User's Manual

統合開発環境 e^2 studio

ユーザーズマニュアル　入門ガイド

ルネサスマイクロプロセッサ
RZファミリ

本資料に記載の全ての情報は本資料発行時点のものであり、ルネサス エレクトロニクスは、
予告なしに、本資料に記載した製品または仕様を変更することがあります。
ルネサス エレクトロニクスのホームページなどにより公開される最新情報をご確認ください。

ルネサス エレクトロニクス
www.renesas.com

Rev.1.02　2020.09

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

   標準水準：　コンピュータ、OA機器、通信機器、計測機器、AV機器、家電、工作機械、パーソナル機器、産業用ロボット等

   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

   当社製品は、データシート等により高信頼性、Harsh environment向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注1.　本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注2.　本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1　2017.11)

本社所在地

〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

© 2020 Renesas Electronics Corporation. All rights reserved.

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策

   CMOS製品の取り扱いの際は静電気防止を心がけてください。CMOS製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

   電源投入時は、製品の状態は不定です。電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS製品の入力がノイズなどに起因して、V_{IL}（Max.）からV_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）からV_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

このマニュアルの使い方

このマニュアルは、アプリケーション・システムを開発する際の統合開発環境である e² studio について説明します。

e² studio は RX ファミリ、RL78 ファミリ、RZ ファミリ用の統合開発環境で、ソフトウェア開発における、設計、実装、デバッグなどの各開発フェーズに必要なツールを単一のプラットフォームに統合しています。

統合することで、さまざまなツールを使い分けずにすべての開発を本製品のみで行うことができます。

対象者	このマニュアルは、e² studio を使用してソフトウェアやハードウェアのアプリケーション・システムを開発するユーザを対象としています。	
目的	このマニュアルは、ユーザがターゲットデバイスを使用してハードウェアやソフトウェアのシステム開発を始める際の e² studio の機能を説明します。	
構成	このマニュアルは、大きく分けて次の内容で構成しています。 第 1 章　概説 第 2 章　インストール 第 3 章　プロジェクトのインポートおよび作成 第 4 章　ビルド 第 5 章　デバッグ 第 6 章　ヘルプ	
読み方	このマニュアルを読むにあたっては、電気、論理回路、マイクロプロセッサに関する一般知識が必要となります。	
凡例	データ表記の重み	:左が上位桁、右が下位桁
	アクティブ・ロウの表記	:XXX（端子、信号名称に上線）
	注	:本文中につけた注の説明
	注意	:気をつけて読んでいただきたい内容
	備考	:本文中の補足説明
	数の表記	:10 進数 … XXXX

目次

1. 概説 1
1.1 システム構成 1
1.2 システム要件 1
1.3 サポートするコンパイラ 2
1.4 コンパイラをアップデートするための追加ツール 2
1.5 サポートするエミュレータ 2
2. インストール 3
2.1 e² studioのインストール（64ビット版） 3
2.2 e² studioのインストール（32ビット版） 13
2.3 追加ソフトウェアのインストール 23
2.4 e² studioのアンインストール 26
2.5 e² studioのメジャーバージョンアップ 26
2.6 インターネットを介したプラグイン更新 27
2.7 ツールチェーンへのe² studioの登録 31
3. プロジェクトのインポートおよび作成 33
3.1 既存プロジェクトのインポート 33
3.2 新規プロジェクトの作成 36
4. ビルド 55
4.1 ビルドオプションの設定 55
プロジェクトのビルド 58
ビルド構成のレポート 59
5. デバッグ 60
5.1 既存デバッグ構成の変更 61
5.2 新規デバッグ構成の作成 65
5.3 Launch Bar 67
5.4 基本的なデバッグ機能 68
5.5 デバッグ手順（RZ/A2Mの場合） 85
6. ヘルプ 86

1. 概説

ルネサス製マイクロプロセッサをサポートする e² studio は、Eclipse ベースの組み込み向け開発環境です。e² studio は、ビルドフェーズ（エディタ、コンパイラ、リンカコントロールなど）から、拡張 GDB インタフェースをサポートするデバッグフェーズをサポートするオープンソース Eclipse CDT (C/C++ 開発ツール)プロジェクトを基本に開発されたものです。

この章では、ハイエンド ARM ベースのルネサス製マイクロプロセッサである RZ ファミリ向けアプリケーションを開発するための、e² studio のシステム構成と動作環境を説明します。

1.1 システム構成

一般的なシステム構成の例を以下に示します。

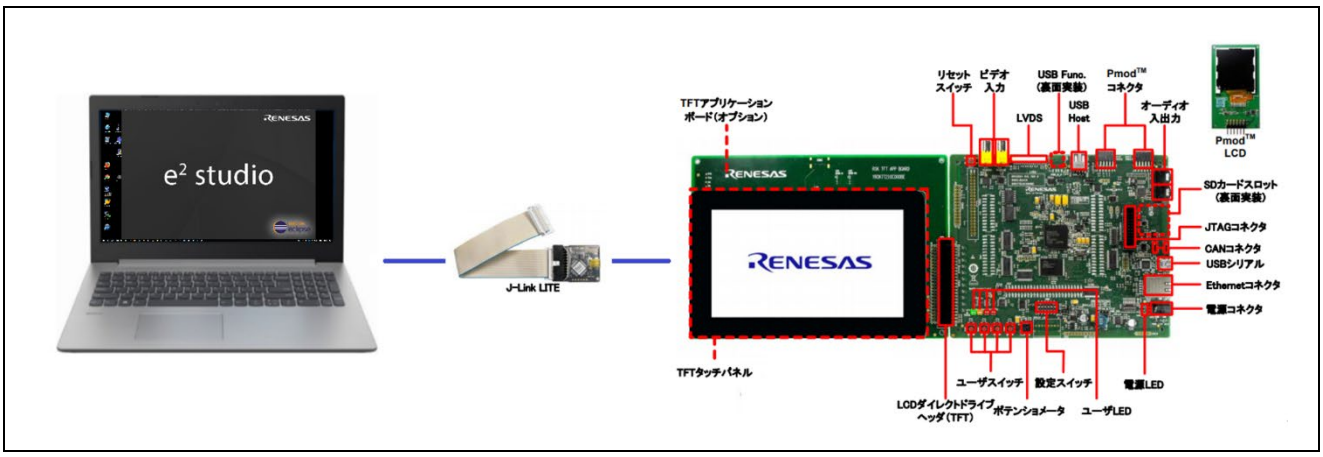

図 1-1 システム構成（RZ/A1H Renesas Starter Kit の例）

1.2 システム要件

本製品に対するシステムの必要条件を以下に示します。

PC ハードウェア環境：

- プロセッサ : 1GHz 以上(ハイパースレッディング及びマルチコア CPU をサポートする)
- メインメモリ : 4GB 以上
- ハードディスク容量 : 空き容量 2GB 以上
- ディスプレイ : 解像度 1,024 x 768 ピクセル以上; 65,536 色以上
- インタフェース : USB 2.0 (ハイスピードまたはフルスピード) ハイスピードが望ましい

動作環境：

- Windows® 10 (32 ビット版、64 ビット版)
- Windows® 8.1 (32 ビット版、64 ビット版)

e² studio 64ビット版（2020-04以降）は64ビット版のOSが必須です。

1.3 サポートするコンパイラ

RZ ファミリデバイスでは、ARM アーキテクチャ用 GNU コンパイラを使用します。（各デバイスには、GNU コンパイラの推奨バージョンがあります。詳細は、https://www.renesas.com/rz を参照してください。）e² studio を[追加ソフトウェア]でインストールする際、コンパイラをインストールすることができます（インターネット接続が必要です。2.1 節を参照してください）。あるいは、コンパイラを以下の Web ページからダウンロードして、個別にインストールすることもできます（ツールチェーンの登録が必要です）。

- GNU ARM Embedded toolchain (GCC 6; 6-2016-q4-major または上位バージョン):
  https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads
- GNU ARM Embedded toolchain (GCC 注 5; 5-2016-q3-update または下位バージョン):
  https://launchpad.net/gcc-arm-embedded/+download
- GNUARM-NONE toolchain:
  https://gcc-renesas.com/rz/rz-download-toolchains/

コンパイラのインストールに関しては 2.2 節、ツールチェーンを統合してコンパイラを e² studio に登録するには、2.6 節を参照してください。

注意：‘GCC’とは、‘GNU Compiler Collection’を指します。（https://gcc.gnu.org/ を参照してください。）

1.4 コンパイラをアップデートするための追加ツール

‘Libgen Update for GNU ARM Embedded Toolchains’ は、組み込みシステム向けに実装される標準 C ライブラリである、‘newlib’ 専用ライブラリのビルドに使用されるツールです。ライブラリの"prebuild"バージョン（ツールチェーンを作成するときの構成でビルドされるライブラリ）の代わりに、ユーザは、カスタムオプションを使用し、Libgen Update によって独自の"projectbuild"ライブラリビルド構成を持つことができます。

e² studio をインストールする際、Libgen Update for GNU ARM Embedded Toolchains を選択してインストールすることができます（インターネット接続が必要です）。あるいは、コンパイラを以下の Web ページからダウンロードすることができます。

- Libgen for GNU ARM embedded toolchain (V1.2018.02 または上位バージョン)
  https://gcc-renesas.com/rz/rz-download-toolchains/

Libgen Update for GNU ARM Embedded Toolchains のインストールに関しては、2.2 節を参照してください。

1.5 サポートするエミュレータ

Segger J-Link Lite (RZ) および、その他エミュレータ
詳細は、RZ ファミリ製品の Web ページ (https://www.renesas.com/rz) を参照してください。

2. インストール

最新の e² studio 統合開発環境インストーラパッケージはルネサスウェブサイトから無償でダウンロードできます。ルネサスエレクトロニクスは現在 e² studio の 32 ビット版と 64 ビット版両方をサポート致します。

詳細な情報は e² studio の製品情報ページ（https://www.renesas.com/e2studio）から確認できます。ソフトウェアをダウンロードするには MyRenesas ページのルネサスアカウントにログインしてください。

この章では、e² studio のインストールおよびアンインストールについて説明します。

e² studio インストーラは e² studio の「インストール」および「アップグレード」、「修正」を提供致します。ただし、V5.x から V6.y もしくは V6.mから V7.n などのインストーラとメジャーバージョンが異なる e² studio には「アップグレード」や「修正」を適用しないでください。

「インストール」を選択して別なフォルダにインストールするか、旧版を一旦アンインストールしてください。

2.1 e² studio のインストール（64 ビット版）

1. e² studioインストーラをダブルクリックしてe² studioインストールウィザードページを開いてください。

   注： e² studio が既に PC にインストールされた場合は図 2-1 の通りに変更、削除、インストールの選択肢が表示されます。

図 2-1 64 ビット版 e² studio のインストールウィザード

2. ようこそページ

インストール先を変更する場合は、[変更...]をクリックして指定してください。[次へ(N)] をクリックします。

注 1： 複数のバージョンの e² studio インストールが必要な場合も、[変更...]をクリックしてインストール先を指定してください。

注 2： インストール先には英文字、数字、アンダーバーのみを含むフォルダパスを指定してください。
また、Windows のアカウント名の中に英文字、数字、アンダーバー以外の文字がある場合、e² studio の動作に問題がある可能性がありますので、Windows のアカウント名にも英文字、数字、アンダーバーだけ使用することをお勧めします。（詳細は e² studio V2020-04 以降のダウンロードページに掲載している注意事項をご参照ください。）

図 2-2 64 ビット版 e² studio のインストール：ようこそページ

3. デバイス・ファミリー

インストールするデバイス・ファミリーを選択できます(複数選択可)。RZファミリのソフトウェアを開発するために[RZデバイス・サポート]チェックボックスを選択してください。[次へ(N)] をクリックします。

