Register Info	GEN3, GEN4, F2803x	Driverlib + Bitfield	
C2000: Enable Register Code Write/Read	GEN3, GEN4	Driverlib	REQUIRED
C2000: Disable Register Code Write/Read	GEN3, GEN4	Driverlib	REQUIRED
C2000: Enable Interrupt Code Templates	GEN3, GEN4	Driverlib	REQUIRED
C2000: Disable Interrupt Code Templates	GEN3, GEN4	Driverlib	REQUIRED
C2000: Project Tree View Refresh	GEN3, GEN4	Bitfield + Driverlib	

5 Summary

The Texas Instruments C2000 SDKs integrate essential tools, including drivers and libraries, to help developers of all levels build real-time control applications. This includes the C2000 SysConfig which simplifies code initialization and peripheral configuration. The C2000 IDEA tool builds on these foundational capabilities to centralize the development environment for easier coding and debugging. The IDE Assist facilitates migration across device portfolios and from legacy code to new, helping users navigate unique C2000 features and speeding up software implementation with live assistance at every stage of development.

6 References

Tools:

- Texas Instruments, *C2000 IDEA Open VSX (VSIX Download)*
- Texas Instruments, *C2000 IDEA GitHub Repository (VSIX Download)*
- Texas Instruments, *Code Composer 20 IDE*
- Texas Instruments, *C2000WARE (F28x SDK)*
- Texas Instruments, *F29H85X-SDK (F29x SDK)*

Documentation:

- Texas Instruments, *C28x Academy - Migration Resources*
- Texas Instruments, *C29x Academy - Migration Resources*
- Texas Instruments, *F28x to F29x Software Migration Guide*
- Texas Instruments, *Application Software Migration to the C29 CPU* application note.
- Texas Instruments, *TMS320F2837x, TMS320F2838x, TMS320F28P65x Migration to TMS320F29H85x* User's Guide
- Texas Instruments, *C2000 Design & Development*