

## Application Note

## C2000 IDE Assist 移行ツールガイド



Shashank Madineni, Nima Eskandari, Delaney Woodward, and Aishwarya Rajesh

## 概要

C2000 IDE Assist または C2000 IDEA は、テキサス インストルメンツの C2000™マイコンの開発を強化する目的で設計された統合開発ツールです。これは、Visual Studio Code™と Code Composer Studio™内で一元的な環境を実現し、プロジェクト検出、対象を絞った資料配信、デベロッパー効率化ツールなどの機能を提供します。主な機能の一つは移行サポートであり、開発者が異なる C2000 デバイス間でアプリケーション コードを効率的に移行できるよう支援します。

このアプリケーション ノートは、プロジェクトの検出と設定から、移行チェックの実行、互換性の問題の解決に至るまで、移行プロセスの各ステップをユーザーが順を追って進められるように案内します。このドキュメントでは、一般的な移行に関する懸念に対するクイック修正、コードの違いを分析するための移行レポート、ならびにデバイス アーキテクチャ間のスムーズで信頼性の高い移行を実現するための重要な制約事項や考慮点について解説しています。

## 目次

1 はじめに.....	3
1.1 はじめに.....	3
2 概要.....	4
3 移行サポート機能.....	6
3.1 プロジェクト検出.....	7
3.2 移行設定ページ.....	7
3.3 移行の実行.....	9
3.4 クイック修正.....	13
3.5 移行レポート.....	16
3.6 ビットフィールドの移行.....	17
4 まとめ.....	18
5 参考資料.....	18

## 図の一覧

図 2-1. C2000 IDEA - 特長パネル.....	4
図 2-2. Any-to-Any デバイス Driverlib 移行用 HTML ページ.....	5
図 3-1. C2000 IDEA - 移行フロー.....	6
図 3-2. C2000 IDEA - プロジェクト検出.....	7
図 3-3. 移行設定ページ.....	8
図 3-4. プロジェクトの現在のデバイスを手動更新.....	9
図 3-5. ファイル移行の実行 (拡張子ツリーとエディタビュー).....	10
図 3-6. ファイル移行の進行状況バーと完了通知.....	10
図 3-7. 現在のファイルの継続的な移行チェックを有効にする.....	11
図 3-8. 現在のファイルの継続的な移行チェックを無効にする.....	11
図 3-9. プロジェクト移行の実行 (拡張子ツリー).....	12
図 3-10. プロジェクト移行進行状況バーと完了通知.....	13
図 3-11. プロジェクト移行後の CCS 出力コンソール.....	13
図 3-12. 移行実行後のファイル表示.....	13
図 3-13. 移行の問題 - 問題点の表示と迅速な修正.....	14
図 3-14. 移行資料の HTML ページ.....	14
図 3-15. #IFDEF クイック修正 (F28x-to-F28x) のラップ.....	15
図 3-16. #IFDEF クイック修正 (F28x-to-F29x) のラップ.....	15

☒ 3-17. All Enum Wrap #IFDEF Quick Fix (F28x-to-F29x) .....	15
☒ 3-18. 移行レポートのエクスポート (拡張ツリー).....	16
☒ 3-19. 移行レポート.....	17

## 表の一覧

表 3-1. C2000 デバイス用移行サポート.....	6
表 3-2. さまざまなファイル形式の移行のサポート.....	6

## 商標

C2000™ and Code Composer Studio™ are trademarks of Texas Instruments.

Visual Studio Code™ is a trademark of Microsoft Corporation.

すべての商標は、それぞれの所有者に帰属します。

## 1 はじめに

テキサス インストルメンツの **C2000** ソフトウェア開発キット (SDK) は、リアルタイム制御アプリケーションの開発を加速するために設計された、ソフトウェアとドキュメントの統合スイートを提供します。これらのリソースには、デバイス固有のドライバ、ライブラリ、ペリフェラルのサンプルが含まれており、開発者がプロジェクトを効率的に進めるのに役立ちます。これらの SDK はあらゆる経験レベルのユーザーに対応できるよう設計されており、初めてのユーザーをサポートするための初心者向け機能も備えています。SDK に含まれる **C2000 SysConfig** ツールは、直感的なビジュアル インターフェイスを通じてデバイスやペリフェラルの複雑さを簡素化し、セットアップ プロセスをより効率的にします。初期化、ランタイム、デバッグにおけるユーザー体験をさらに向上させるために、**C2000 IDEA** はツールや関連資料を単一の環境に集約し、より効率的なワークフローを実現します。

このドキュメントでは、**C2000 IDE Assist** の移行機能の使い始め方と、その機能を効果的に活用して **C2000** デバイス間の移行を簡素化・加速する方法について説明します。

### 1.1 はじめに

このセクションでは、拡張機能を使用する開発環境を設定する方法など、**C2000 IDEA** ツールの基本について紹介します。

- **Code Composer Studio (CCS)** バージョン 20.0.0 以降用の最新オフライン インストーラをダウンロードしてインストールします。
- **CCS** を起動し、左側のサイドバーにある「**拡張機能**」タブに移動します。
- **C2000 IDEA** 拡張機能を検索してインストールします。
- **CCS** を再起動して、拡張機能を有効にします。ユーザーは左サイドバーに **C2000 IDEA** のアイコンが表示されるようになります。
- **CCS** で通常どおりに、新規または既存のワークスペースを開きます。

セットアップ手順と追加ガイダンスの詳細については、[C2000 IDE Assist Tool 機能ガイド アプリケーション ノート](#)を参照してください。

## 2 概要

このセクションでは、C2000 IDEA のインターフェイスの概要を紹介し、ツールの操作を容易にする方法について説明します。画面左側にある「C2000 IDEA - 特長」パネルは、本ツールのすべての機能にアクセスするための入り口として機能します。次の画像は、C2000WARE のサンプルを使用してこのパネルを示しており、ユーザーがレイアウトやオプションに慣れるのに役立ちます。

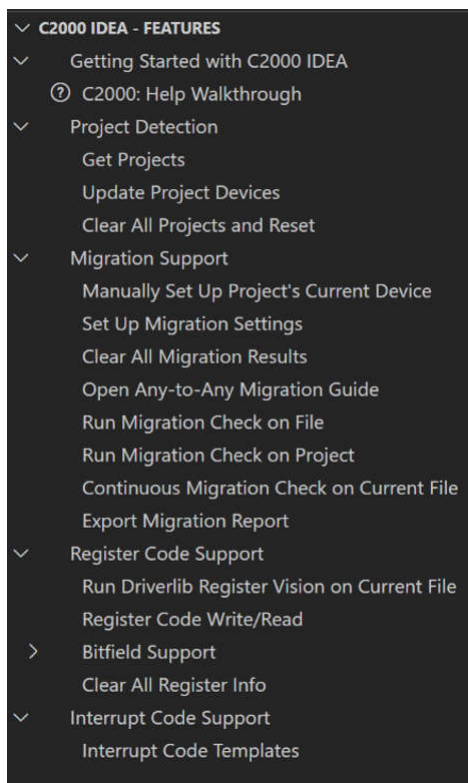


図 2-1. C2000 IDEA - 特長パネル

このパネルでは、専用の移行サポートセクションを使用して、ツールが提供するすべての移行関連機能にすばやく簡単にアクセスできます。ユーザーは、[図 2-2](#) に示すように、*Any-to-Any* 移行ガイドの HTML ページを参照し、現在のデバイスと移行先デバイスの両方に対応する C2000ware のバージョンを選択できます。これにより、ユーザーは driverlib コードの違いを確認し、異なる C2000 デバイス間の 1 対 1 の対応関係を調べることができます。