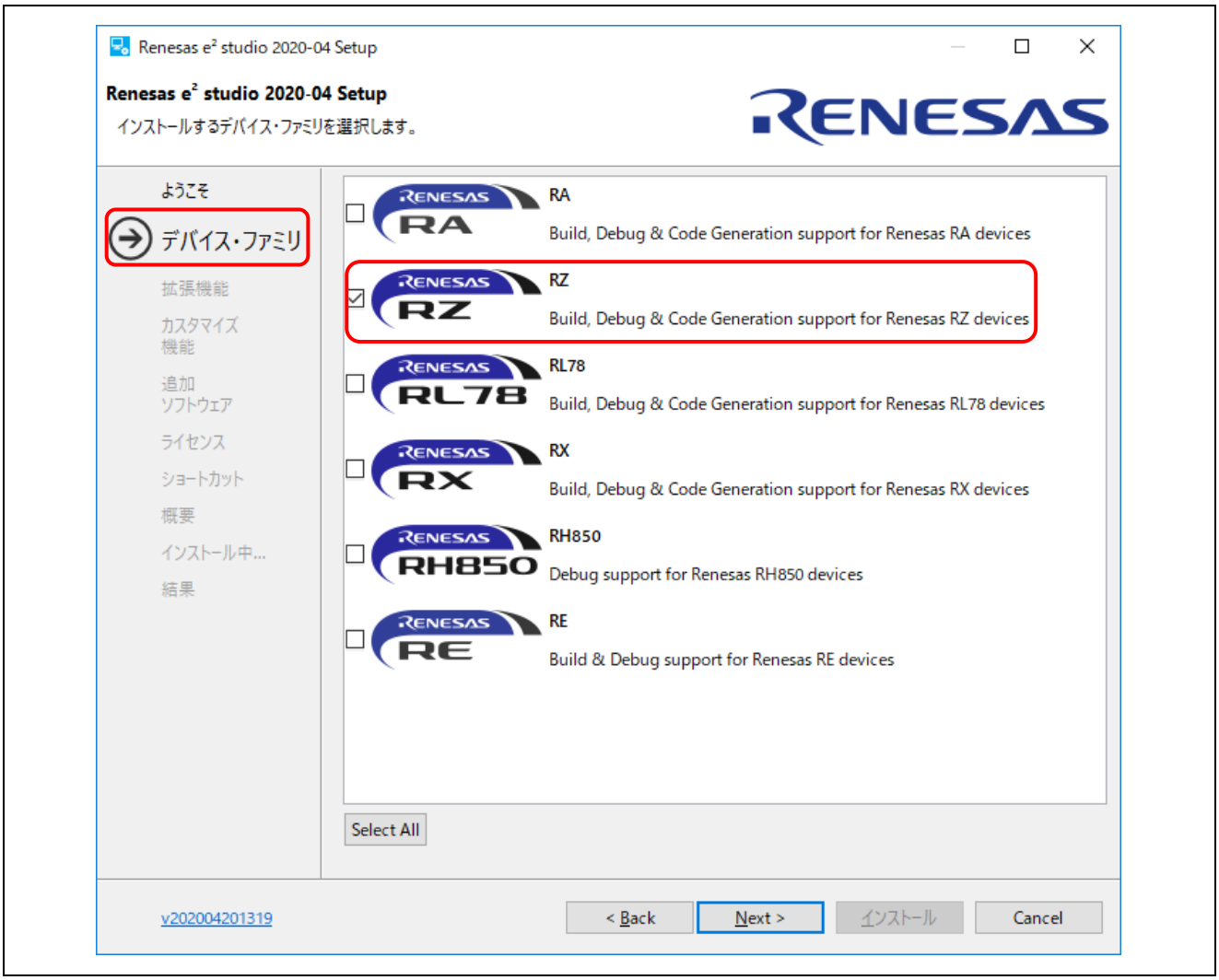

図 2-3 64 ビット版 e² studio のインストール：デバイス・ファミリ

4. 拡張機能

インストールする追加コンポーネント（言語パック、SVN および Git サポート、RTOS サポート）を選択してください。

FreeRTOS、OpenRTOS などの RTOS を使用している場合は、[RTOS]を選択してください。

[次へ] をクリックします。

注意： 日本語の言語パックを指定しないとメニュー等が日本語表示になりません。

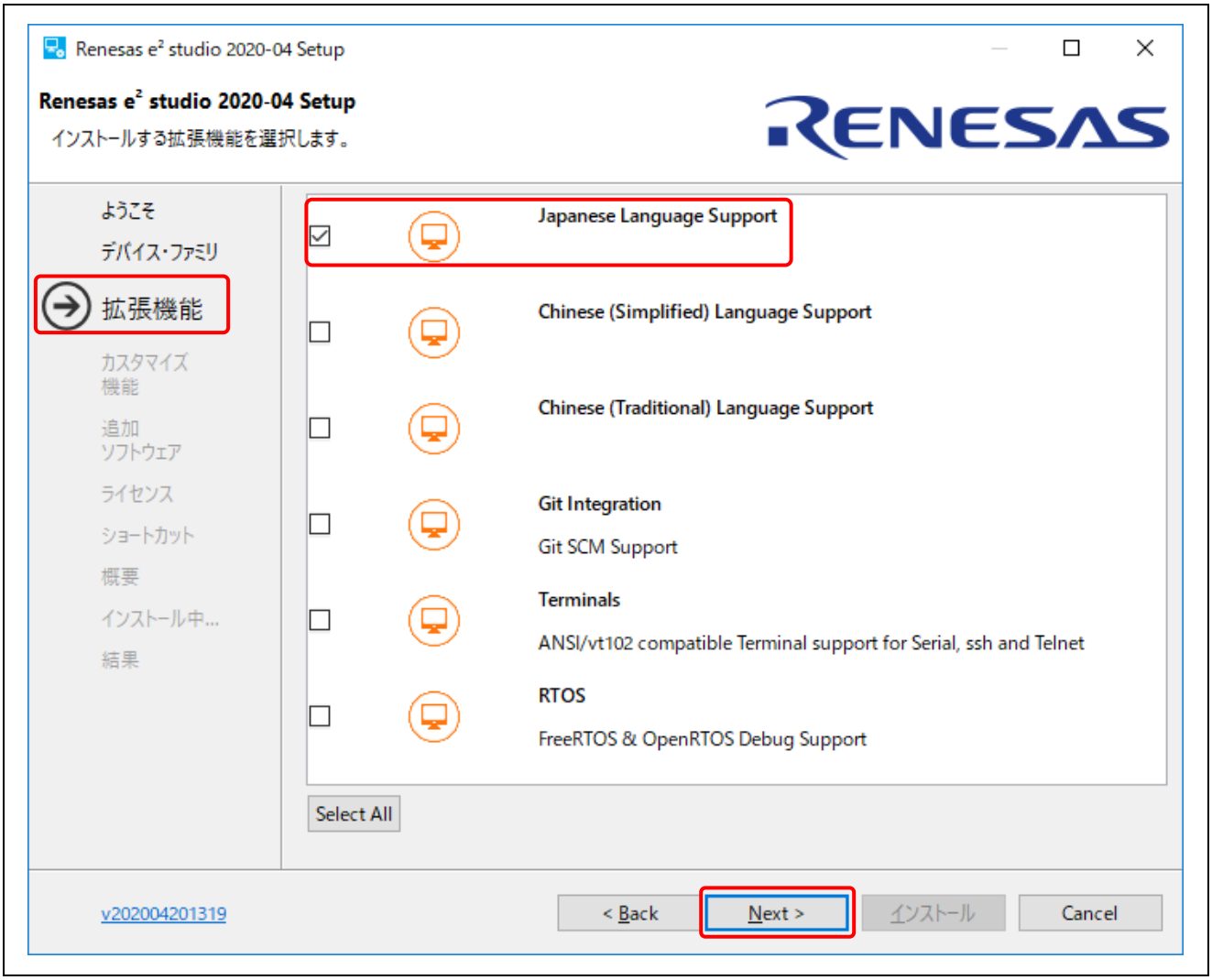

図 2-4 64 ビット版 e² studio のインストール：拡張機能

5. コンポーネント
   [デバイス・ファミリー] で選択した必要なコンポーネントは、すべて自動的に選択されます。
   選択したコンポーネントを確認し、[次へ] をクリックします。

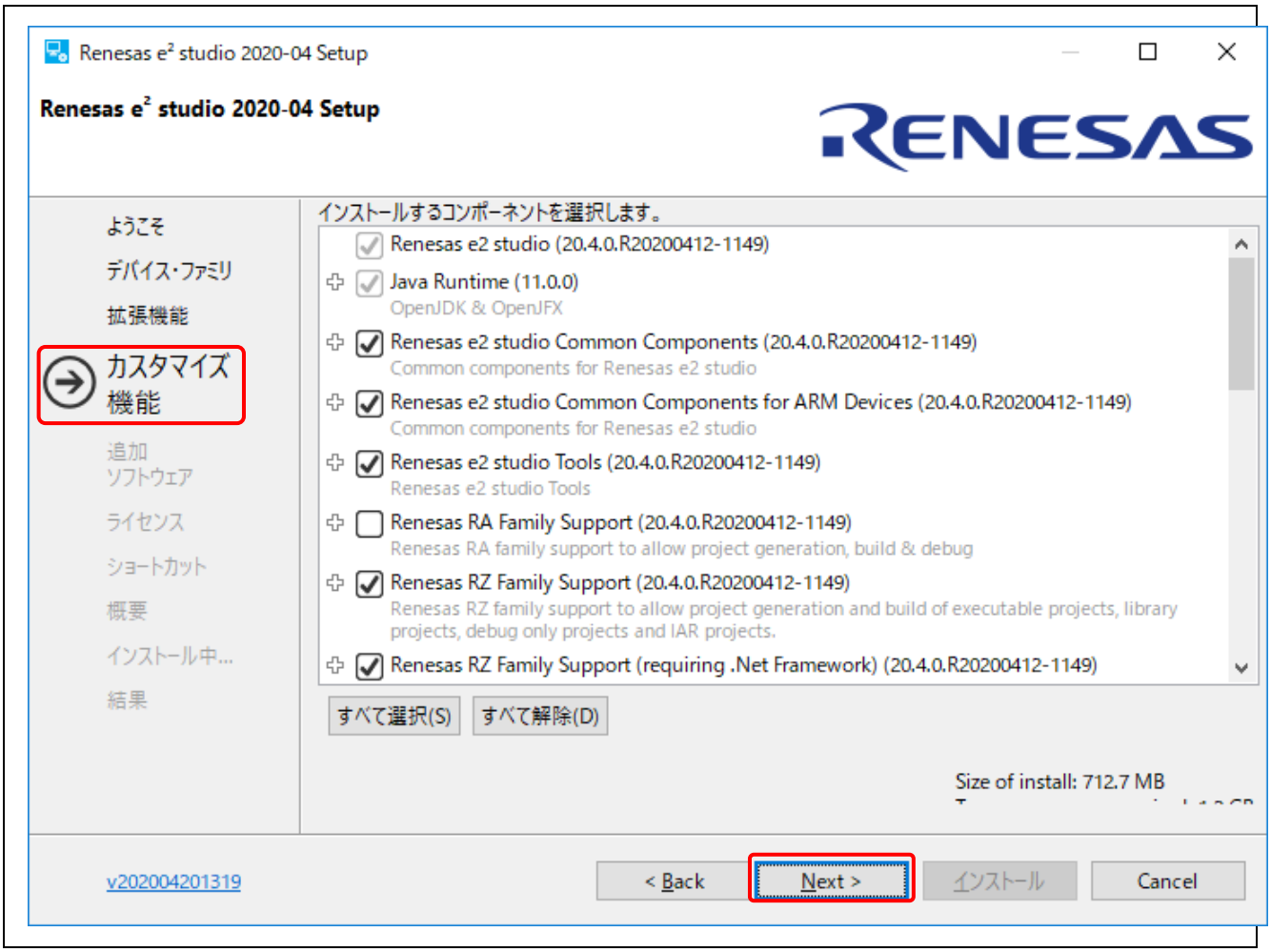

図 2-5　64 ビット版 e² studio のインストール：カスタマイズ機能

6. 追加ソフトウェア

GCC ARM Embedded（GNU ARM Embedded toolchain と同様）、LibGen for GCC ARM Embedded などの追加ソフトウェアが選択されていることを確認してください。（自動的に選択された追加ソフトウェアの一覧は、選択したデバイス・ファミリーによって決定します。）選択したソフトウェアのインストーラは、e² studio インストーラに呼び出されます。（詳細は、2.3 節を参照してください。）一覧に含まれない、GCC ARM Embedded の他のバージョンを使用したい場合は、別途インストールしてください（1.3 節を参照）。[次へ] をクリックします。

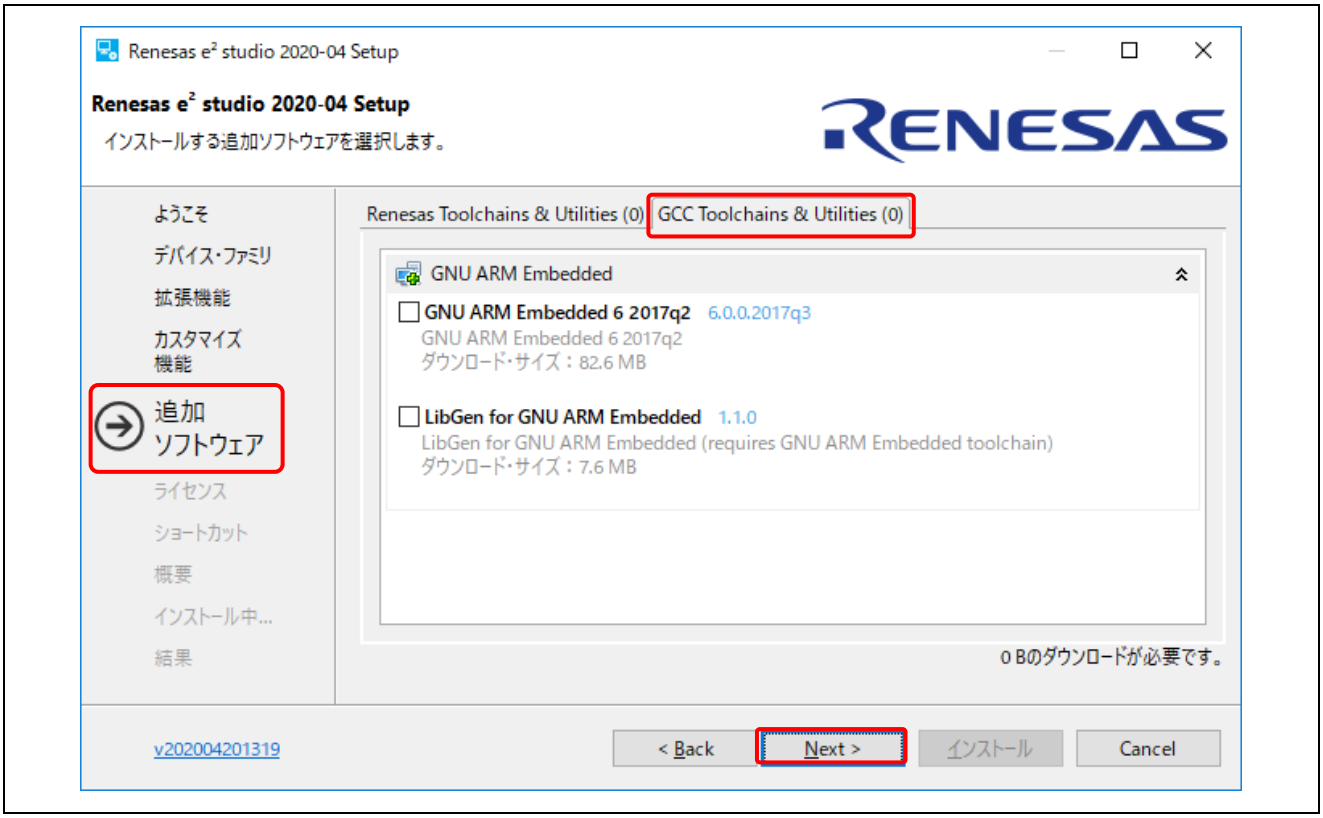

図 2-6　64 ビット版 e² studio のインストール：追加ソフトウェア

7. ライセンス

ライセンス契約を読んで同意した後、[次へ] をクリックします。

ライセンス契約に同意しない場合、インストールは続行できません。

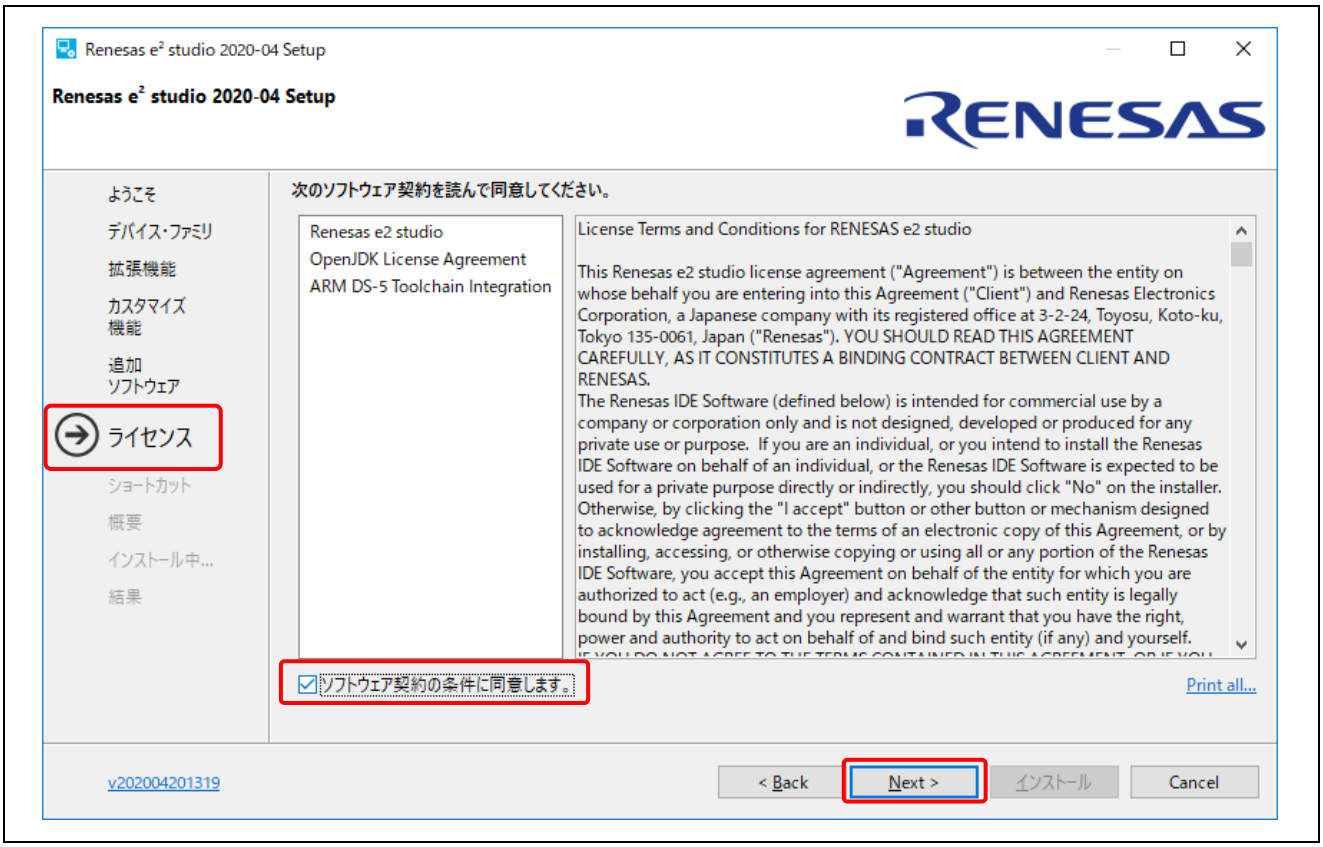

図 2-7　64 ビット版 e² studio のインストール：ライセンス

8. ショートカット

スタートメニューに表示するショートカット名を選択し、[次へ] をクリックします。

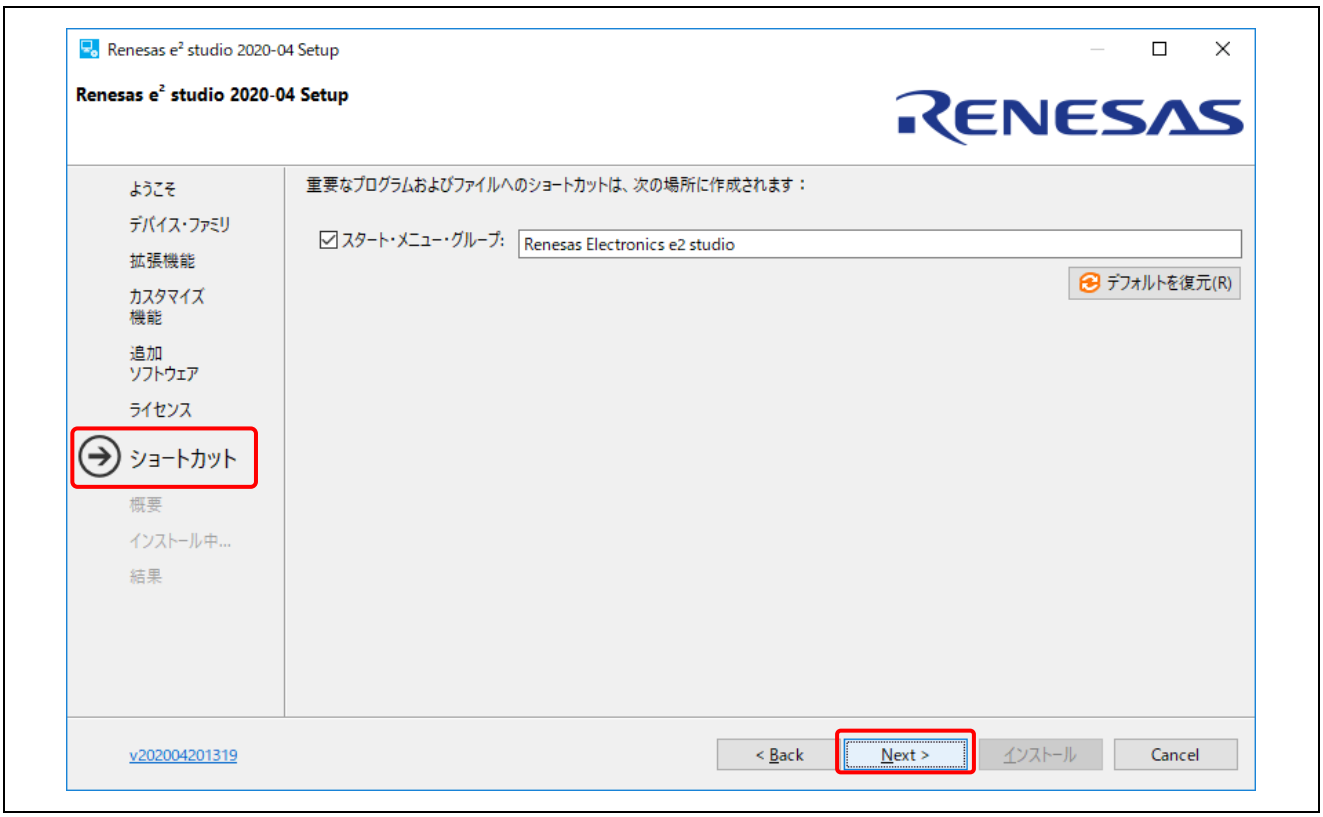

図 2-8 64 ビット版 e² studio のインストール：ショートカット

9. 概要

[インストール]をクリックして、e² studio をインストールしてください。

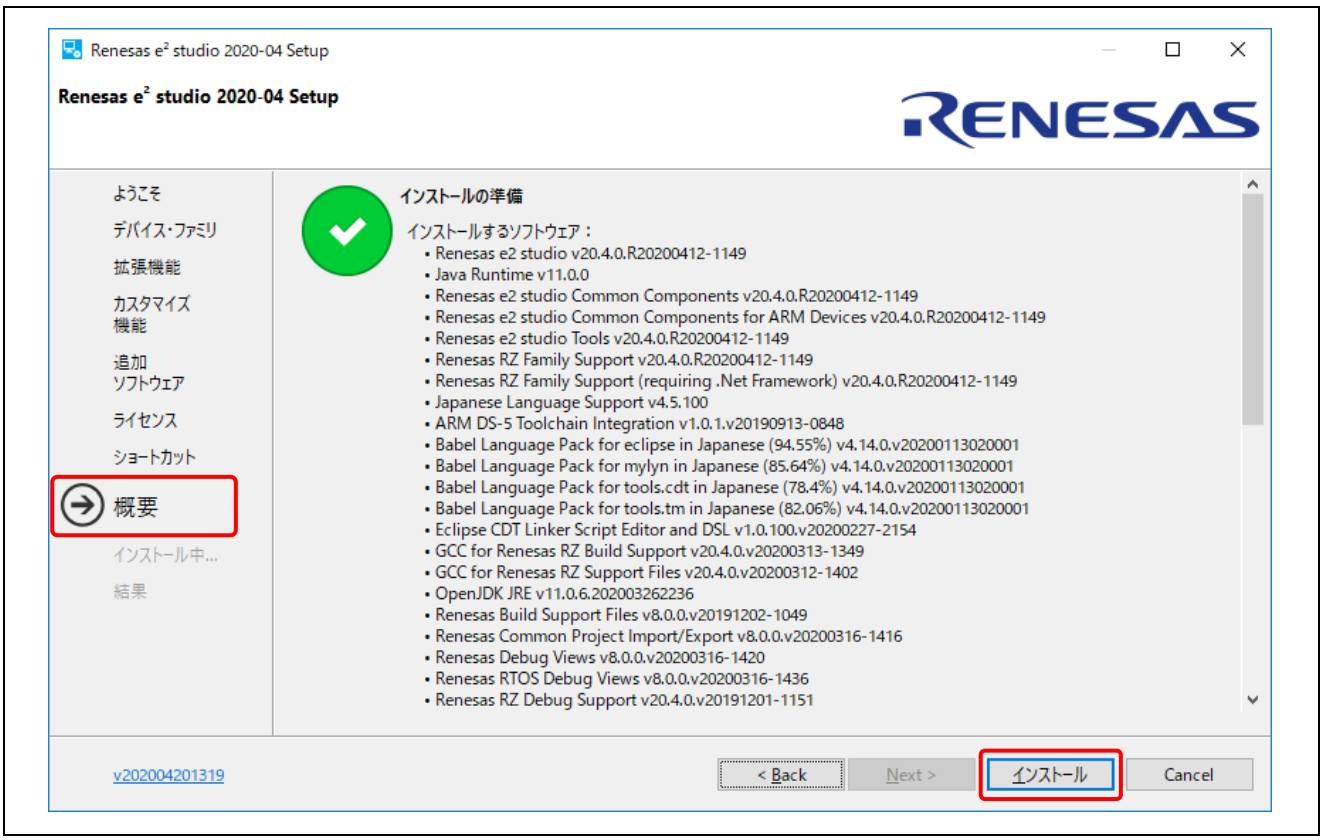

図 2-9　　64 ビット版 e² studio のインストール：概要

10. インストール中…

インストールの実行中は、追加ソフトウェアで選択した項目に応じて、ソフトウェアをインストールするためのダイアログボックスが開きますので、画面の指示に従って操作を行ってください。詳細は、2.3 節を参照してください。

11. 結果

インストールの結果が表示されます。エラーメッセージが無いか確認してください。
[OK]をクリックすると、インストールが終了します。

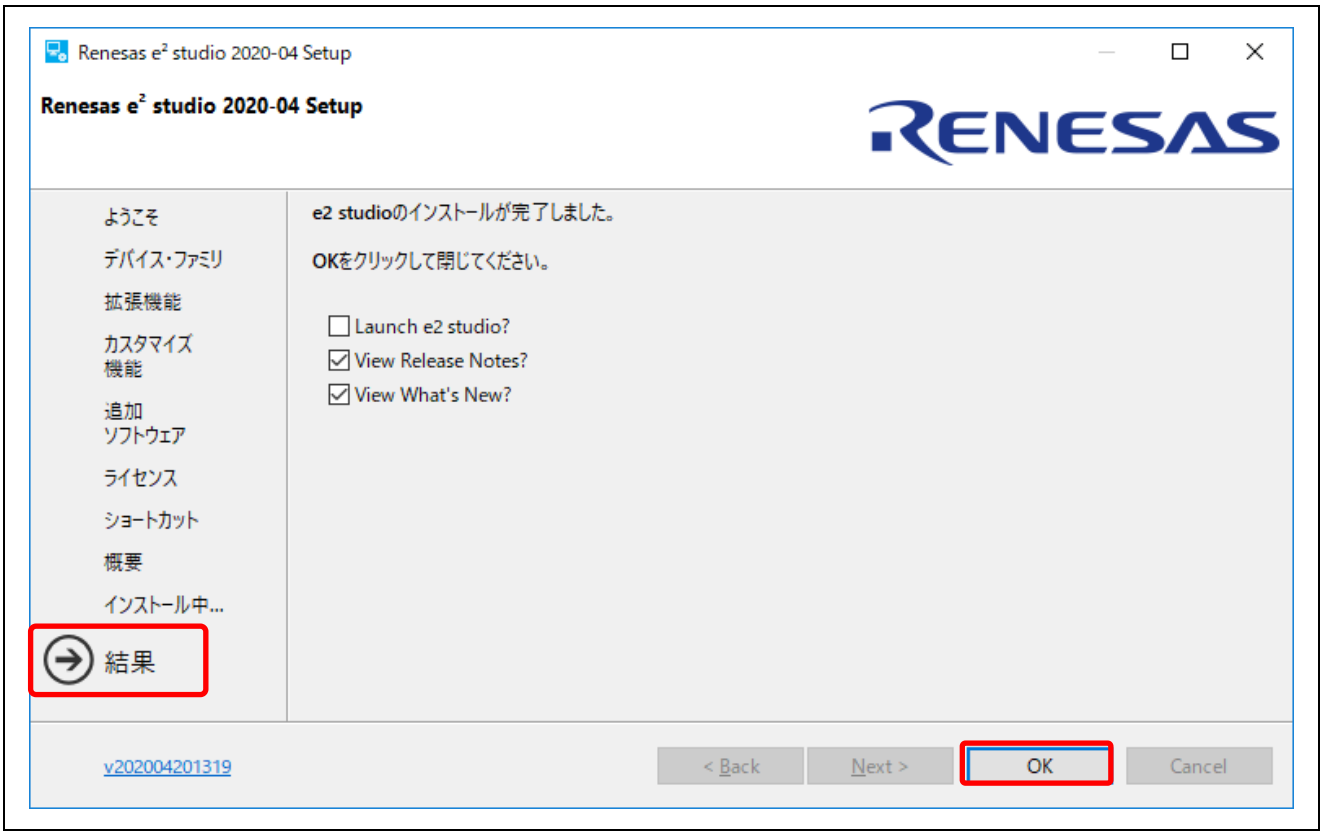

図 2-10　64 ビット版 e² studio のインストール：結果

2.2 e² studio のインストール（32 ビット版）

1. e² studioインストーラをダブルクリックしてe² studioインストールウィザードページを開いてください。
   [次へ(N)] をクリックします。

   注 1 ： e² studio が既に PC にインストールされた場合は図 2-1 の通りに変更、削除、インストールの選択肢が表示されます。複数のバージョンの e² studio インストールが必要な場合も、[インストール]をクリックしてインストール先を指定してください。

図 2-11 32 ビット版 e2 studio のインストールウィザード

2. ようこそページ
   [次へ(N)] をクリックします。

図 2-12 32 ビット版 e² studio のインストール：ようこそ

3. インストール・フォルダー

デフォルトのインストール先は “C:¥Renesas¥e2_studio” に設定されています。変更する場合は、インストールするフォルダをテキストボックスに直接入力するか [参照...] ボタンをクリックして指定してください。[次へ(N)] をクリックします。

注 1： 複数のバージョンの e² studio インストールが必要な場合も、[インストール]をクリックしてインストール先を指定してください。

注 2： インストール先には英文字、数字、アンダーバーのみを含むフォルダパスを指定してください。
また、Windows のアカウント名の中に英文字、数字、アンダーバー以外の文字がある場合、e² studio の動作に問題がある可能性がありますので、Windows のアカウント名にも英文字、数字、アンダーバーだけ使用することをお勧めします。（詳細は e² studio V2020-04 以降のダウンロードページに掲載している注意事項をご参照ください。）

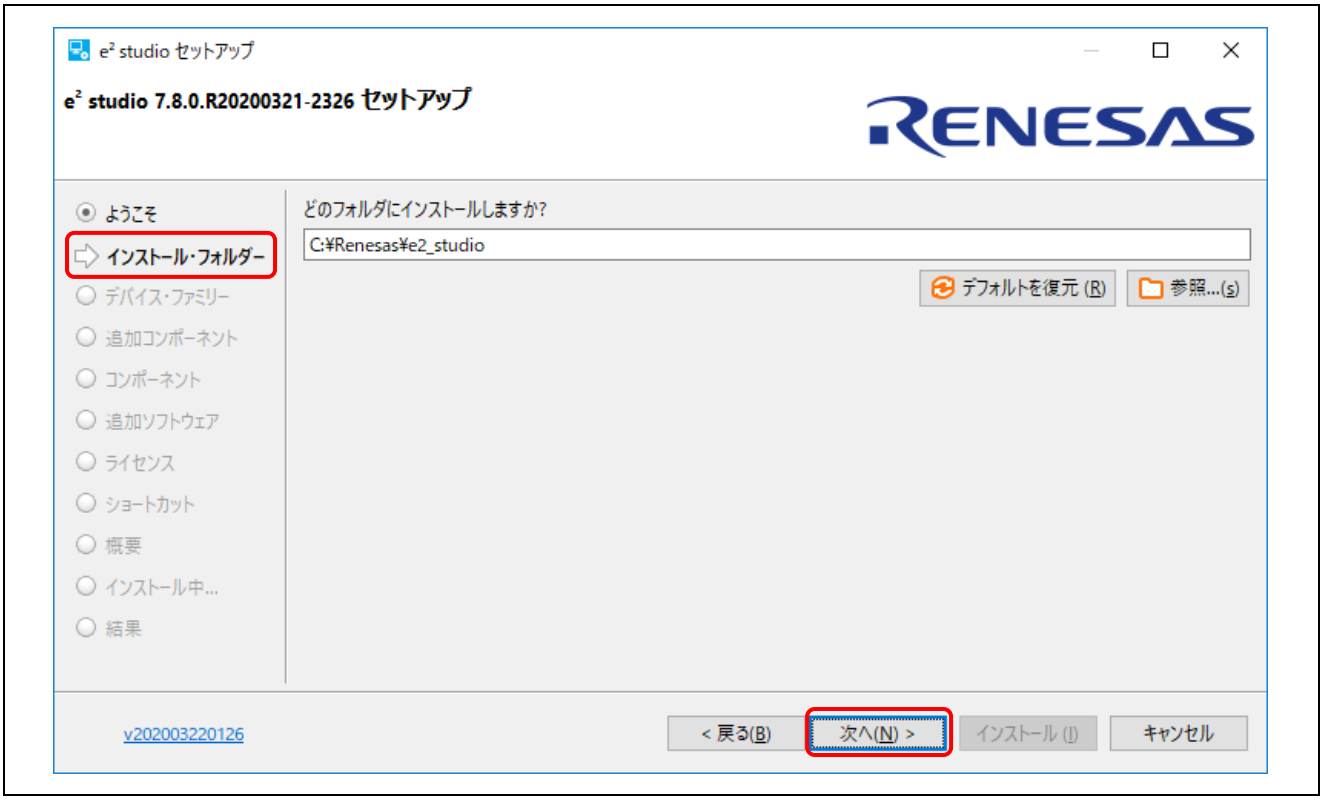

図 2-13 32 ビット版 e² studio のインストール：インストール・フォルダー

4. デバイス・ファミリー

インストールするデバイス・ファミリーを選択できます(複数選択可)。RZファミリのソフトウェアを開発するために[RZデバイス・サポート]チェックボックスを選択してください。