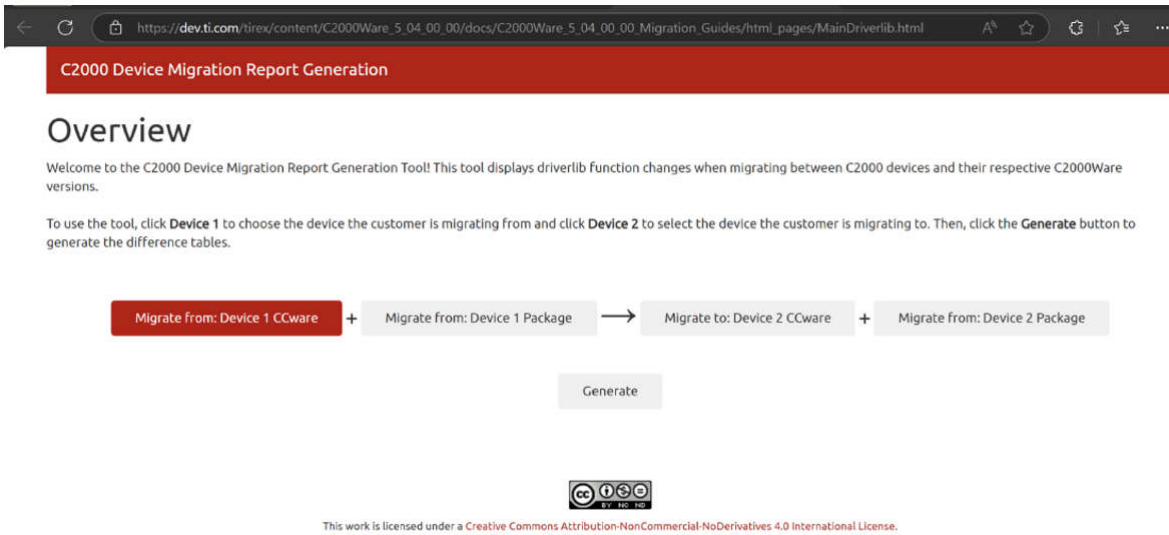


図 2-2. Any-to-Any デバイス Driverlib 移行用 HTML ページ

### 3 移行サポート機能

C2000 IDEA ツールは、ドライバライブラリ(driverlib) コードにおける追加、削除、変更されたアプリケーション ペリフェラル インターフェース (API)、レジスタ、フィールド、列挙体、マクロなどの変更点を自動的に識別し、ハイライトすることで、C2000 デバイス間のコード移行を簡素化します。この自動化により、手作業の負担が軽減され、エラーのリスクが最小限に抑えられ、スムーズかつ効率的な移行プロセスが確実に行われます。この自動化機能は、プロジェクト全体だけでなく、スタンドアロンの driverlib ファイルの移行にも対応しており、さまざまなユーザーの要件に柔軟に対応します。特定のデバイスに対して、このツールはビットフィールド コードの移行も可能にします。C2000 IDEA は、F28x から F28x、または F28x から F29x への移行において、アーキテクチャの違いを検出し、より信頼性の高い移行を実現するための的確な推奨事項を提供することで、移行作業を簡素化します。表 3-1 は、C2000 デバイス間の移行サポートを明確に要約して提供し、ユーザーが Driverlib 実装およびビットフィールド実装の互換性と利用可能なサポートを迅速に評価できるよう支援します。

表 3-1. C2000 デバイス用移行サポート

デバイス ファミリ	Driverlib 移行のサポート	ビットフィールド移行のサポート
F29H85x	あり	なし
F28P55x	あり	なし
F28P65x	あり	なし
F28002x	あり	なし
F28004x	あり	なし
F28003x	あり	なし
F280013x	あり	あり(移行元)
F280015x	あり	なし
F2838x	あり	なし
F2837x	あり	なし
F2807x	あり	なし
F2803x	なし	はい (移行元)

このツールは、表 3-2 に示すように、特定のファイル形式のソース ファイルをサポートしています。

表 3-2. さまざまなファイル形式の移行のサポート

ソース ファイル形式	移行のサポート
*.c	あり
*.h	あり
*.asm	なし
*.cmd	なし
*.lib	なし
リンカ ファイル	なし

図 3-1 はスタンドアロン ファイルまたはプロジェクト全体を移行するために必要な主要ステップを概説し、シームレスかつ効率的な移行プロセスを確実にします



図 3-1. C2000 IDEA - 移行フロー

### 3.1 プロジェクト検出

C2000 デバイスを使用しているすべてのプロジェクトは、プロジェクト取得要素をクリックすることで検出できます。このステップにより、拡張機能は各プロジェクトに対応するデバイスを認識し、各ファイルがどのプロジェクトに属しているかを把握できるようになります。プロジェクト検出により、デバイス間の Driverlib 移行など、拡張機能のリアルタイムでの全機能が有効化されます。一部の拡張機能はプロジェクト検出 (またはデフォルトデバイスの設定) によってのみ利用可能ですが、その他の機能は現在のデバイス情報を入力することで単一ファイルに対して実行できます。

以下の手順に従って、プロジェクトを検出します：

1. ワークスペース内のすべてのプロジェクトを検出するには、以下のいずれかのオプションを使用します：
  - a. CTRL + SHIFT + P を入力し、C2000 をクリックします。プロジェクト取得。
  - b. 拡張機能ツリーの C2000 IDEA - 特長パネルで、プロジェクト検出 > プロジェクト取得をクリックします。
2. 実行すると、拡張子によって検出されたワークスペース内のすべてのプロジェクトが、Extension ツリーの C2000 IDEA - プロジェクトペインに表示されます。プロジェクトを表示し、デバイスバリエーションおよび現在のデバイスの詳細が想定どおりであることを確認します。

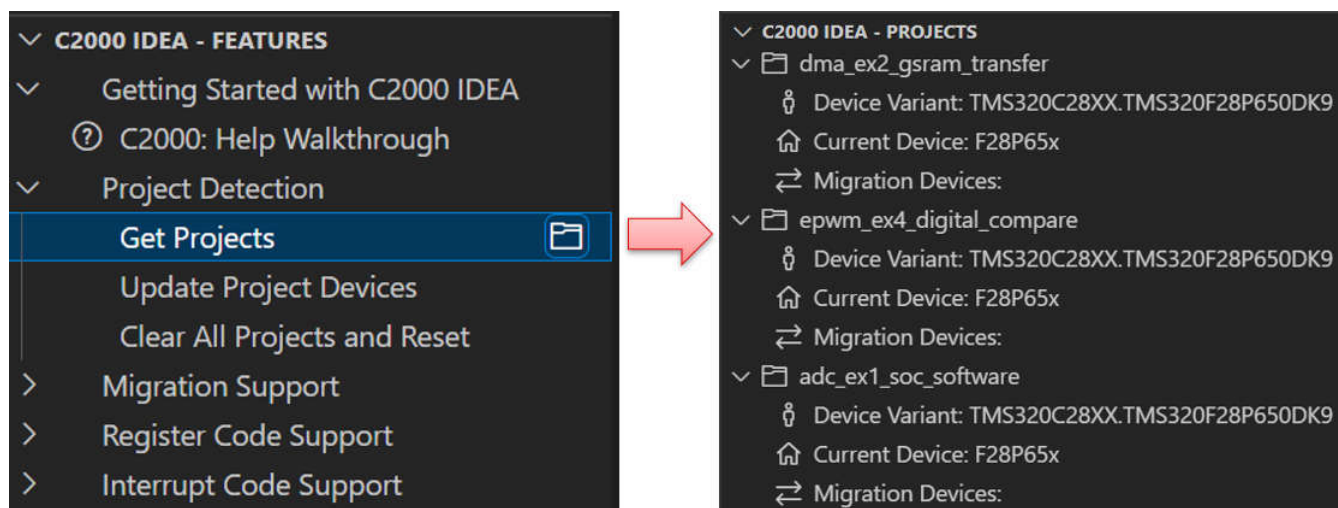


図 3-2. C2000 IDEA - プロジェクト検出

### 3.2 移行設定ページ

このページは、プロジェクト全体にわたる移行プロセスをカスタマイズするための包括的なインターフェイスを提供しており、ユーザーは特定の要件に応じて移行設定を構成することができます。

このページにアクセスするには、次のいずれかの方法を使用します。

1. CTRL + SHIFT + P を入力し、C2000 をクリックします。移行設定。
2. 拡張ツリーのメソッド：
  - a. 移行サポート > 移行設定を C2000 IDEA - 特長ペインで選択します。
  - b. 図 3-3 に示すように、C2000 IDEA - プロジェクトペインで検出されたプロジェクトの下にある「移行デバイス」の横にあるアイコンをクリックします。