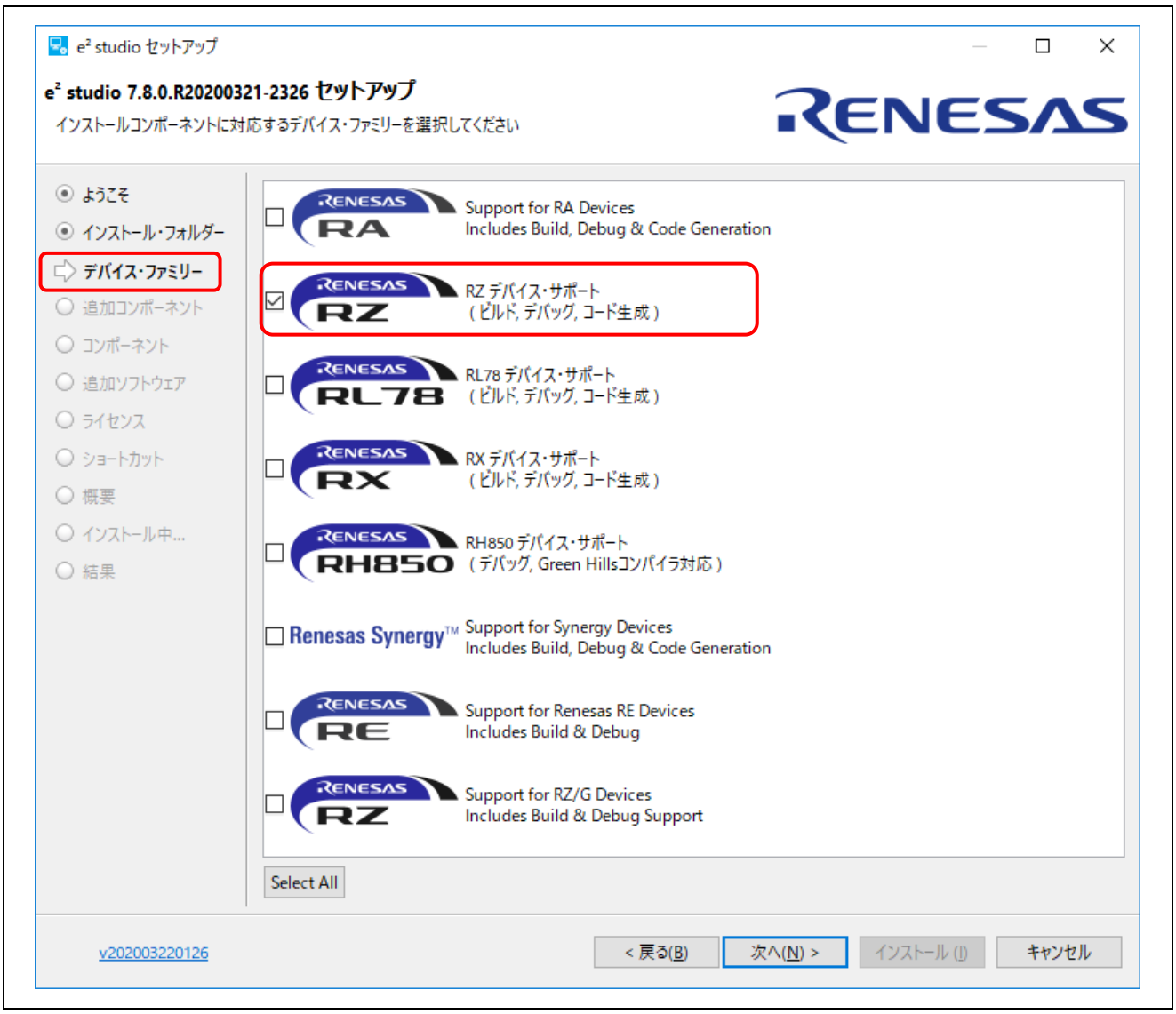

図 2-14 デバイス・ファミリー（32 ビット版 e² studio）：RZ デバイス・サポート

注意： [Support for RZ/G Devices]は、RZ/G Linux アプリケーションプロジェクトを生成、デバッグするためのオプションです。（e² studio 32 ビット版のみ選択可能）RZ/G デバイス上で Linux 用アプリケーションを開発する場合は、こちらも選択してください。

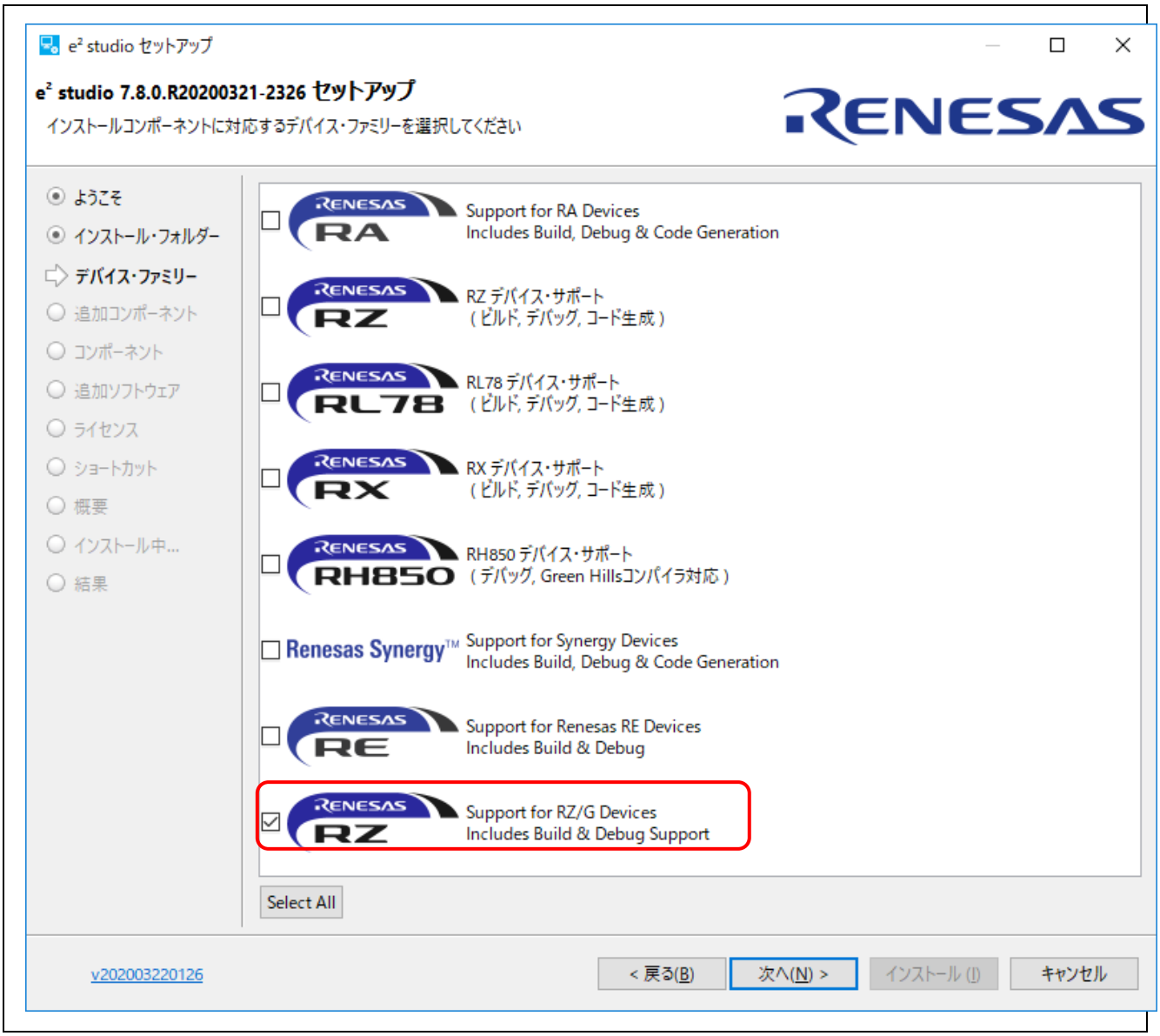

図 2-15 デバイス・ファミリー（32 ビット版 e² studio）：Support for RZ/G Devices

5. 追加コンポーネント

インストールする追加コンポーネント（言語パック、SVN および Git サポート、RTOS サポート）を選択してください。

FreeRTOS、OpenRTOS などの RTOS を使用している場合は、[RTOS]を選択してください。

[次へ] をクリックします。

注意： 日本語の言語パックを指定しないとメニュー等が日本語表示になりません。

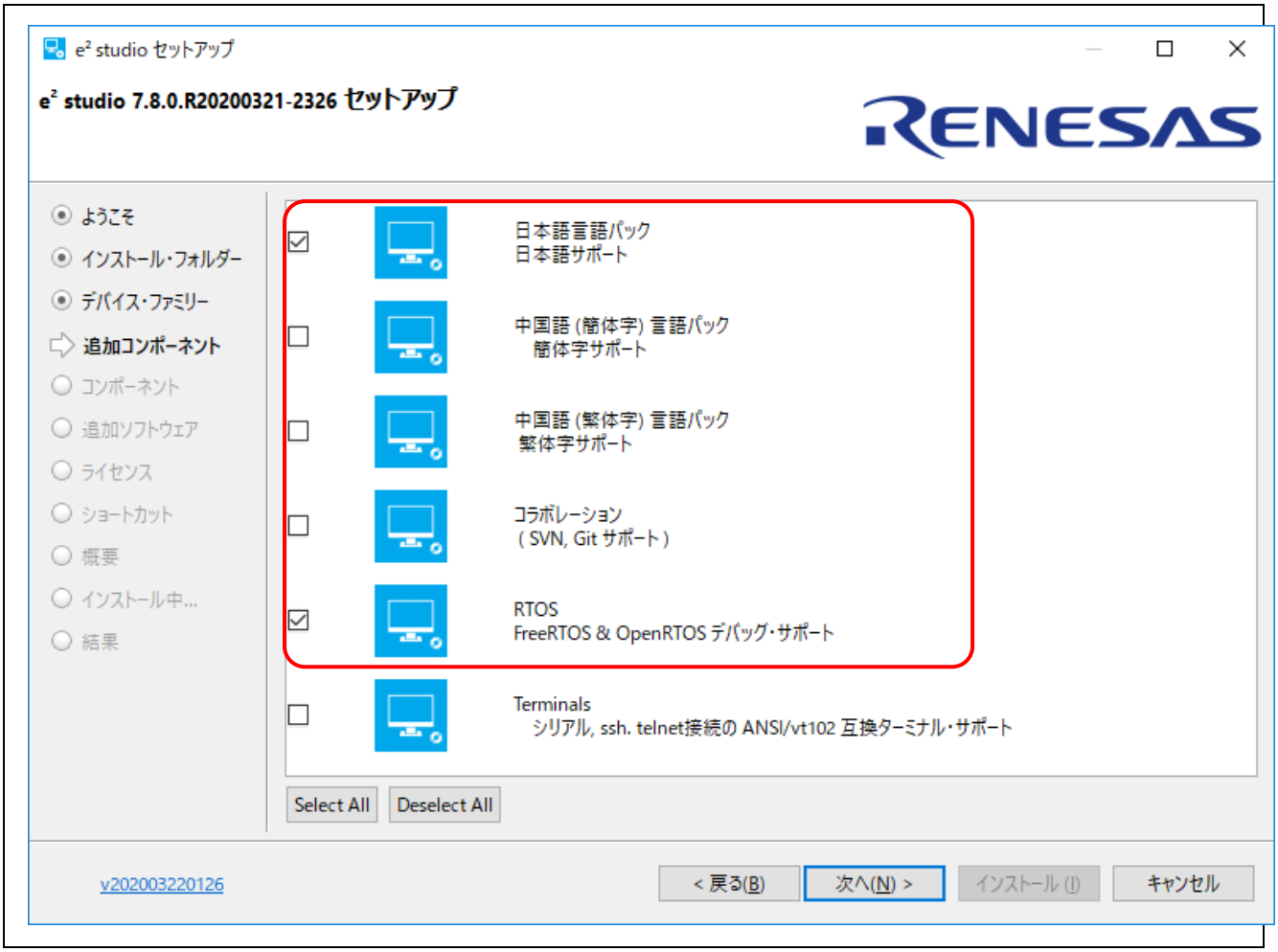

図 2-16 追加コンポーネント（32 ビット版 e² studio）：RTOS の選択

6. コンポーネント
   [デバイス・ファミリー] で選択した必要なコンポーネントは、すべて自動的に選択されます。
   選択したコンポーネントを確認し、[次へ] をクリックします。

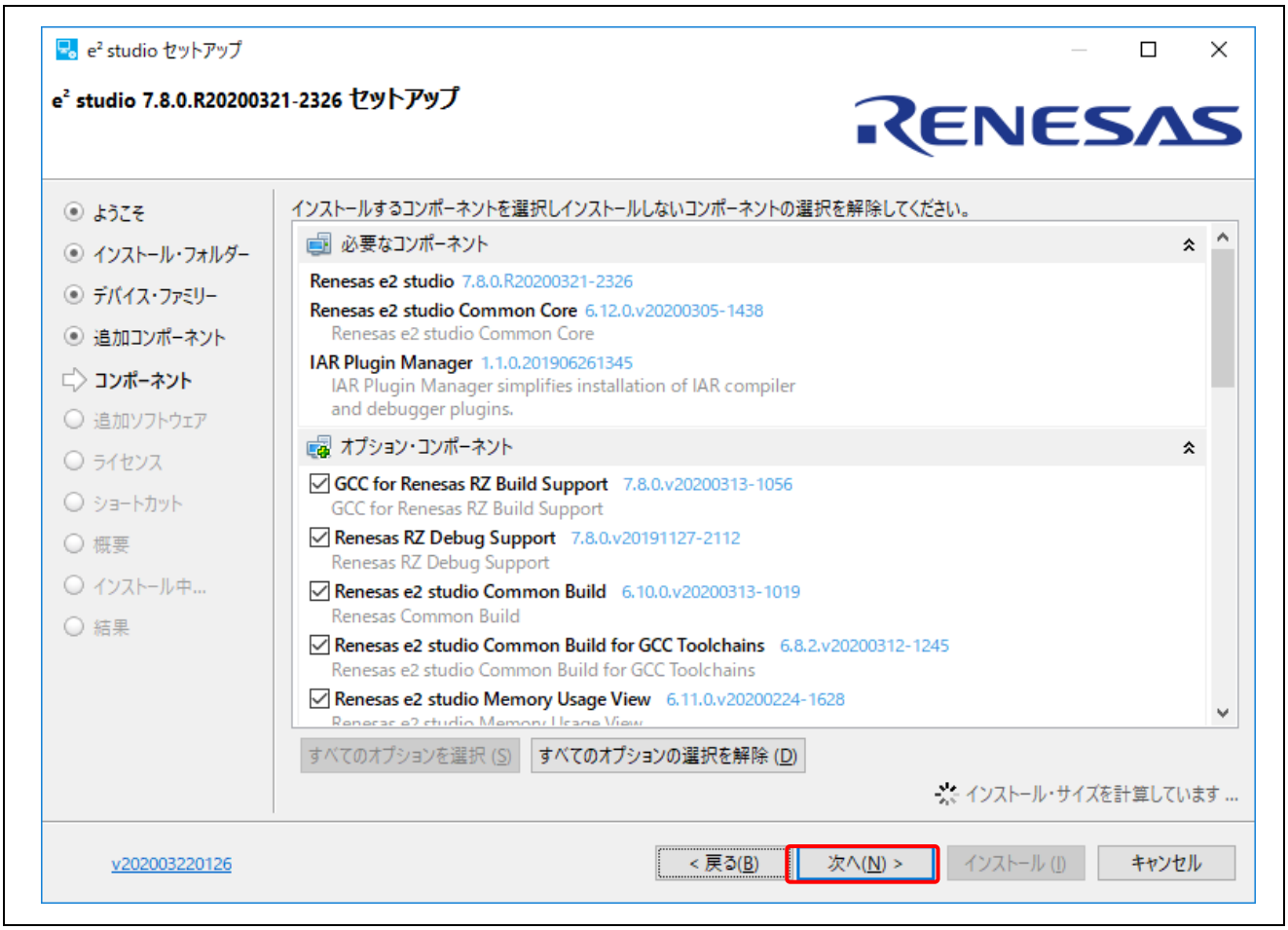

図 2-17　コンポーネント（32 ビット版 e² studio）

7. 追加ソフトウェア

GCC ARM Embedded（GNU ARM Embedded toolchain と同様）、LibGen for GCC ARM Embedded などの追加ソフトウェアが選択されていることを確認してください。（自動的に選択された追加ソフトウェアの一覧は、選択したデバイス・ファミリーによって決定します。）選択したソフトウェアのインストーラは、e² studio インストーラに呼び出されます。（詳細は、2.3 節を参照してください。）一覧に含まれない、GCC ARM Embedded の他のバージョンを使用したい場合は、別途インストールしてください（1.3 節を参照）。[次へ] をクリックします。