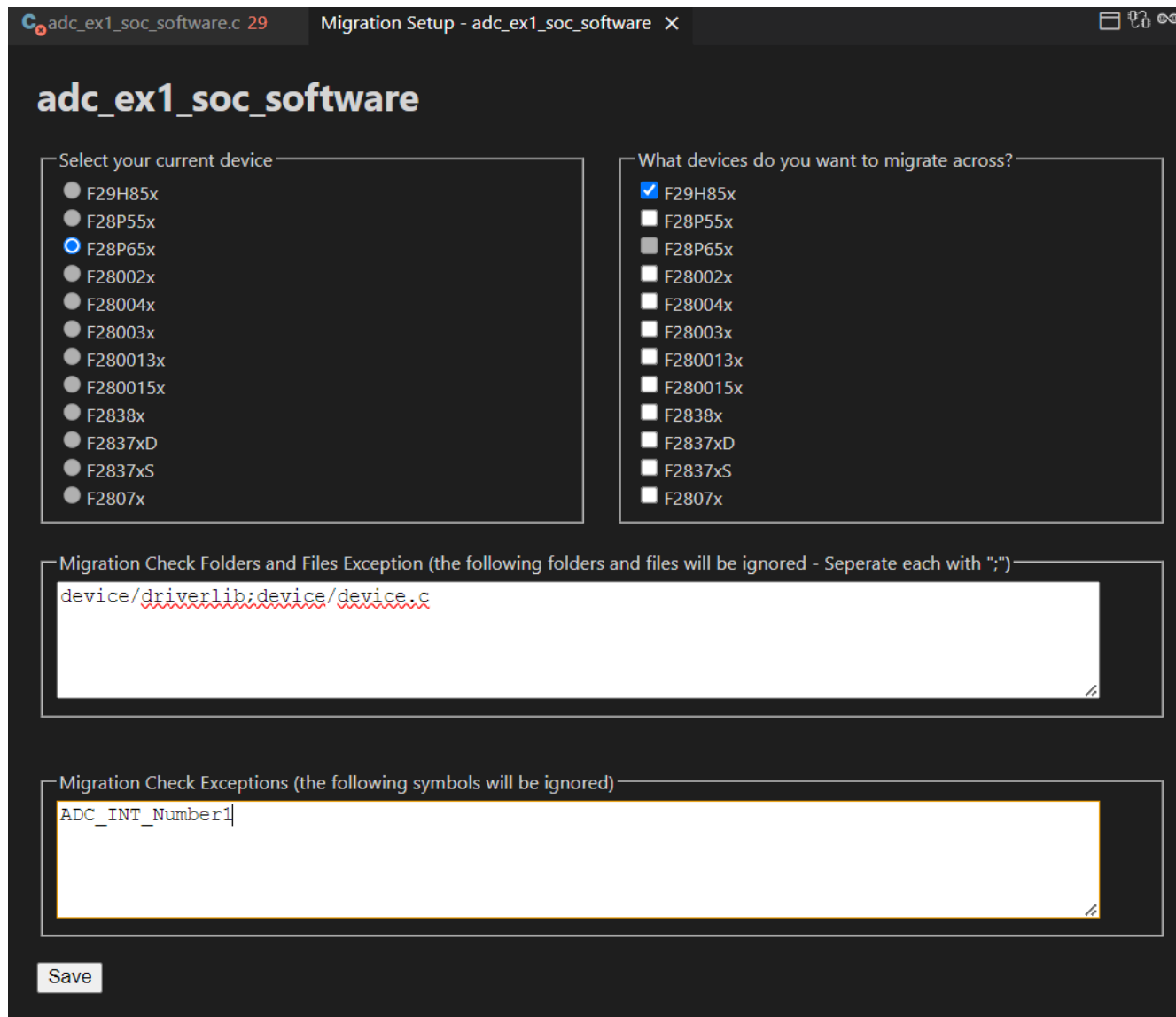


図 3-3. 移行設定ページ

移行設定ページでは、前のステップで完了したプロジェクト検出プロセスに基づいて、現在のデバイスを自動的に検出し識別します。デバイスはあらかじめ定義されているため、ユーザーは移行設定ページ内で直接変更することはできません。

プロジェクトの現在のデバイスを変更する必要がある場合は、次の方法で変更できます。

1. CTRL + SHIFT + P を入力し、**C2000** をクリックします。プロジェクトの現在のデバイスを手動で設定する。
2. **C2000 IDE** -プロジェクトペインで、検出されたプロジェクトの下にある現在のデバイスの横にあるアイコンをクリックします。



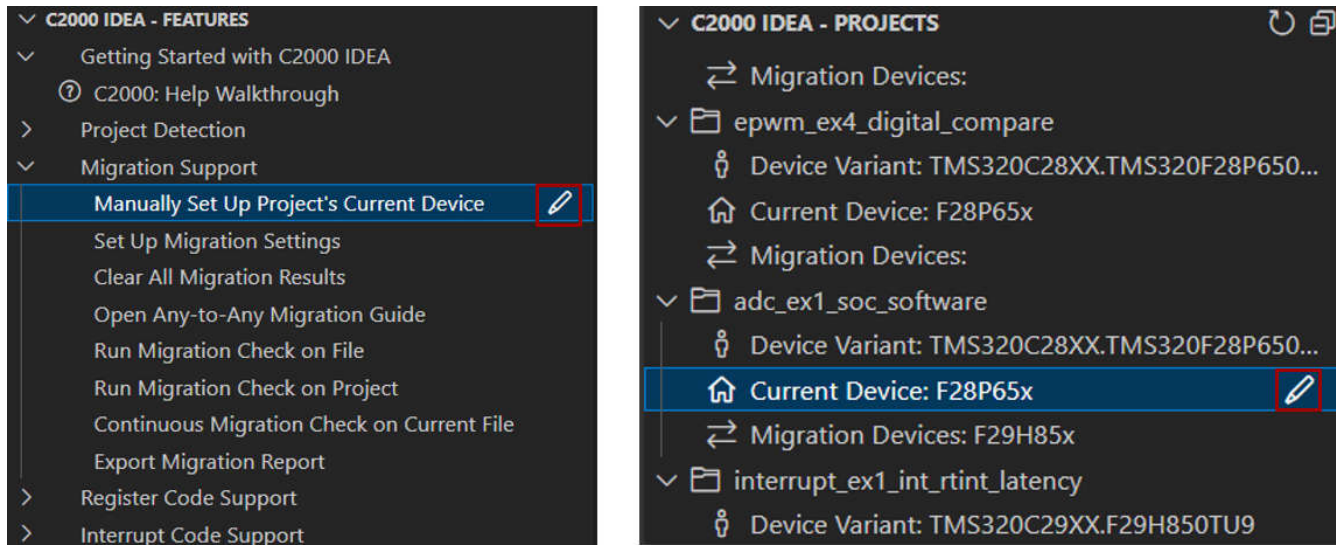


図 3-4. プロジェクトの現在のデバイスを手動更新

このページでは、デバイスを選択し、ファイル、フォルダ、およびコード変更の例外を指定することで、移行プロセスをカスタマイズできます。

- **移行デバイスの選択:**
  - 移行設定ページの選択ボックスをオンにすると、1 つまたは複数の移行デバイスを選択できます。
- **ファイルとフォルダの除外:**
  - 移行時に無視するファイルとフォルダを、移行ファイルとフォルダの例外セクションで指定します。
  - エントリはセミコロン (;) で区切る必要があります。
- **コード変更の除外:**
  - 移行中に無視されるコードの変更を移行チェックの例外セクションで指定します。
  - エントリはセミコロン (;) で区切る必要があります。

必要な詳細を入力したら、移行セットアップページを保存します。拡張ツリー内の「C2000 IDEA -プロジェクト」パネルで、移行デバイスの情報がそれに応じて更新されます。

### 3.3 移行の実行

C2000 IDEA 拡張機能は、`driverlib` 形式で記述されたプロジェクトまたはファイルに対して、F28x から F28x または F28x から F29x への移行チェックを実行するために使用できます。このコード スタイルは、`driverlib` のソース ファイルで定義されている関数の呼び出しを使用することで特徴づけられます (例: `Device_init ()`) および/または `_o_syntax` を含むレジスタへのアクセス。

#### 3.3.1 スタンドアロン ファイルでの移行チェックの実行

この拡張機能では、スタンドアロン ファイルに対して移行チェックを実行することができます。移行チェックを実行するには、次の手順を実行します。

- スタンドアロン ファイルを開き、そのファイルがエディター内でアクティブ ファイルとして設定されていることを確認します。
- 次のいずれかの方法を使用して、移行チェックを実行します。
  1. **CTRL + SHIFT + P** を入力し、**C2000** をクリックします。ファイルの移行チェックを実行します。
  2. 拡張ツリーのメソッド:
    - a. 移行サポート > ファイルに対して移行チェックを実行を **C2000 IDEA - 特長** ペインで選択します。
    - b. 図 3-5 に示すように、アクティブ ファイル エディタの右上にあるアイコンをクリックします。