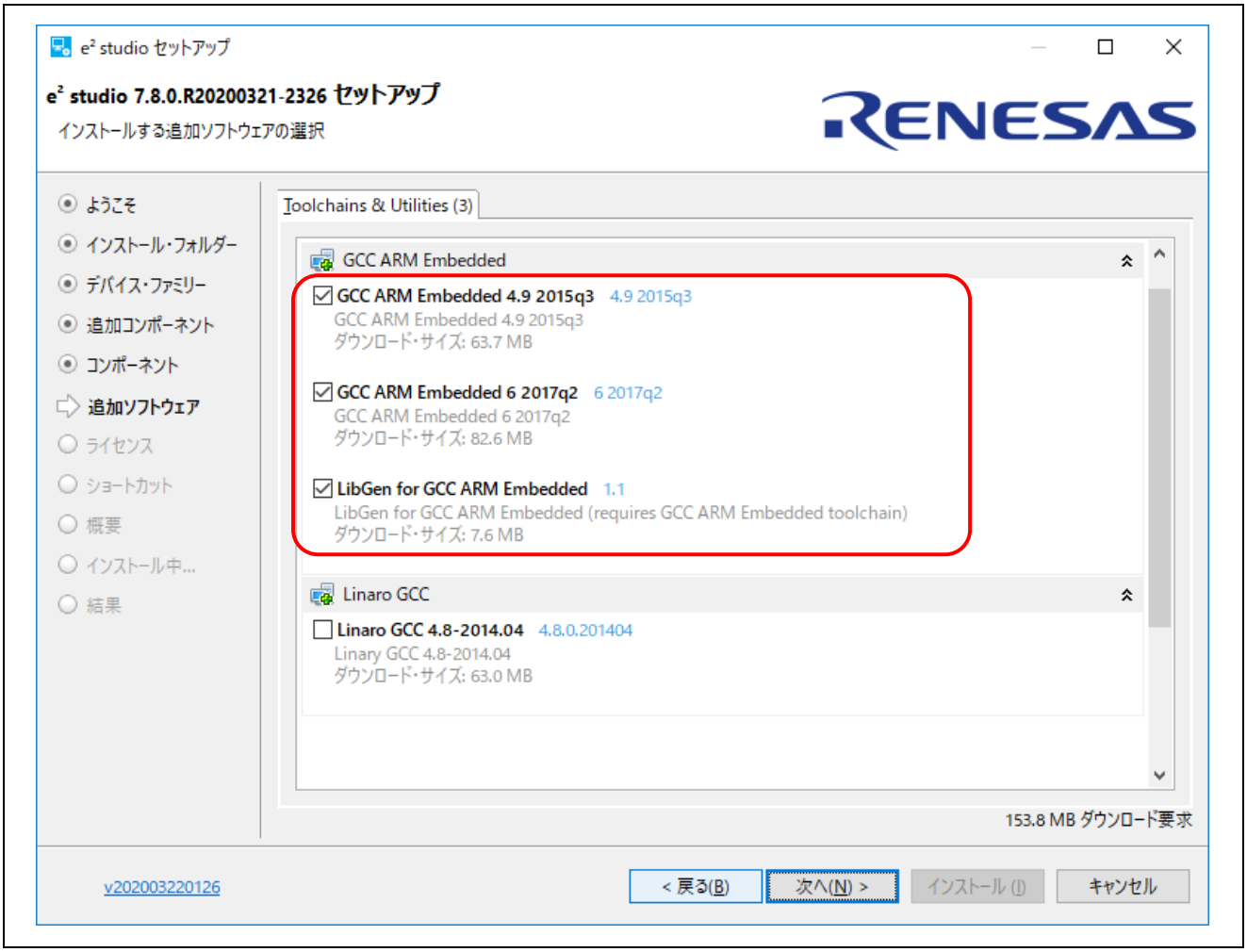

図 2-18 追加ソフトウェア（32 ビット版 e² studio）

8. ライセンス

ライセンス契約を読んで同意した後、[次へ] をクリックします。

ライセンス契約に同意しない場合、インストールは続行できません。

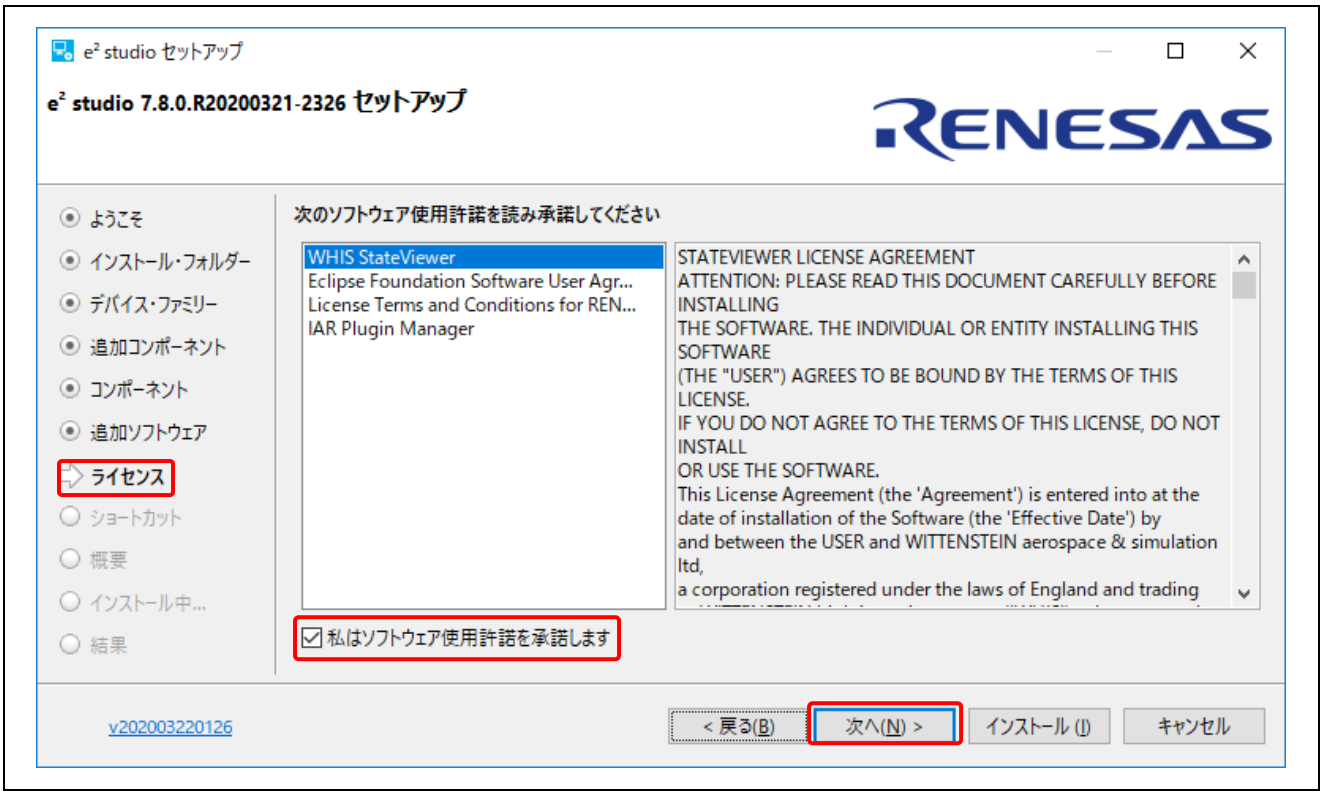

図 2-19 ライセンス（32 ビット版 e² studio）

9. ショートカット

スタートメニューに表示するショートカット名を選択し、[次へ] をクリックします。

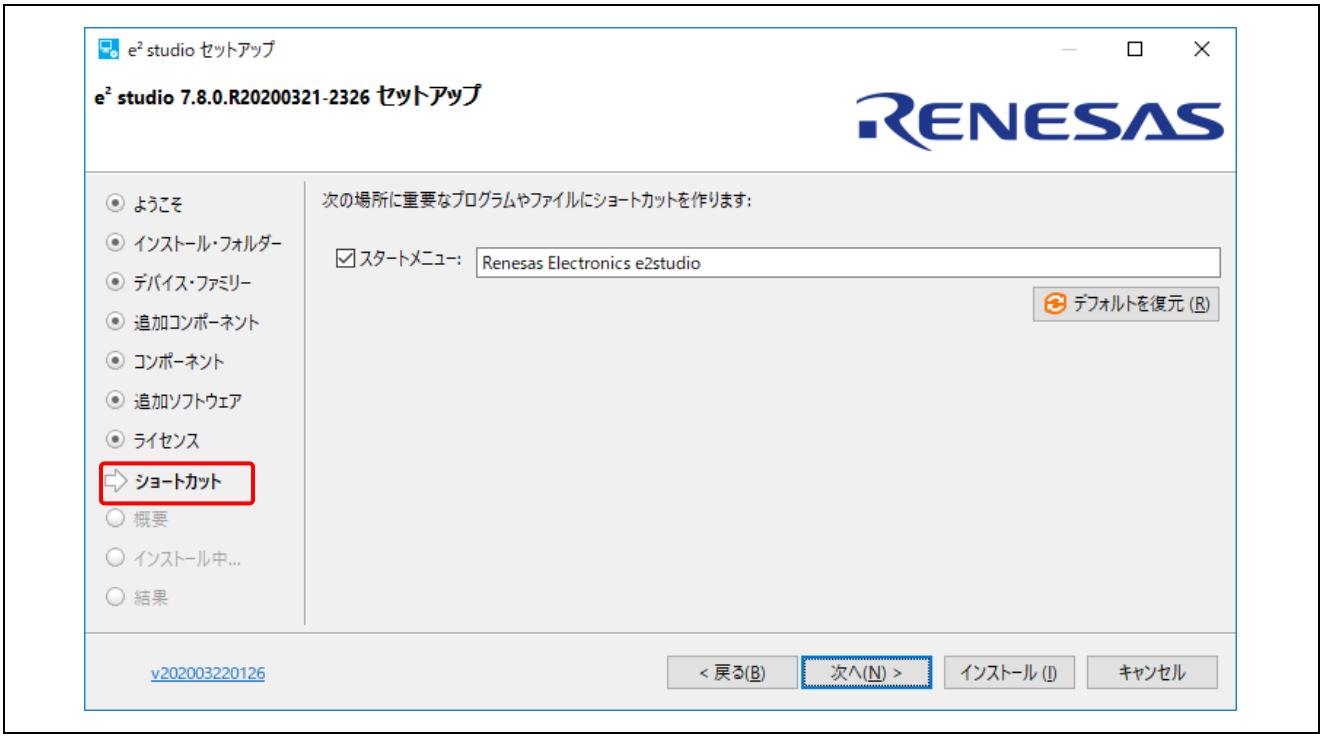

図 2-20 ショートカット（32 ビット版 e² studio）

10. 概要

[インストール]をクリックして、e² studio をインストールしてください。

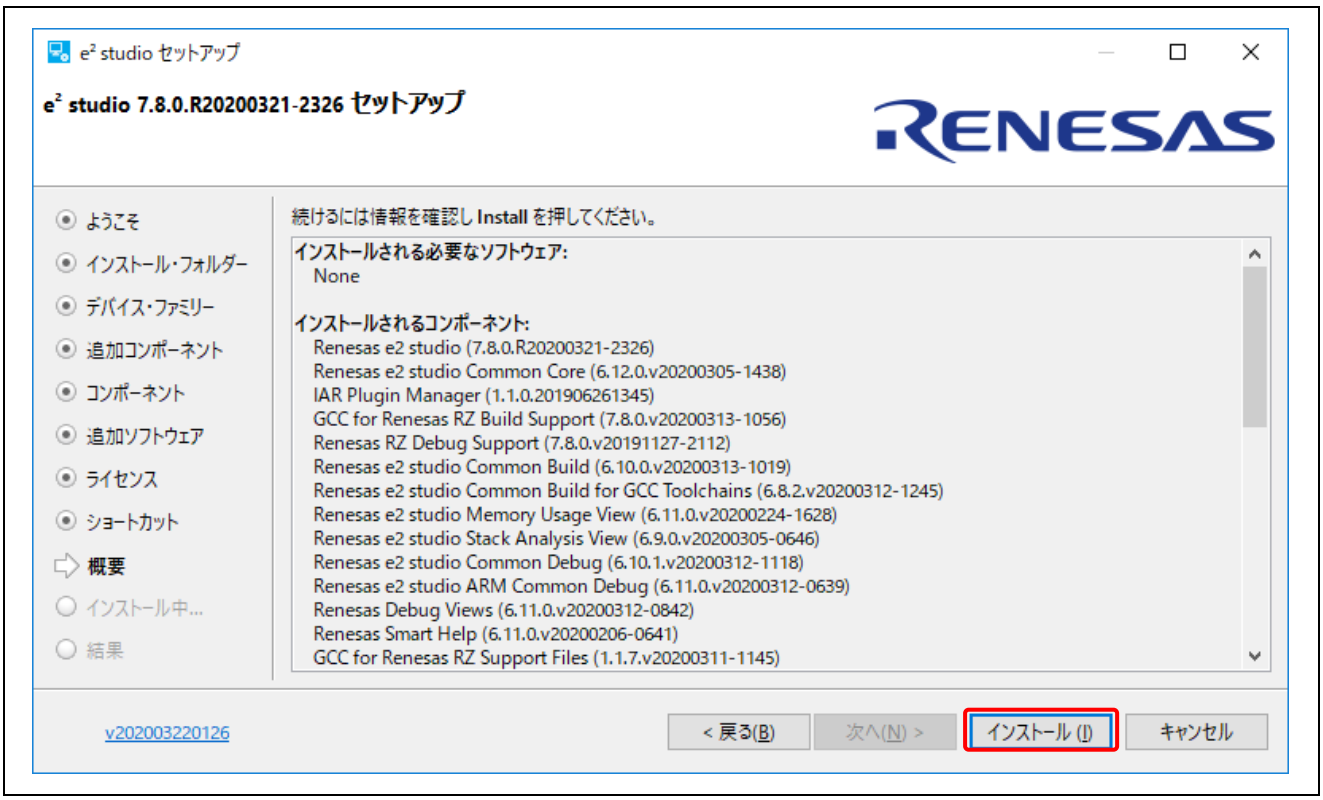

図 2-21　概要（32 ビット版 e² studio）

11. インストール中…

インストールの実行中は、追加ソフトウェアで選択した項目に応じて、ソフトウェアをインストールするためのダイアログボックスが開きますので、画面の指示に従って操作を行ってください。詳細は、2.3 節を参照してください。

12. 結果

[OK]をクリックすると、インストールが終了します。

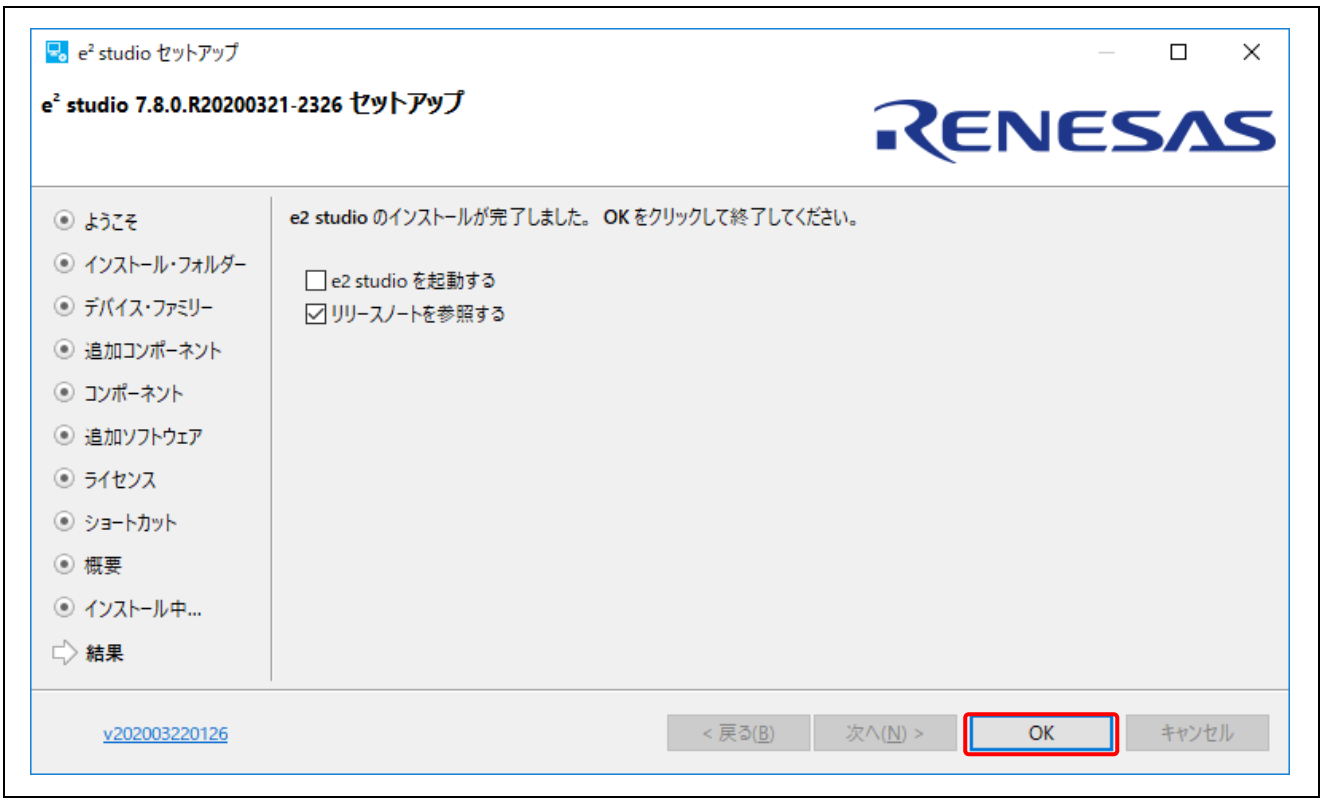

図 2-22　結果（32 ビット版 e² studio）

2.3 追加ソフトウェアのインストール

e² studio インストーラは、[追加ソフトウェア]で選択された GNU Compilers、Libgen Update for GNU ARM Embedded Toolchains などの他のインストーラを呼び出します。