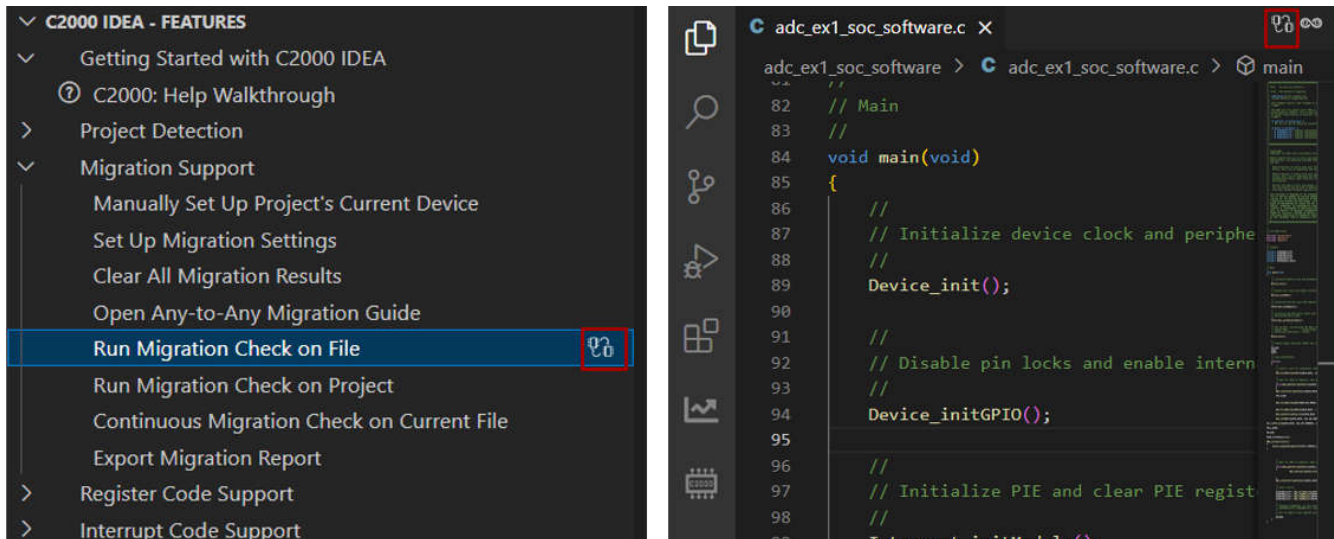


図 3-5. ファイル移行の実行 (拡張子ツリーとエディタビュー)

アクティブファイルに対して移行チェックを実行すると、画面右下に進行状況バーが表示されます。この進行状況バーはリアルタイムで更新され、現在処理中のファイルパスが表示されます。移行プロセスが完了すると、進行状況バーが同じ場所にあるステータスメッセージに置き換えられ、移行チェックが完了したことが示されます。

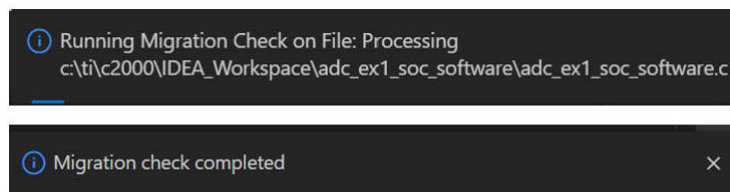


図 3-6. ファイル移行の進行状況バーと完了通知

**現在のファイルの継続的な移行チェック:** この機能は、変更が検出されるたびに、一定の間隔でアクティブファイルに対して自動的に移行チェックを実行します。これにより、ユーザーは各変更後に手動でチェックを再実行することなく、移行に関する懸念点を迅速に特定して、対処できるようになり、効率と作業の流れが向上します。

#### 注

*C2000* の場合は、移行機能を使用しないでください。現行ファイルの継続的な移行チェック機能がツールで有効になっています。

- ユーザーは、次のいずれかの方法を使用して、アクティブファイルの継続的な移行チェックを有効にできます。
  1. CTRL + SHIFT + P を入力し、C2000 をクリックします。現在のファイルの継続的な移行チェックを有効にします。
  2. 拡張ツリーのメソッド:
    - a. 移行サポート > 現在のファイルに対して継続的な移行チェックを実行を C2000 IDEA - 特長ペインで選択します。
    - b. アクティブファイルエディタの右上にあるアイコンをクリックします。

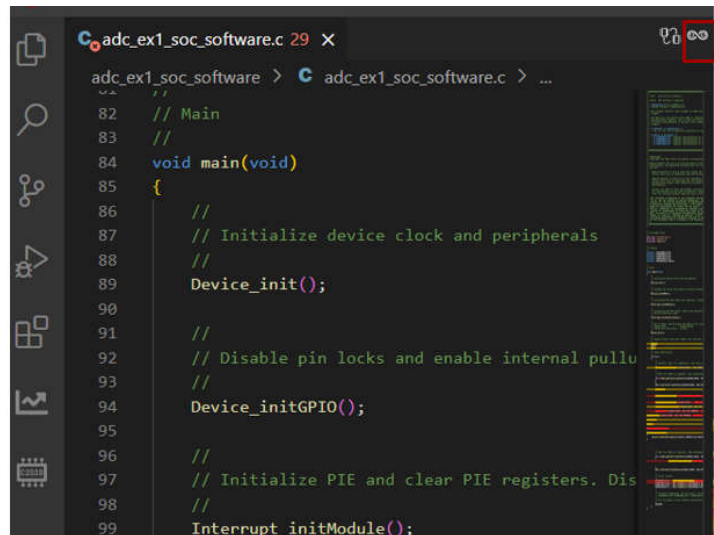
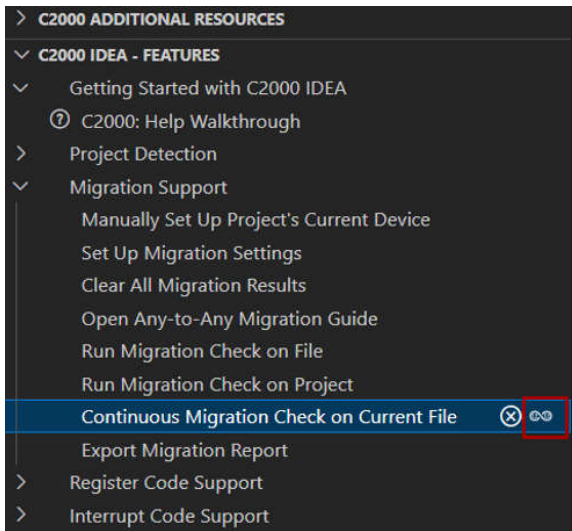


図 3-7. 現在のファイルの継続的な移行チェックを有効にする

- ユーザーは、次のいずれかの方法を使用して、アクティブ ファイルの継続的な移行チェックを無効にできます。
  1. CTRL + SHIFT + P を入力し、C2000 をクリックします。現在のファイルの継続的な移行チェックを無効にします。
  2. 拡張ツリーのメソッド:
    - a. 移行サポート > 現在のファイルに対して継続的移行チェックを実行を C2000 IDEA - 特長ペインで選択します。
    - b. アクティブ ファイル エディタの右上にあるアイコンをクリックします。

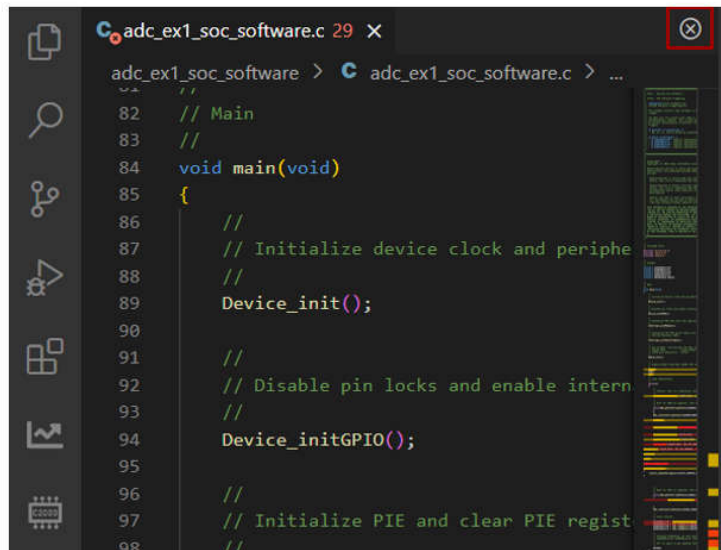
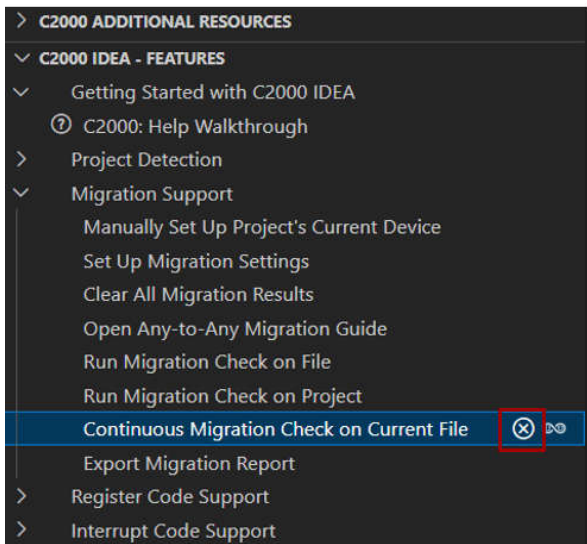



図 3-8. 現在のファイルの継続的な移行チェックを無効にする

### 3.3.2 プロジェクトでの移行チェックの実行

この拡張機能を使用すると、C2000 デバイス プロジェクト全体に対して移行チェックを実行できます。移行チェックを実行するには、次の手順を実行します。

- [移行設定] ページが要件に応じてカスタマイズされていることを確認します。
- プロジェクトで移行チェックを実行するには、次のいずれかの方法を使用します。
  1. CTRL + SHIFT + P を入力し、C2000 をクリックします。プロジェクトで移行チェックを実行します。
  2. 拡張ツリーのメソッド:
    - a. 移行サポート > プロジェクトに対して移行チェックを実行を C2000 IDEA - 特長ペインで選択します。
    - b. C2000 IDEA - プロジェクトペインで 、検出されたプロジェクトの下にある移行先デバイスの横にあるアイコンをクリックします。

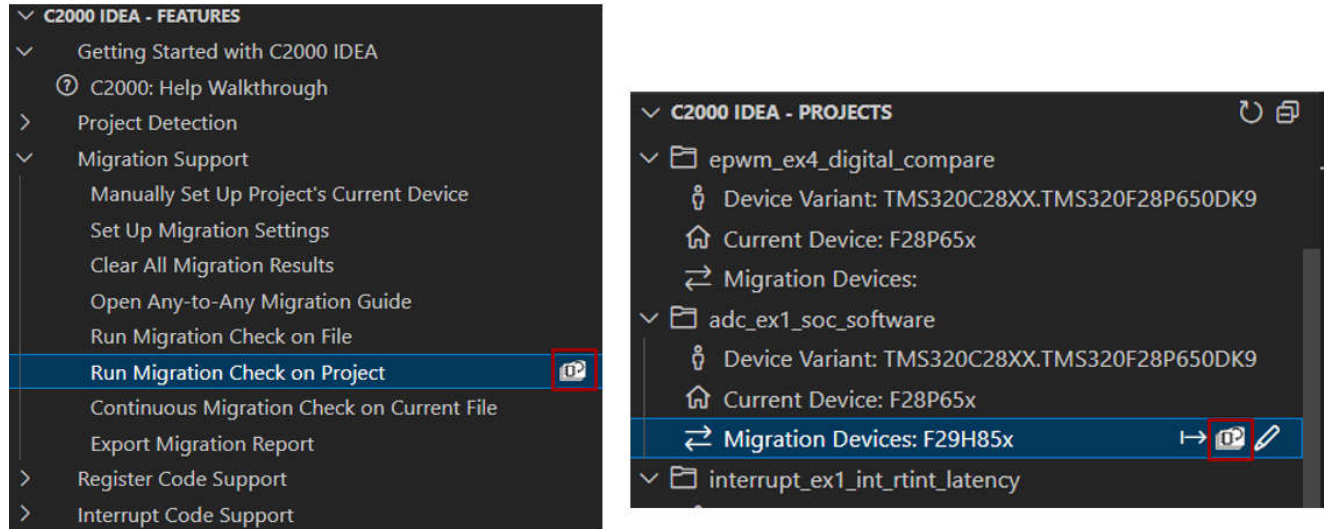


図 3-9. プロジェクト移行の実行 (拡張ツリー)

#### 注

移行設定ページで指定された「無視するファイルやフォルダ」の情報と、移行対象のフォルダやファイルに重複がある場合、拡張機能によってエラーがスローされます。

#### 注

ツールでプロジェクトの移行チェックがすでに実行されている間は、これらの移行機能を使用しないでください。他の移行機能を有効にする前に、画面右下の[プロジェクト名]で移行チェックが完了するまで待ちます。チェックの実行にかかる時間は、プロジェクト内のファイル数、行数、およびコード変更の量によって完全に決まります。移行レポートには、各ファイルにかかった時間が含まれます。

プロジェクト全体に対して移行チェックを実行すると、画面右下に進行状況バーとパーセンテージのインジケータが表示されます。この進行状況バーはリアルタイムで更新され、移行中に現在処理されているファイルのパスが表示されます。

移行プロセスが完了すると、[プロジェクト名]で移行チェックが完了したことを示すステータスメッセージが同じ場所に表示されます。

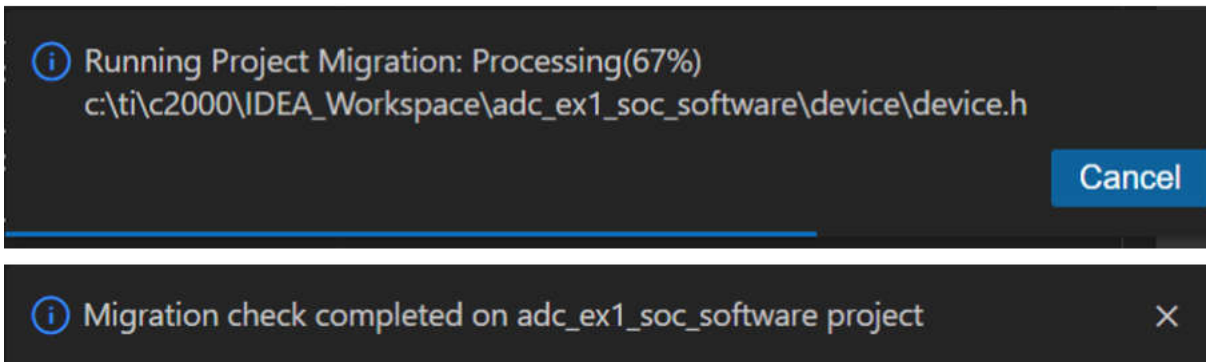


図 3-10. プロジェクト移行進行状況バーと完了通知

プロジェクトの移行の概要は、Code Composer Studio (CCS) の出力コンソールで表示できます。このサマリーでは、無視されるファイルやフォルダなど、移行プロセスに関する情報を提供します。

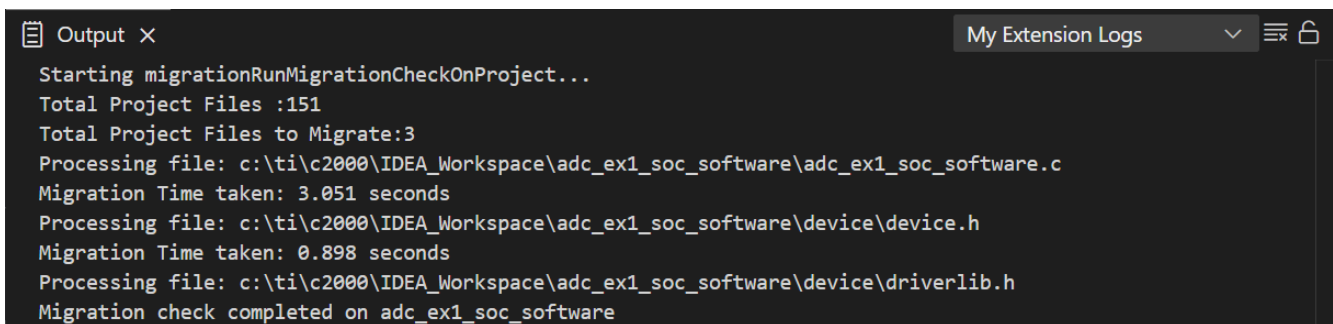


図 3-11. プロジェクト移行後の CCS 出力コンソール

### 3.4 クイック修正

C2000 IDEA 拡張機能は、スタンドアロンファイルまたはプロジェクト全体で移行実行を行う際に、移行に関する懸念点を検出してハイライト表示します。ファイル内の主な移行に関する懸念点はすべて赤い波線で下線が引かれ、それ以外の移行に関する警告は黄色い波線で下線が引かれます。