2.3.1 GNU ARM Embedded toolchain

インストール中、e² studio インストーラの[追加ソフトウェア]で GNU ARM Embedded Toolchain が選択されている場合、このインストールウィザードが表示されます。

1. [次へ]をクリックします。
2. ライセンス契約を読んだ後、[同意する]をクリックします。
3. [インストール先フォルダ]にパスを指定してください。デフォルトのフォルダを推奨します。
4. インストールが終わると、Windows環境変数のパスや、レジストリ情報を追加するオプションを選択することができます。

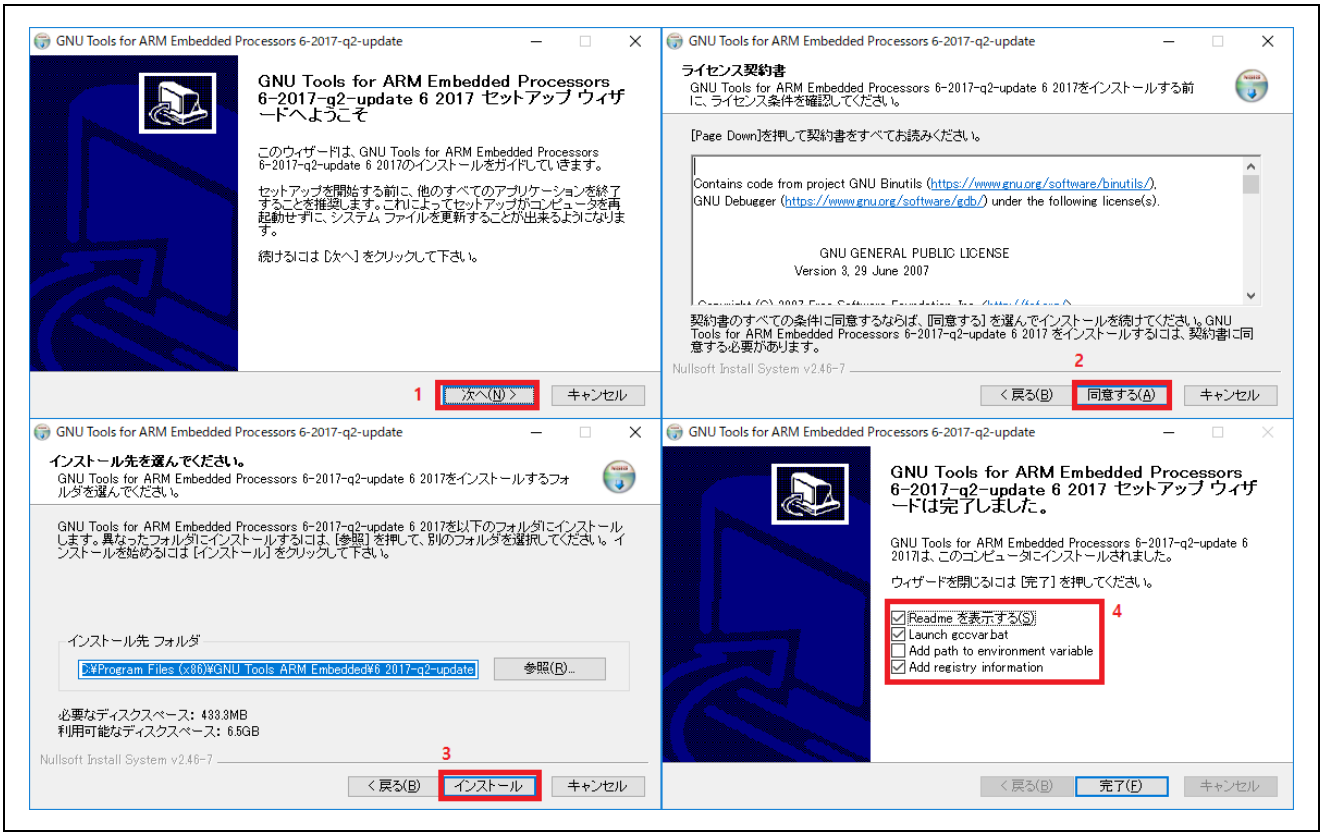

図 2-23 GNU ARM Embedded Toolchain のインストール

2.3.2 Libgen Update for GNU ARM Embedded Toolchains

インストール中、e² studio インストールの[追加ソフトウェア]で Libgen Update for GNU ARM Embedded Toolchain が選択されている場合、このインストールウィザードが表示されます。GNU ARM Embedded Toolchain を更新しない場合は、[Close] (×)をクリックして、Libgen Update for GNU ARM Embedded Toolchains を終了します。

1. [Click here to select your GNU ARM Embedded Toolchain installation folder]をクリックします。
2. 更新する[GNU ARM Embedded Toolchain]のフォルダを選択します。

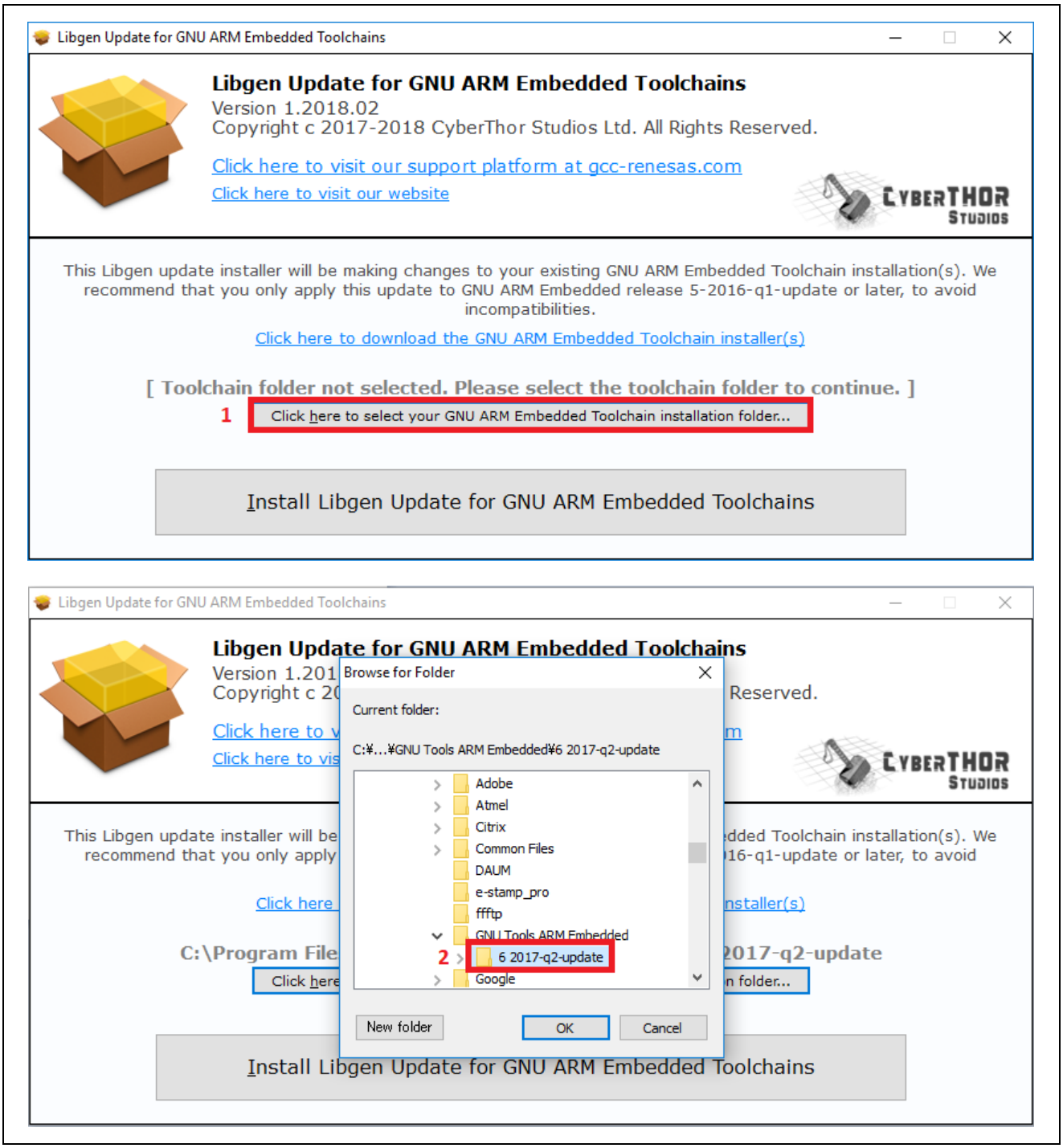

図 2-24 Libgen Update for GNU ARM Embedded：ツールチェーンフォルダの選択

3. [Install Libgen Update for GNU ARM Embedded Toolchains]をクリックします。

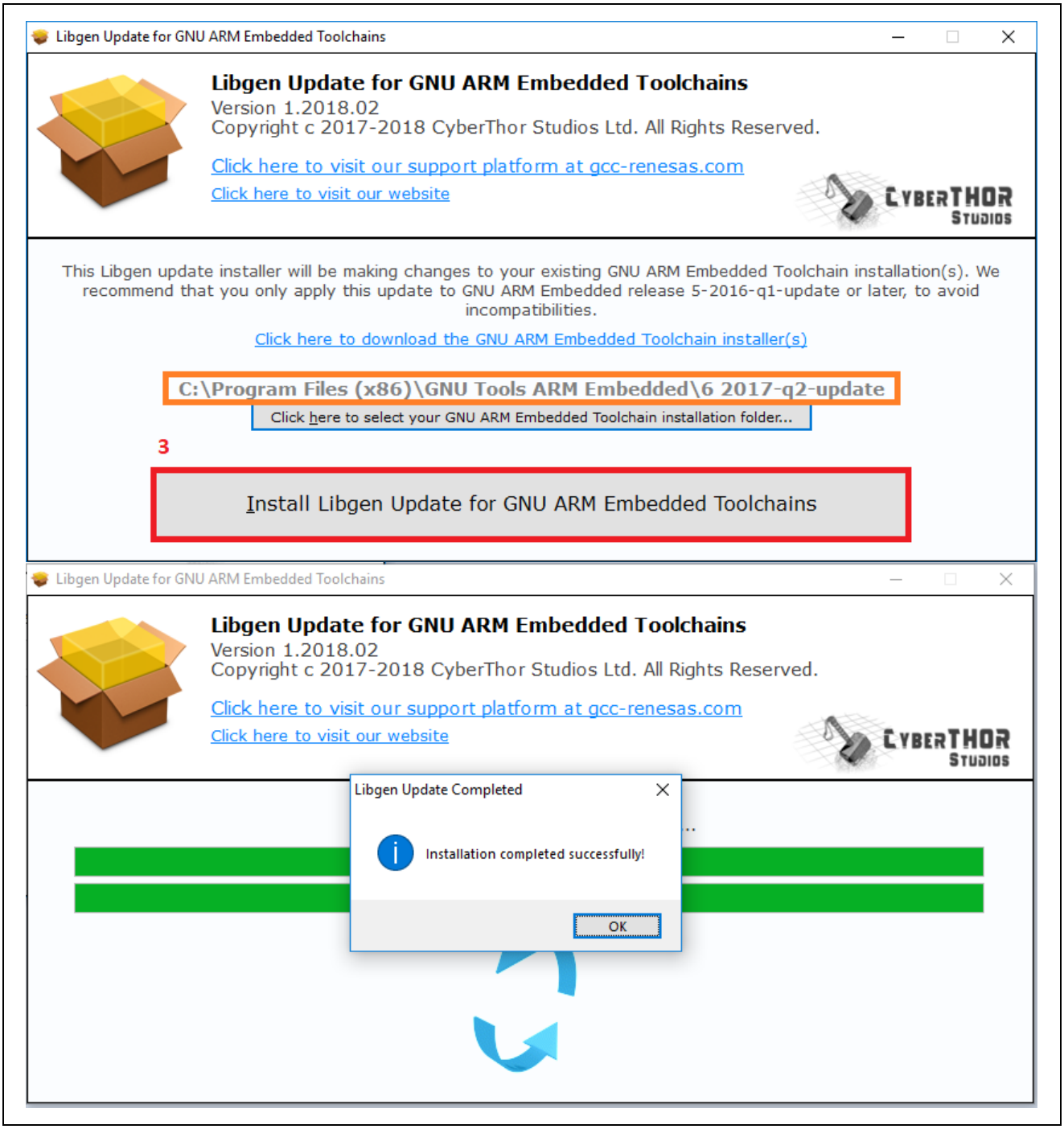

図 2-25 Libgen Update for GNU ARM Embedded：ツールチェーンの更新

他のツールチェーンを更新する場合は、[Libgen Update for GNU ARM Embedded]をダウンロードして実行してください（1.4 節を参照）。

2.4 e2 studio のアンインストール

e2 studio のアンインストールは、Windows OS での通常のプログラムアンインストール手順で行えます。

1. [スタート] → [コントロールパネル] → [プログラムと機能]を選択します。
2. インストール済みプログラムのリストから、"e2 studio" を選択し、[アンインストール(U)] ボタンをクリックします。
3. [アンインストール] ダイアログボックスの [アンインストール] ボタンをクリックして削除を確認してください。

アンインストールの最後に、e2 studio はインストール先から削除され、ショートカットメニューも削除されます。

2.5 e2 studio のメジャーバージョンアップ

メジャーバージョンアップ(v.6.3.0 から v.7.0.0 へのような大きな桁のバージョンアップ)では、[更新の検査]は適用できませんので、e2 studio インストーラにより新たにインストールを行ってください。その際、古いバージョンの e2 studio と同じフォルダに異なるバージョンの e2 studio を上書きインストールしないようご注意ください。元のバージョンの e2 studio を一旦アンインストールするか、もし古いバージョンを残したままで別のバージョンを利用したい場合は、新たに作成した別のフォルダにインストールしてください。

2.6 インターネットを介したプラグイン更新

e² studio は、マイナーバージョンアップ(v7.3.0 から v7.4.0 へのような小さな桁のバージョンアップ)をオンライン、オフラインの両方でサポートしています。インターネット接続の有無に関わらず、ソフトウェアコンポーネントを更新することができます。

2.6.1 プラグインの更新方法

本節では、オンラインマイナーバージョンアップを起動する手順を示します。

1. [ヘルプ] メニューから [更新の検査] をクリックして、[利用できる更新] パネルを表示してください。

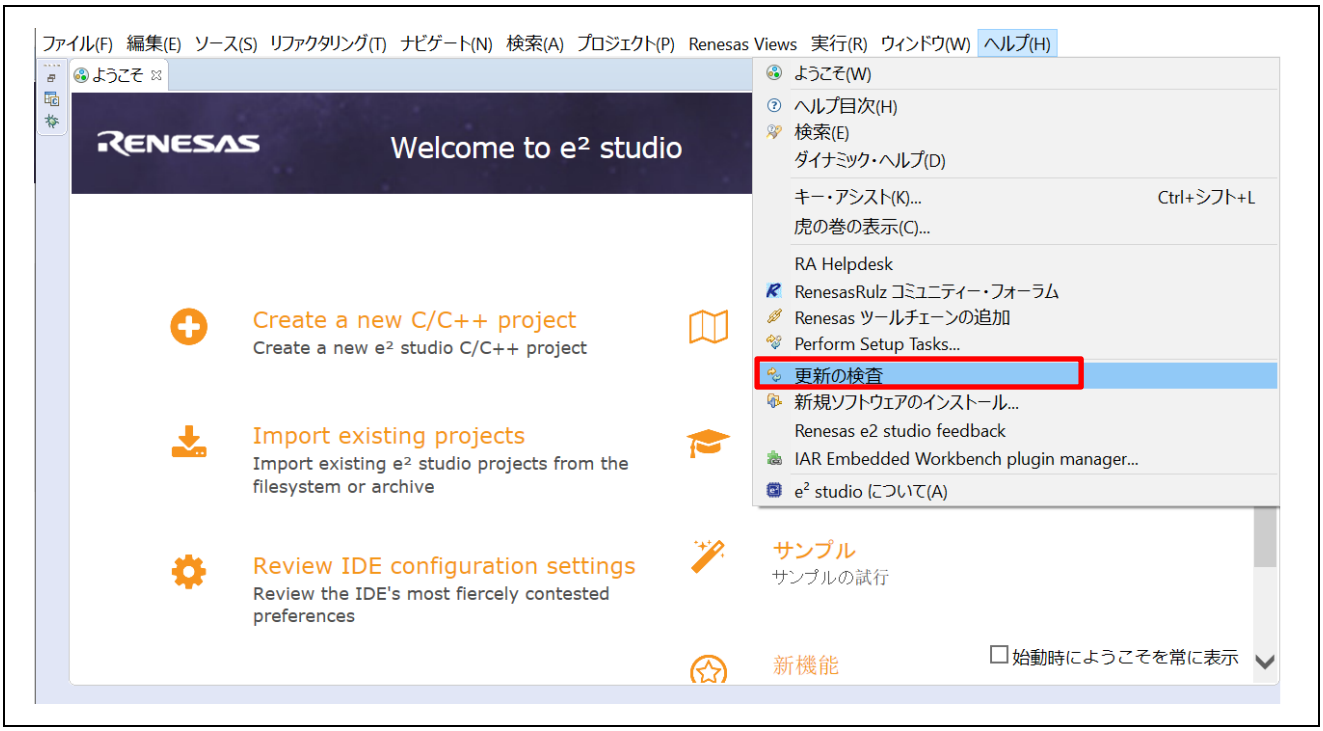

図 2-26 [更新の検査]メニュー

2. [利用できる更新] パネルでは、デフォルトですべてのソフトウェアコンポーネントが選択されています。これにより、すべてのソフトウェアを最新のものにアップデートすることができます。 (図2-2に例を示します。) [次へ(N)>] ボタンをクリックしてください。

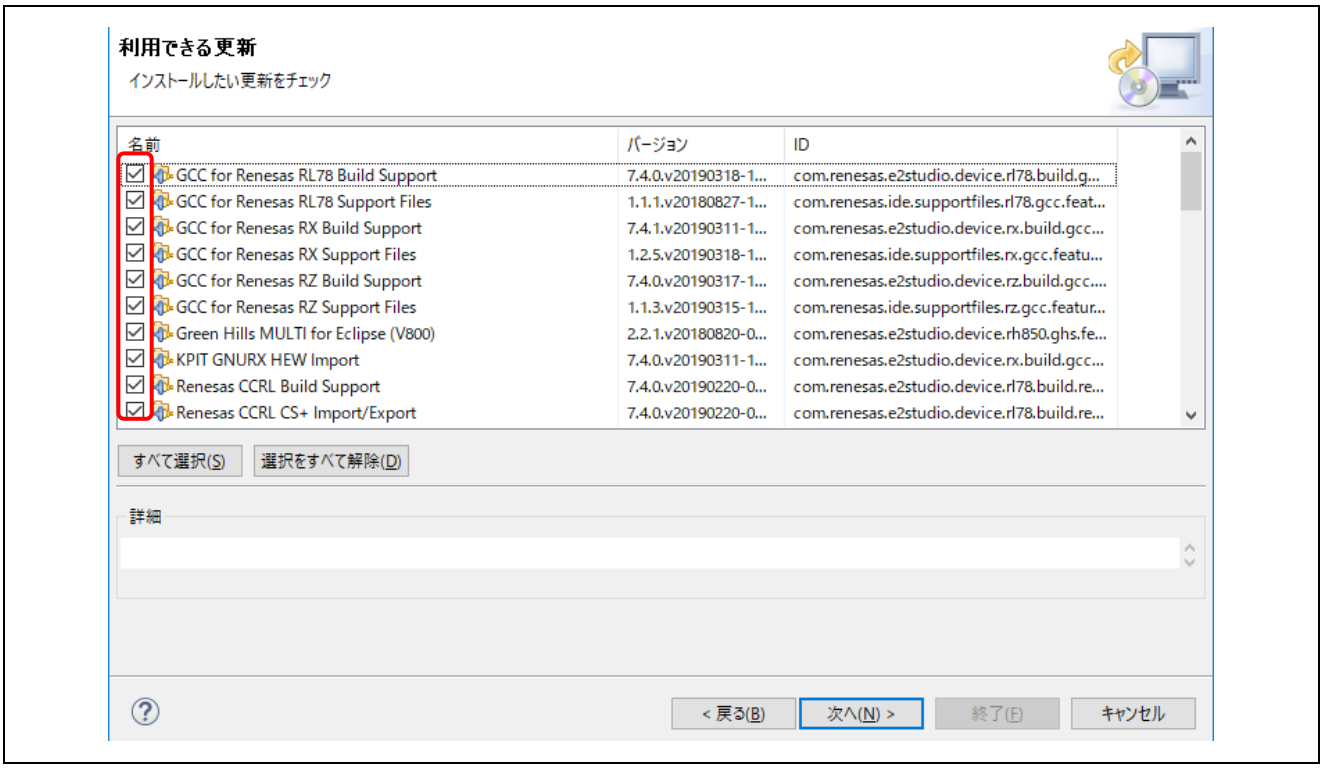

図 2-27 e² studio – [利用できる更新] パネル (1/3)

3. [次へ(N)>] ボタンを選択してアップデートを続行します。

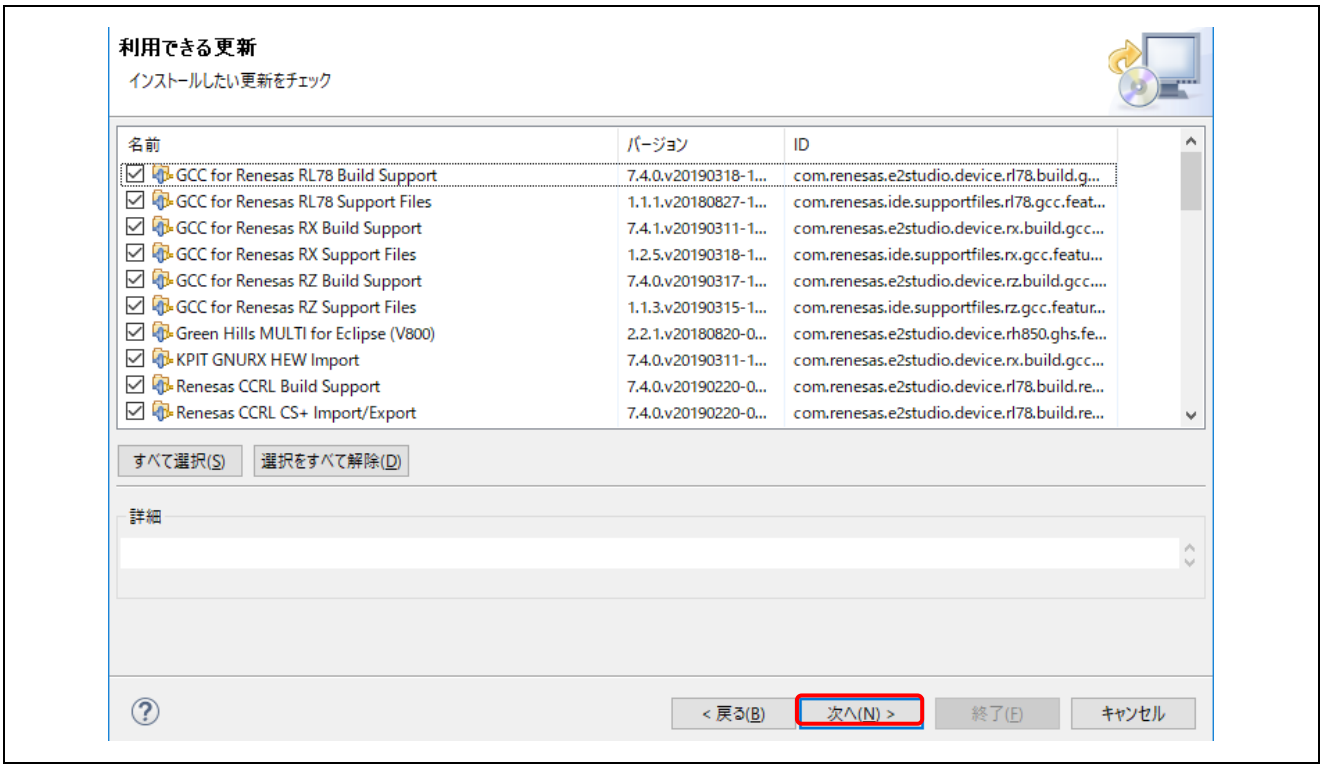

図 2-28 e² studio – [利用できる更新] パネル (2/3)

4. ソフトウェアライセンス契約を読み同意します。[終了(F)] をクリックしてアップデートを完了してください。

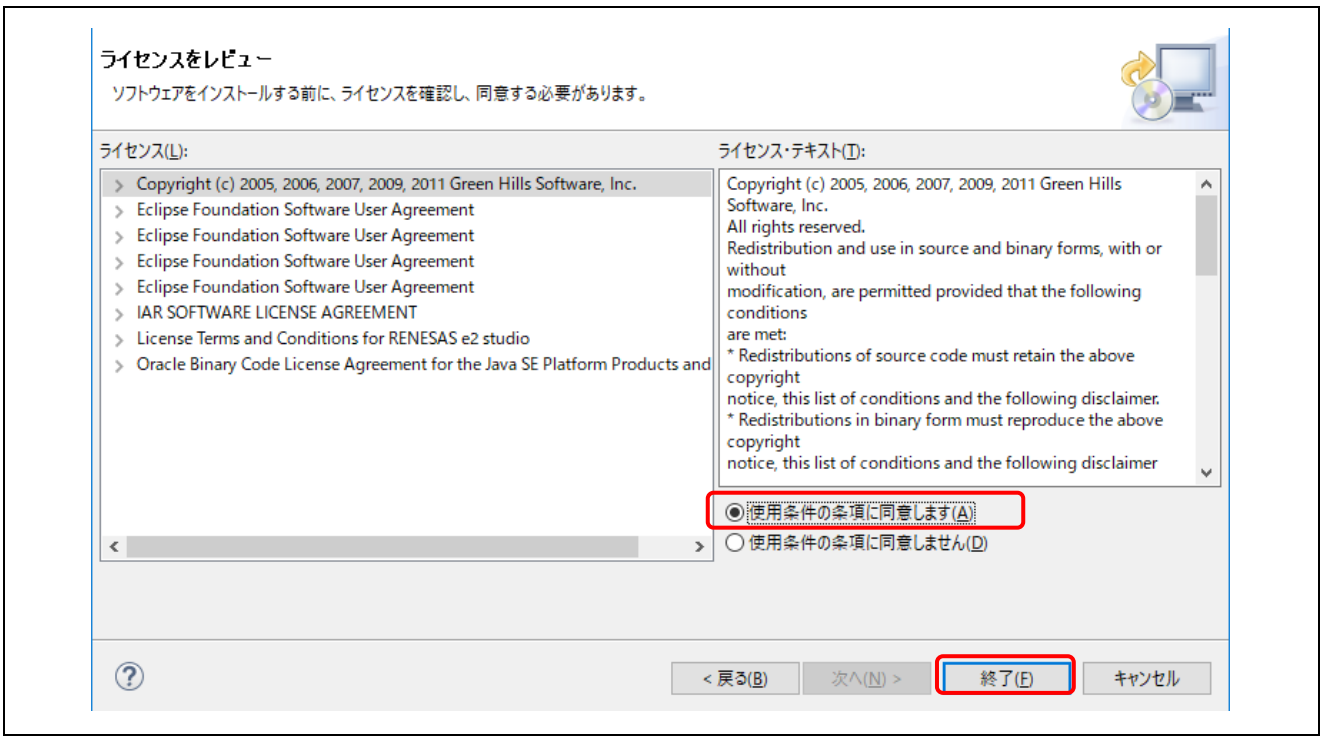

図 2-29 e² studio – [利用できる更新] パネル (3/3)

5. [ヘルプ] メニューの [e² studio について] をクリックし、アップデートされたバージョンを確認してください。

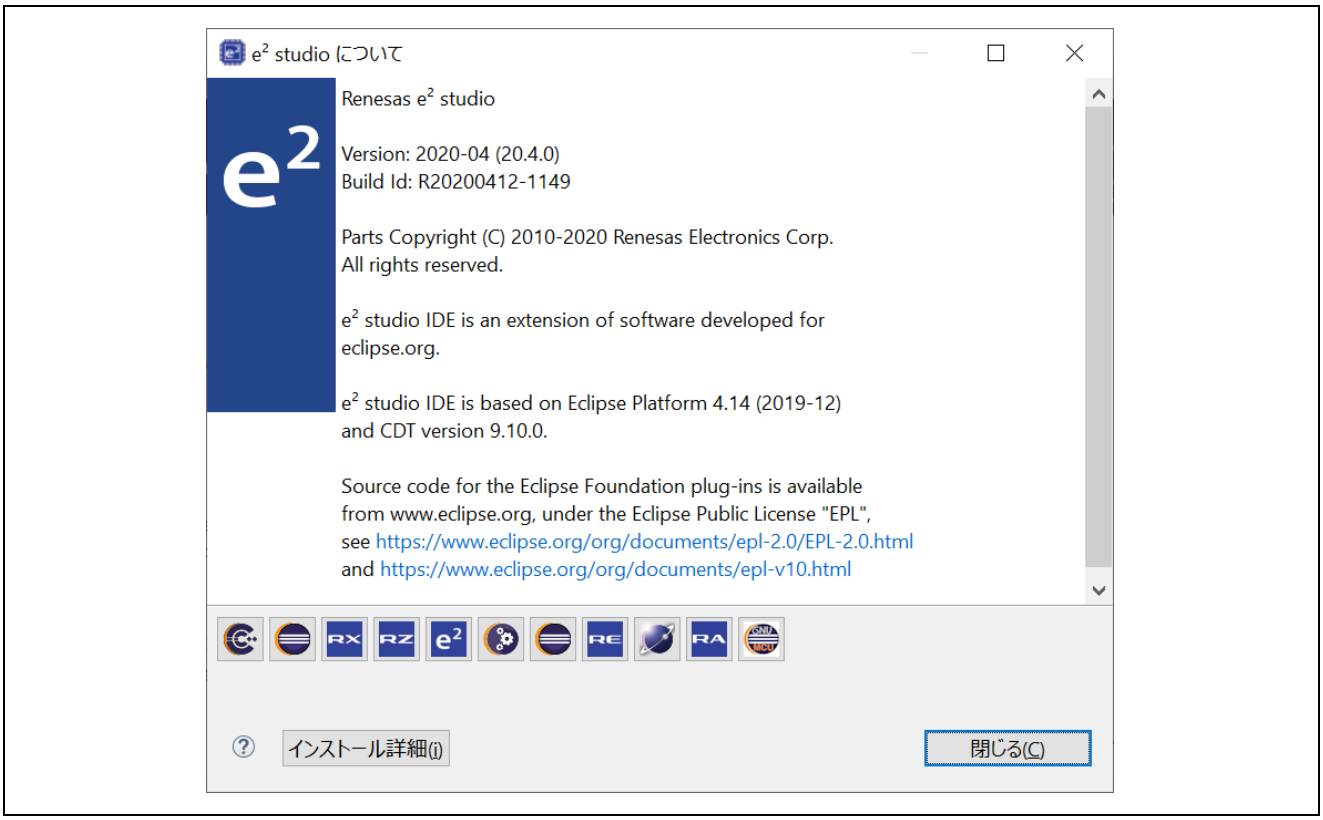

図 2-30 e² studio – [e² studio について] パネル

2.6.2 インストーラによるマイナーバージョンの更新方法

本節では、インストーラのバージョンアップ機能を用いて更新するための手順を説明します。

1. 下記のルネサスページからインストールしたいバージョンのe² studioインストーラ（オフライン用）をダウンロードしてください。http://www.renesas.com/e2studio_download

注意： 差分アップデートプログラムを使用した更新方法は、バージョン 3.X およびそれ以前のバージョンでのみ利用できます。

2. 1.でダウンロードしたインストーラファイルをダブルクリックして実行してください。インストーラがインストール済みのバージョンがあれば「インストール」以外の選択肢が現われます。
3. [アップグレード]、[次へ] をクリックして、更新を始めてください。