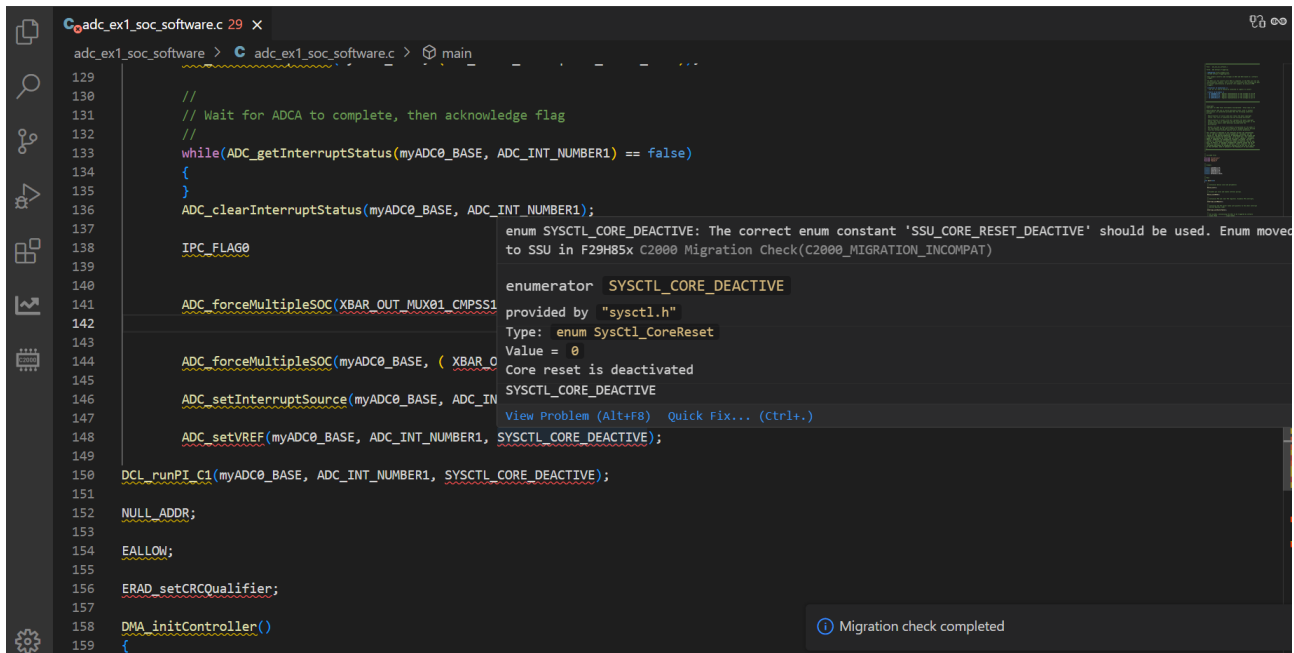


図 3-12. 移行実行後のファイル表示

強調表示されている各移行の問題には、プロンプトメッセージと、問題点の表示および迅速な修正オプションが含まれます。

- 問題点の表示: 現在のデバイスと移行先のデバイスの違いを説明する形で、移行に関する懸念点の詳細な説明が表示されます。
- 迅速な修正: 移行に関する懸念点に効果的に対処するための対応策が提案されます。

```
150 ADC_forceMultipleSOC(myADC0_BASE, ( XBAR_OUT_MUX30_CLB4_OUT6 | SYSCCTL_CORE_DEACTIVE));
adc_ex1_soc_software.c 2 of 14 problems
enum SYSCCTL_CORE_DEACTIVE: This selection is not available on f280013x C2000 Migration Check(C2000_MIGRATION_INCOMPAT)
```

```
ADC_forceMultipleSOC(myADC0_BASE, ( XBAR_OUT_MUX30_CLB4_OUT6 | SYSCCTL_CORE_DEACTIVE));
ADC_setInterruptSource(myADC0_BASE, ADC_INT_NUMBER1, ADC_INT_NUMBER1);
ADC_setVREF(myADC0_BASE, ADC_INT_NUMBER1, SYSCCTL_CORE_DEACTIVE);
```

Quick Fix

- Review migration collateral for F28P65x to F280013x
- Wrap in device specific #IFDEF for F28P65x and F280013x
- Ignore SYSCCTL\_CORE\_DEACTIVE related errors

図 3-13. 移行の問題 - 問題点の表示と迅速な修正

この拡張機能は、ユーザーが移行に関する懸念点を効率的に解決し、デバイス間のスムーズな移行を確実に実行できるように、さまざまなオプションを提供します。ユーザーは、次の分解能から選択できます。

- [現在のデバイス] から [移行先のデバイス] への移行に関する資料を確認します。
  - 特定の移行コードの変更に関する詳細な指針を示す、最新の C2000WARE オンライン移行資料に関する「C2000 デバイス移行レポート生成」へのリンクを開きます。

dev.ti.com/tirex/content/C2000Ware\_5\_04\_00\_00/docs/C2000Ware\_5\_04\_00\_00\_Migration\_Guides/html\_pages/diff\_reports/C2000Ware\_5\_04\_00\_00\_...

ADC_Select	-	ADC_D	D
ADC_Select	-	ADC_E	Select ADCC instance
ADC_SyncInput	ADC_SYNCIN_ECAP7SYNCOUT	-	ADC Syncin is ECAP7SYNCOUT
ADC_Trigger	ADC_TRIGGER_ECAP7	-	eCAP7
ADC_Trigger	-	ADC_TRIGGER_CLB1_OUT27	CLB1 OUT27
ADC_Trigger	-	ADC_TRIGGER_CLB2_OUT27	CLB2 OUT27
ADC_Trigger	-	ADC_TRIGGER_CLB3_OUT27	CLB3 OUT27
ADC_Trigger	-	ADC_TRIGGER_CLB4_OUT27	CLB4 OUT27
ADC_Trigger	-	ADC_TRIGGER_CLB5_OUT27	CLB5 OUT27
ADC_Trigger	-	ADC_TRIGGER_CLB6_OUT27	CLB6 OUT27
ADC_Trigger	-	ADC_TRIGGER_CPU3_TINT0	CPU3 Timer 0, TINT0
ADC_Trigger	-	ADC_TRIGGER_CPU3_TINT1	CPU3 Timer 1, TINT1
ADC_Trigger	-	ADC_TRIGGER_CPU3_TINT2	CPU3 Timer 2 TINT2

Overview  
Summary of Differences  
Pinout  
Incompatibilities and Warnings

adc

- Device Interconnects Differences
- Register Differences
- ADC\_setupSOCRefloCh
- ADC\_setMode
- ADC\_setINLTrim
- ADC\_setVREF
- ADC\_setPPBTripLimits
- ADC\_configureRepeate
- ADC\_isBaseValid
- ADC\_setPrescaler
- ADC\_setupSOC
- ADC\_disableIntRefloCo
- ADC\_selectSOCExtChar

### Register Differences

f28p65x	f29h85x	Description
INTSOCSEL2.SOC8	-	SOC8 ADC Interrupt Trigger Select

図 3-14. 移行資料の HTML ページ

- [現在のデバイス] と [移行先のデバイス] のデバイス固有の #IFDEF を折り込みます。
  - コードの該当行を自動的にプリプロセッサ条件(#IFDEF) で囲み、新しいデバイス用に更新されたバージョンのコードをコンパイルできるようにします。また、ファイル内の適切な場所に現在のデバイス用の #define を追加します。
  - ユーザーは、新しいデバイス用にカスタマイズされた代替コード実装を定義できます。プレースホルダ コメント (// Enter alternate code) を、変更が必要な場所に挿入します。

- この拡張機能には、F28x から F29x への移行に関するほとんどの懸念点に対して、コードの置き換えを提案する機能があります。

```

Start of device specific migration code - F28P65x
#if F28P65x // _DEVICE_MIGRATION_
    ADC_forceMultipleSOC(XBAR_OUT_MUX01_CMPSS1_CTRIPOUTL, ( XBAR_OUT_MUX30_CLB4_OUT6 | SYSCTL_CORE_DEACTIVE));
Change of device specific migration code - F280013x
#elif F280013x // _DEVICE_MIGRATION_
    // Enter alternate code
End of device specific migration code - F28P65x/F280013x
#endif // _DEVICE_MIGRATION_
    
```

図 3-15. #IFDEF クイック修正 (F28x-to-F28x) のラップ

```

Start of device specific migration code - F28P65x
#if F28P65x // _DEVICE_MIGRATION_
    ADC_forceMultipleSOC1(XBAR_OUT_MUX01_CMPSS1_CTRIPOUTL, ( XBAR_OUT_MUX30_CLB4_OUT6 | SYSCTL_CORE_DEACTIVE));
Change of device specific migration code - F29H85x
#elif F29H85x // _DEVICE_MIGRATION_
    // Suggested replacement: ADC_forceMultipleSOC1(XBAR_OUT_MUX01_CMPSS1_CTRIPOUTL, ( XBAR_OUT_CLB4_OUT6 | SYSCTL_CORE_DEACTIVE));
End of device specific migration code - F28P65x/F29H85x
#endif // _DEVICE_MIGRATION_
    
```

図 3-16. #IFDEF クイック修正 (F28x-to-F29x) のラップ

- コード関連エラーを無視
  - 移行設定設定ページの「移行チェックの例外」セクションに追加することで、該当の移行に関する懸念を抑制します。
  - 通常、ユーザーが移行に関する懸念に対して別の方法で手動で対応済みの場合に使用されます。
- すべての列挙修正 [現在のデバイス] と [移行先のデバイス] のデバイス固有の #IFDEF でラップを修正
  - 以前の #IFDEF のクイック修正と同様の機能を持ちますが、列挙型に関する移行の懸念に特化して設計されています。
  - F29H85x に移行している場合、および列挙に関連する複数の変更が同じ行に存在する場合にのみ表示されます。
  - その行の関連する enum 移行に関するすべての懸念に #IFDEF ラッパーを自動的に適用します。

```

Start of device specific migration code - F28P65x
#if F28P65x // _DEVICE_MIGRATION_
    ADC_forceMultipleSOC1(XBAR_OUT_MUX01_CMPSS1_CTRIPOUTL, ( XBAR_OUT_MUX30_CLB4_OUT6 | SYSCTL_CORE_DEACTIVE));
Change of device specific migration code - F29H85x
#elif F29H85x // _DEVICE_MIGRATION_
    // Suggested replacement: ADC_forceMultipleSOC1(XBAR_OUT_CMPSS1_CTRIPOUTL, ( XBAR_OUT_CLB4_OUT6 | SSU_CORE_RESET_DEACTIVE));
End of device specific migration code - F28P65x/F29H85x
#endif // _DEVICE_MIGRATION_
    
```

図 3-17. All Enum Wrap #IFDEF Quick Fix (F28x-to-F29x)

これらのクイック修正オプションを活用することで、ユーザーは手動の労力を大幅に削減し、移行プロセスを簡素化し、デバイス間でコードの互換性を確保できます。

### 3.5 移行レポート

移行レポートは、C2000 デバイス間での移行時にコードの変化を分析する上で重要な役割を果たします。これにより、変更内容の構造的な概要が提供され、ユーザーは移行の複雑さを評価し、潜在的な問題を特定し、最適な移行先デバイスを選定するのに役立ちます。詳細な診断情報とエクスポート機能により、移行レポートは移行プロセスを簡素かつ効率的に進めることを可能にします。

このレポートは、移行を実行した後に、プロジェクト全体または特定のアクティブ ファイルに対して生成することができます。この柔軟性により、開発者は移行の影響を全体的に把握しつつ、重要なセクションに集中して対応することができます。

移行レポートを生成およびエクスポートするには、次の方法があります。

1. **CTRL + SHIFT + P** を入力し、**C2000** をクリックします。移行レポートをエクスポートします。
2. 拡張ツリーのメソッド:
  - a. 移行サポート > 移行レポートのエクスポートを **C2000 IDEA - 特長** ペインで選択します。
  - b. **C2000 IDEA - プロジェクト** ペインで [図 3-18](#)、検出されたプロジェクトの下にある移行先デバイスの横にあるアイコンをクリックします。

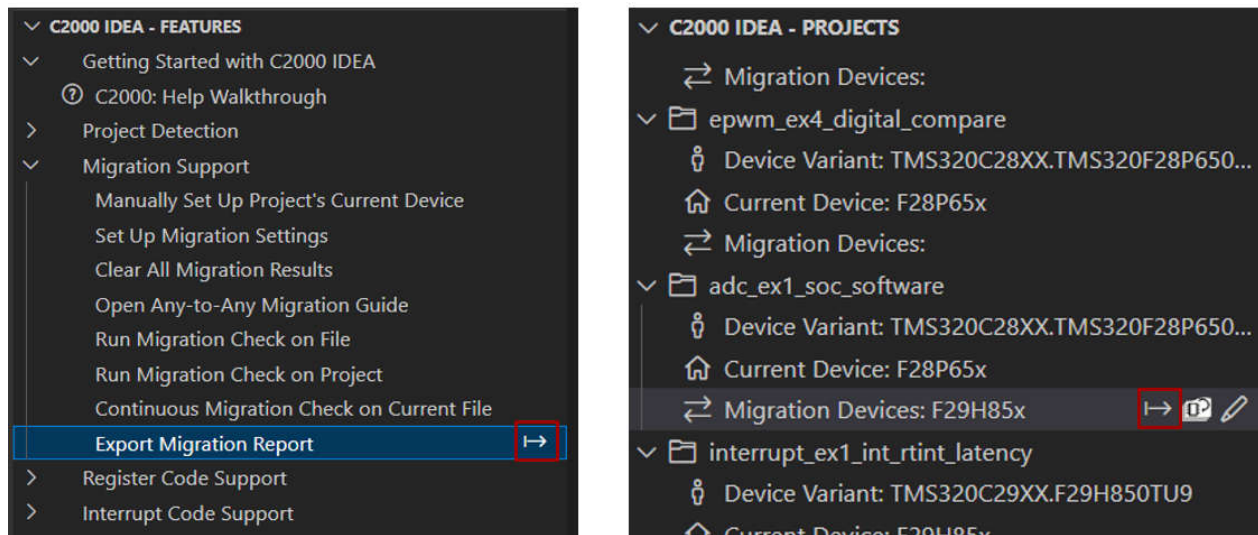


図 3-18. 移行レポートのエクスポート (拡張ツリー)

レポートが生成されると、ファイルブラウザを使用して任意の場所にエクスポートおよび保存することができ、移行データが今後の分析や文書化のためにすぐに利用できるようになります。

#### 移行レポートの主なセクション

1. 移行情報 - 移行プロセスの概要を示します
  - a. **ソース デバイスとターゲット デバイス:** 元のデバイスと選択した移行先デバイスが表示されます。
  - b. **無視されたシンボル、ファイルとフォルダ:** 移行中に考慮されなかった除外されたシンボル、ファイル、およびディレクトリを一覧表示します。
  - c. **処理済みファイル:** 移行を実行した分析済みファイルを指定します。
  - d. **移行時間:** 各ファイルの移行にかかる時間を示し、パフォーマンスと複雑さに関するインサイトを提供します。
2. 移行診断: 移行プロセスで検出された変更と潜在的な問題を強調表示します。
  - a. **分類:** 検出された変更は、重大度に基づいて警告またはエラーに分類されます。
  - b. **コード変更の詳細分析:** ソース デバイスとターゲット デバイスの違いを明確に説明します。
  - c. **場所:** 影響を受けるコードの正確な行 (Ln) と列 (Col) を指定し、問題解決をより効率的にします。



```

Untitled-1
1 Migration Info
2 From: F28P65x
3 To: F29H85x
4 Ignores the following migration incompatibilities: XBAR_OUT_MUX01_CMPSS1_CTRIPOUTL
5 Ignores the following folders: /c:/ti/c2000/IDEA_Workspace/adc_ex1_soc_software/device/driverlib; /c:/ti/c2000/IDEA_Workspace/adc
6 File: c:\ti\c2000\IDEA_Workspace\adc_ex1_soc_software\adc_ex1_soc_software.c
7 Migration time taken: 2.891seconds
8 Warning - CPU Macros ERTM: CPU Macro, ERTM is not available on F29 Devcies[Ln 120, Col 5]
9 Error - enum XBAR_OUT_MUX30_CLB4_OUT6: The correct enum constant 'XBAR_OUT_CLB4_OUT6' should be used. XBAR architecture changed i
10 Warning - function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 (
11 Error - enum SYSCTL_CORE_DEACTIVE: The correct enum constant 'SSU_CORE_RESET_DEACTIVE' should be used. Enum moved to SSU in F29H8
12 Error - enum XBAR_OUT_MUX30_CLB4_OUT6: The correct enum constant 'XBAR_OUT_CLB4_OUT6' should be used. XBAR architecture changed i
13 Warning - function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 (
14 Error - enum SYSCTL_CORE_DEACTIVE: The correct enum constant 'SSU_CORE_RESET_DEACTIVE' should be used. Enum moved to SSU in F29H8
15 Error - enum XBAR_OUT_MUX30_CLB4_OUT6: The correct enum constant 'XBAR_OUT_CLB4_OUT6' should be used. XBAR architecture changed i
16 Warning - function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 (
17 Warning - function ADC_setInterruptSource: Function is compatible, argument intTrigger can be used as ADC_IntTrigger enum from AD
18 Error - function ADC_setVREF: Function changed to ASysCtl_setVREF(from Asysctl driver) with enum change and number of arguments cl
19 Error - enum SYSCTL_CORE_DEACTIVE: The correct enum constant 'SSU_CORE_RESET_DEACTIVE' should be used. Enum moved to SSU in F29H8
20 Error - enum XBAR_OUT_MUX30_CLB4_OUT6: The correct enum constant 'XBAR_OUT_CLB4_OUT6' should be used. XBAR architecture changed i
21 Warning - function ADC_setInterruptSource: Function is compatible, argument intTrigger can be used as ADC_IntTrigger enum from AD
22 Error - enum SYSCTL_CORE_DEACTIVE: The correct enum constant 'SSU_CORE_RESET_DEACTIVE' should be used. Enum moved to SSU in F29H8
23 Error - enum XBAR_OUT_MUX30_CLB4_OUT6: The correct enum constant 'XBAR_OUT_CLB4_OUT6' should be used. XBAR architecture changed i
24 Warning - function ADC_forceMultipleSOC: Function is compatible but socMask argument changed from uint16_t to uint32_t since F29 (
25
26 File: c:\ti\c2000\IDEA_Workspace\adc_ex1_soc_software\device\device.h
27 Migration time taken: 0.877seconds
28 Warning - function SysCtl_setClock: Function is not compatible. Multiple parameter entry needed. Refer to API guide for more deta
29 Warning - function SysCtl_setClock: Function is not compatible. Multiple parameter entry needed. Refer to API guide for more deta
30 Warning - function SysCtl_delay: Function is compatible. Total cycles taken by the function is different. In C29, it takes 4 cycl
31 Warning - function SysCtl_delay: Function is compatible. Total cycles taken by the function is different. In C29, it takes 4 cycl
32
    
```

図 3-19. 移行レポート

レポートをエクスポートして文書として保存することで、開発者は必要なときにいつでも移行の詳細を参照できるようになります。このレポートは、複数の移行先デバイスにわたる移行の複雑さを分析する際に特に有用であり、開発者が潜在的な課題を評価し、最適な移行方法を決定するのに役立ちます。コードの修正内容や潜在的な障害について明確なインサイトが得られるため、開発者は最適な移行先デバイスを選定するうえで、十分な情報に基づいた判断を下すことができま

す。構造化された診断情報、エクスポート機能、詳細なインサイトにより、移行レポートは、C2000 デバイス間のシームレスで効率的な移行を検証するために不可欠なリソースです。

### 3.6 ビットフィールドの移行

C2000 IDEA Extension は、ビットフィールド形式で記述されたファイルに対して、F28x から F28x または F28x から F29x への移行チェックを実行するために使用できます。このコード スタイルは、ビットフィールドのソース ファイルで定義されている関数の呼び出しを使用することで特徴づけられます (例: `InitSysCtrl()`) および/または `[base name].[register name].all` または `[base name].[register name].bit.[field name]` 構文を含むレジスタ アクセス。

ビットフィールド移行を有効にするには、以下の手順に従ってください:

1. C2000 アプリケーションの C 言語コードファイルを開きます。
2. CTRL + SHIFT + P を押し、C2000 を入力し、選択します。ファイルに対してビットフィールド移行チェックを実行しま
- す
3. ファイル内のコードが適用される現在の C2000 デバイスを選択します。
4. ファイルを移行する C2000 デバイスを選択します。
5. 画面右下のステータス バーに、[現在のデバイス]から[移行先デバイス]へのビットフィールドの移行が完了しましたと表示されます。ファイル内のすべての移行に関する懸念点は、赤い波線で下線が引かれます。
6. ファイル全体で問題を確認し、解決します。下線の付いたコードにカーソルを合わせると、以下のオプションが表示されます。
  - a. [問題の表示] を選択して、ファイル内で検出された問題をすばやくループします。
  - b. [クイック修正] を選択して、移行の問題を緩和します。次のいずれかのオプションを選択します。

- i. **[現在のデバイス] から [移行先のデバイス] への移行に関する資料を確認**- このオプションを選択すると、特定の移行パスに対応した最新バージョンの **C2000WARE** を使用したオンラインの移行関連資料へのリンクが開かれます。
- ii. **[現在のデバイス] から [移行先のデバイス] に対応するデバイス固有の #IFDEF でコードを囲む**- このオプションは、コード行を自動的にプリプロセッサのラッパーで囲み、新しいデバイス用に更新されたコードをコンパイルできるようにします。変更したコードで `// 代替コードコメント` を入力し、ファイルのどこかに現在のデバイスの `#define` を追加します。
- iii. **コード関連エラーを無視**- このオプションは、この移行に関する問題を無視します。

## 4 まとめ

テキサス インストルメンツの **C2000 SDK** は、リアルタイム制御アプリケーションの開発を簡素化するために、ドライバやライブラリを含む基本的なツールを提供します。**C2000 SysConfig** のような機能を活用して、コード初期化とペリフェラル構成を簡素化する方法で、エコシステムはあらゆるレベルでデベロッパーをサポートできるように設計済みです。この基礎に基づいて構築された **C2000 IDEA ツール** は、開発環境を一元化し、コーディングとデバッグの効率を向上します。**C2000 IDE Assist Extension (IDEA)** は、デバイスポートフォリオ間のスムーズな移行や従来のコードから新しいデバイスへの移行を支援することで、この取り組みをさらに強化します。

**Code Composer Studio (CCS) 20** 用の強力な拡張機能である **IDEA** は、ユーザーのコードを自動的に解析し、TI MCU 間の移行に伴う潜在的な問題を特定し、文脈に応じたドキュメントとともに最適な解決策を提供することで、ソフトウェア移行を加速します。このツールは移行レポートを生成し、コードの変更点を構造的に分析することで、互換性に関する懸念を明確にし、最適な移行先デバイスの選定を支援します。**IDEA** は初心者と経験豊富な開発者の両方を対象に設計されており、移行作業を簡素化することで生産性を向上させ、デバイス間のスムーズな移行を実現します。**IDEA ツール** は、開発のあらゆる段階でライブ アシスタンスを統合することで、組み込みソフトウェアの移行に必要な時間と労力を大幅に削減します。高度な自動化機能とシンプルなインターフェイスにより、**IDEA ツール** はリアルタイム マイコン向けの効率的かつ高性能な開発を可能にする、組み込みソフトウェア分野における最初のソリューションとなっています。

## 5 参考資料

ツールとソフトウェア:

- テキサス インストルメンツ、『[C2000 IDEA Open VSX \(VSIX ダウンロード\)](#)』
- テキサス インストルメンツ、『[C2000 IDEA GitHub レポジトリ \(VSIX ダウンロード\)](#)』
- テキサス インストルメンツ、『[Code Composer Studio \(CCS\) IDE](#)』
- テキサス インストルメンツ、『[C2000WARE \(F28x SDK\)](#)』
- テキサス インストルメンツ、『[F29X-SDK \(F29x SDK\)](#)』

資料:

- テキサス インストルメンツ、『[C28x アカデミー - リソース移行](#)』
- テキサス インストルメンツ、『[C29x アカデミー - リソース移行](#)』
- テキサス インストルメンツ、『[F28x ~ F29x ソフトウェア移行ガイド](#)』
- テキサス インストルメンツ、『[アプリケーション ソフトウェアから C29 CPU への移行アプリケーション ノート](#)』
- テキサス インストルメンツ、『[TMS320F2837x、TMS320F2838x、TMS320F28P65x から TMS320F29H85x への移行ユーザー ガイド](#)』
- テキサス インストルメンツ、『[C2000 設計 & 開発](#)』

## 重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、[テキサス・インスツルメンツの販売条件](#)、または [ti.com](https://www.ti.com) やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。テキサス・インスツルメンツがこれらのリソースを提供することは、適用されるテキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2025, Texas Instruments Incorporated

## 重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ(データシートを含みます)、設計リソース(リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適したテキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとし、ます。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されているテキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、[テキサス・インスツルメンツの販売条件](#)、または [ti.com](https://www.ti.com) やかかるテキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。テキサス・インスツルメンツがこれらのリソースを提供することは、適用されるテキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2025, Texas Instruments Incorporated