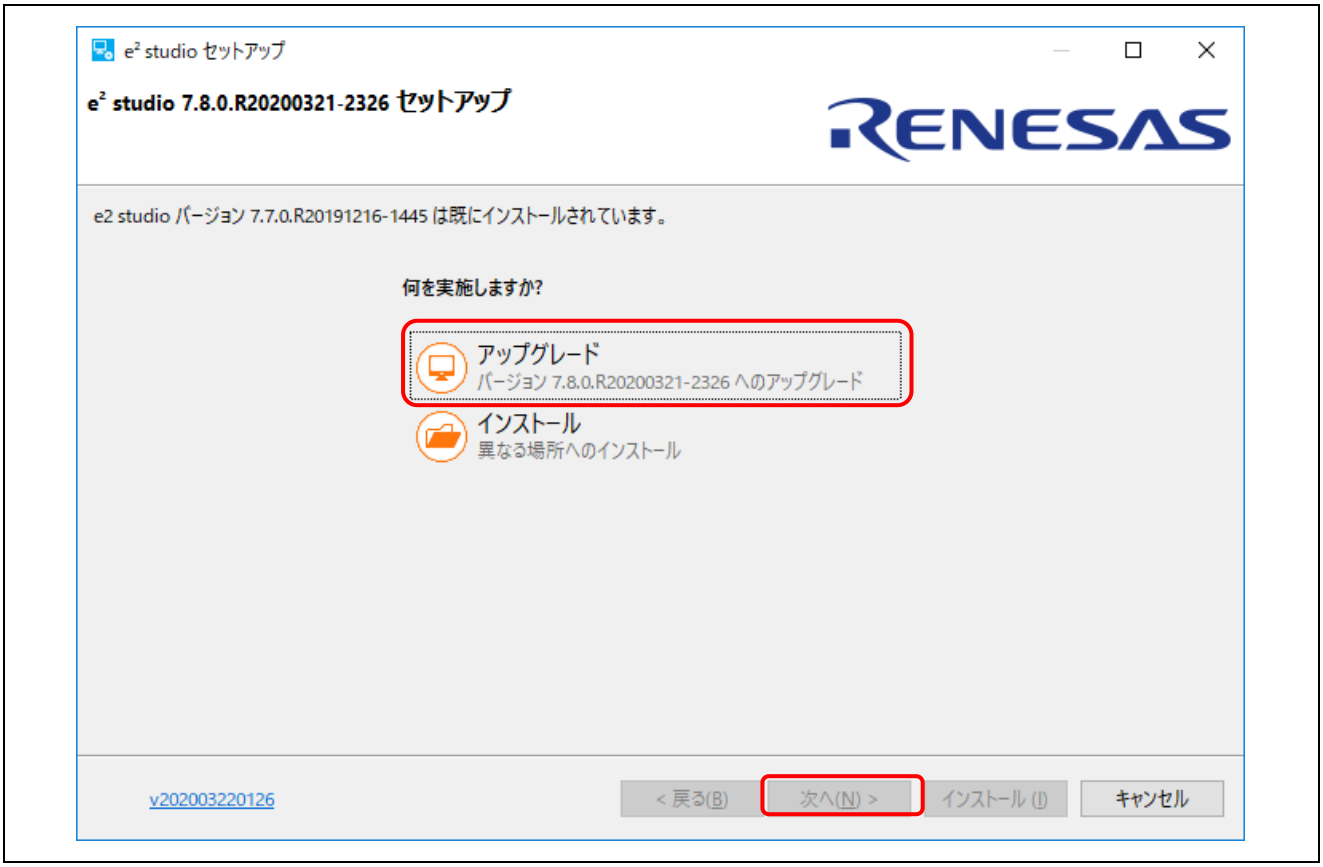

図 2-31 インストーラの起動画面 (インストール済のものがある場合)

4. 以降は 「2.1 e² studioのインストール」以下の手順を参考に、画面に従って操作を行ってください。アップグレードオプションでは、既存のe² studioと同じフォルダを使用するため、「3. インストール・フォルダー」の手順は省略します。

2.7 ツールチェーンへの e² studio の登録

インターネット接続時に e² studio をインストールする際、GCC ARM Embedded などのツールチェーンを e² studio インストーラで自動的にインストールすることができますが、インターネット接続のない環境で e² studio をインストールする際、ウェブサイトからダウンロードしたインストーラファイルを使用してツールチェーンを別途インストールし、e² studio に登録する必要があります。(RZ 用ツールチェーンのダウンロードサイトについては、1.3 節を参照してください。)

e² studio は、起動時に、インストールしたツールチェーンをスキャンします。ツールチェーンのインストール後に e² studio を実行する場合、ツールチェーンを選択するための Toolchain Integration ウィンドウが表示されます。

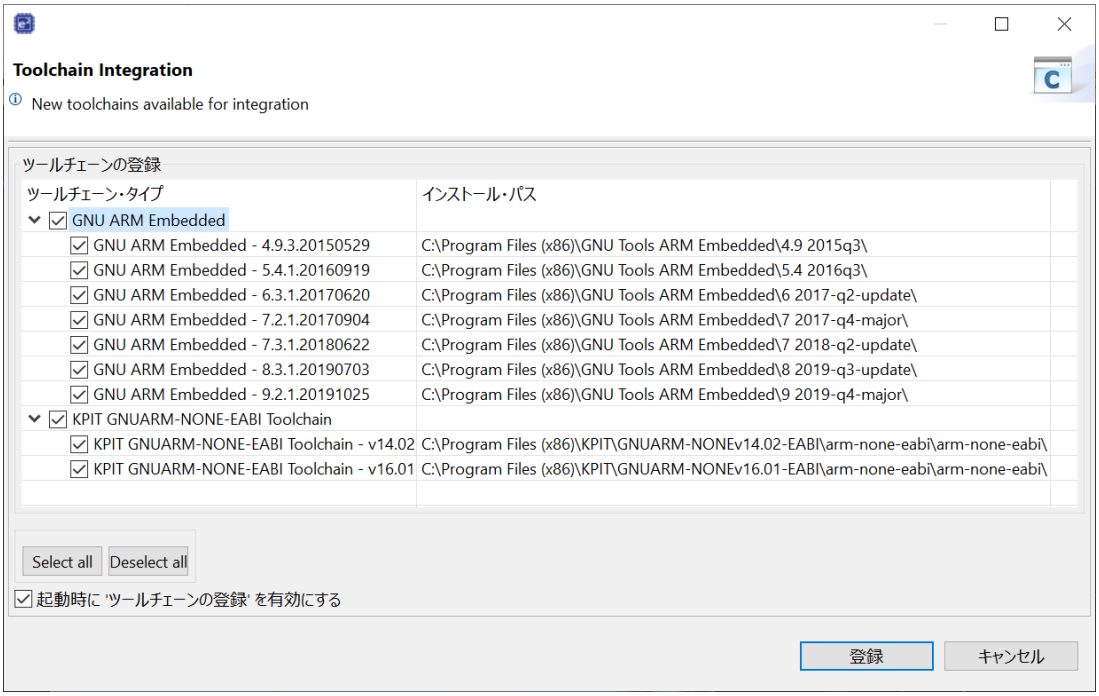

図 2-32 ツールチェーンの登録

インストール済みのツールチェーンを確認するには、ツールバーの[未登録ツールチェーンの登録]アイコンをクリックするか、[ヘルプ]メニューから[Renesas ツールチェーンの追加]をクリックして、[Renesas ツールチェーン管理] (下図)を開きます。e² studio に登録したいツールチェーンを確認してください。

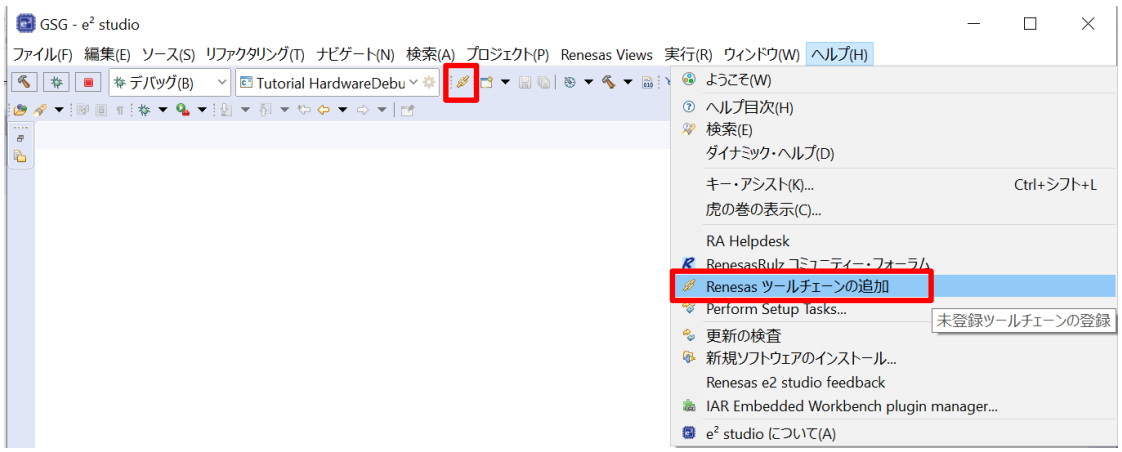

図 2-33 Renesas ツールチェーンの追加

使用したいツールチェーンがリストにない場合は、[追加] ボタンをクリックしてインストールされている場所を指定してください。

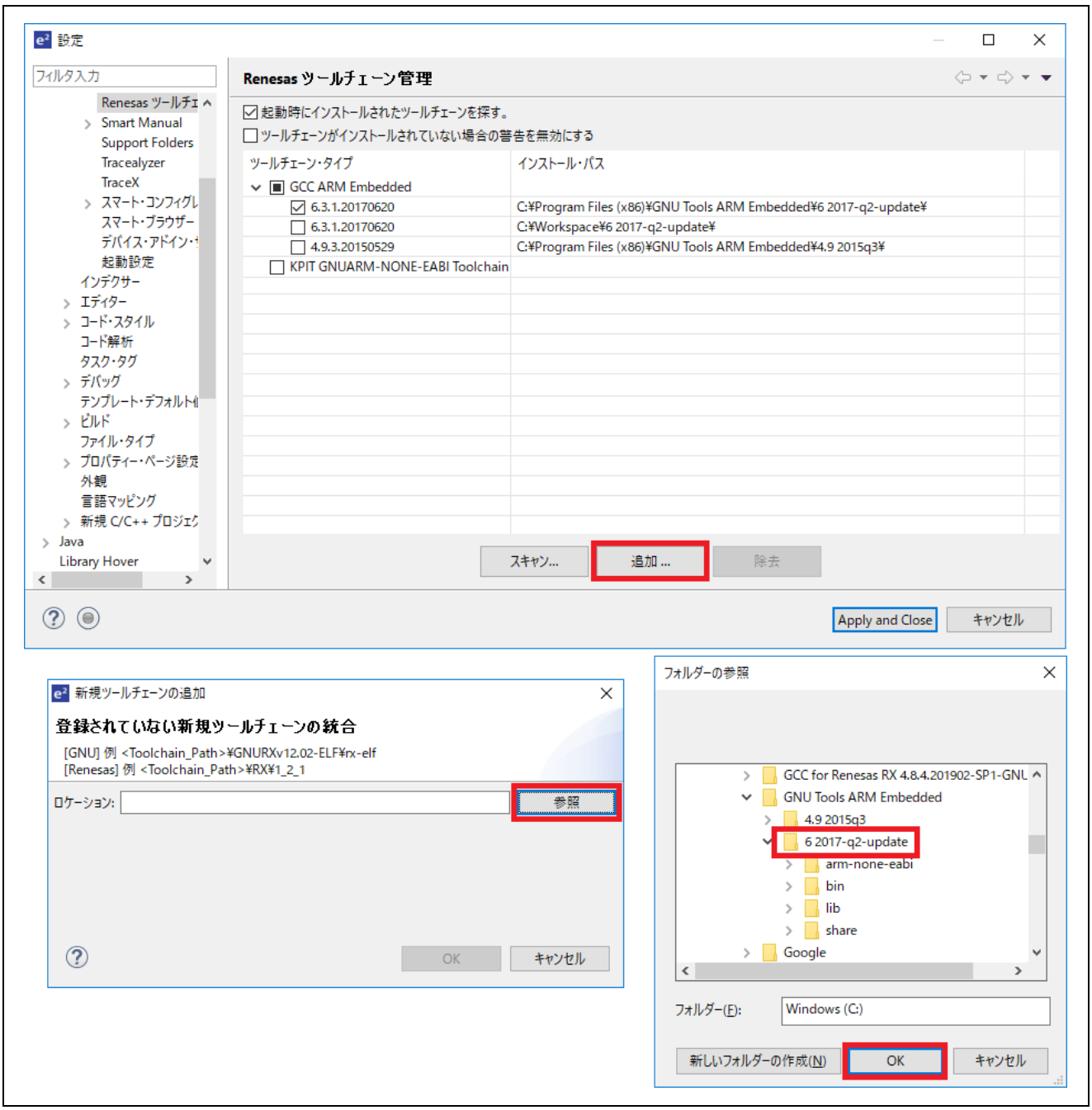

図 2-34 ツールチェーン管理：新規ツールチェーンの登録

3. プロジェクトのインポートおよび作成

この章では、既存プロジェクト（ソフトウェアパッケージのサンプルコードなど）をインポートする方法、および、新規プロジェクトを作成する方法について説明します。

注意： e² studio インストールフォルダ名、プロジェクト名とそのフォルダ、ソースファイル名に英文字、数字、アンダーバーのみを使用してください。

3.1 既存プロジェクトのインポート

ルネサスでは、RZ デバイス開発用のソフトウェアパッケージとして、便利なサンプルコードをサポートしています。このサンプルコードは、e² studio に簡単にインポートすることができます。また、プロジェクトをインポートすることにより、古いプロジェクトも使用することができます。

1. ソフトウェアパッケージをダウンロードします。
2. [プロジェクト・エクスプローラー] を右クリックするか、[ファイル] メニューから [インポート] をクリックして、[インポート] ダイアログボックスを開きます。

ツールチェーンへのe² studioの登録

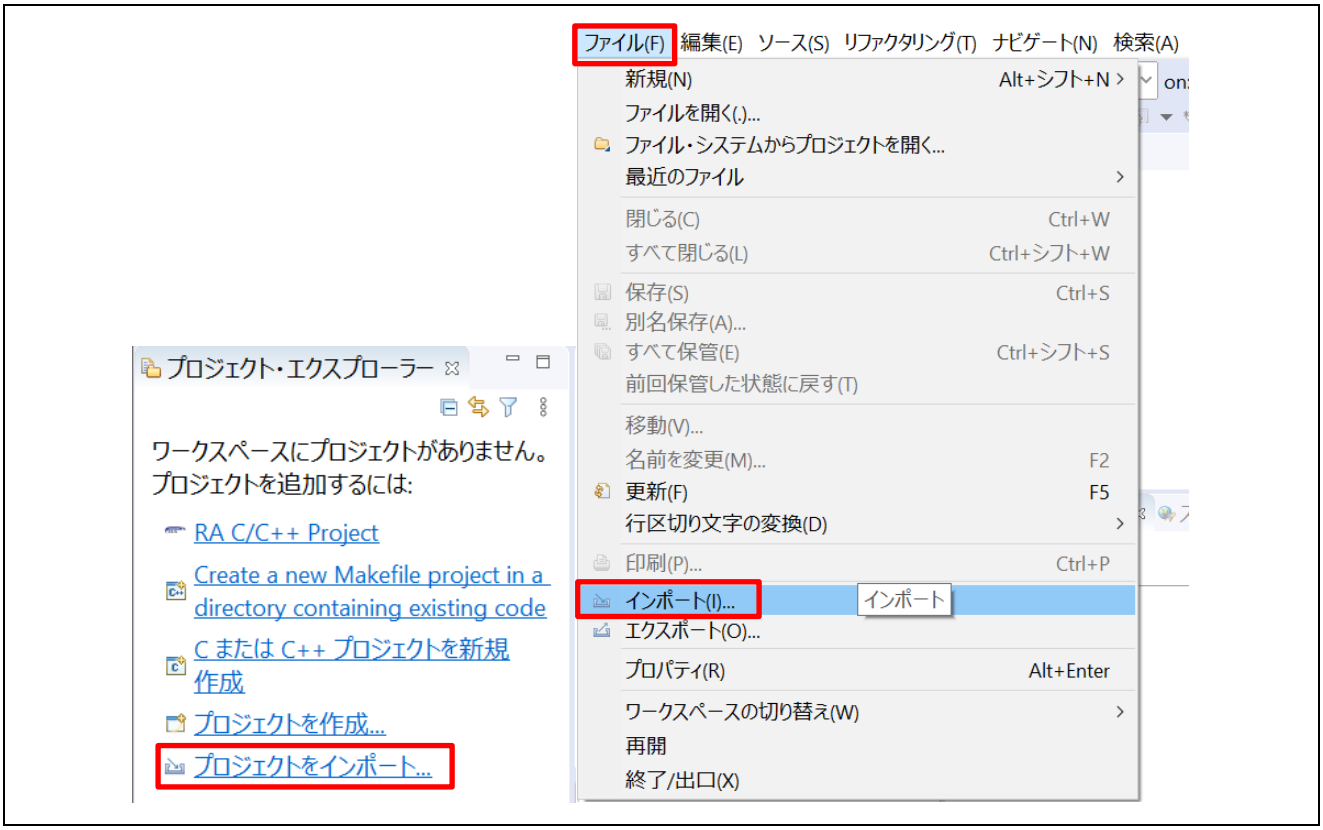

図 3-1 インポートメニュー

3. [一般] から [既存プロジェクトをワークスペースへ] を選択し、[次へ] をクリックします。
4. プロジェクトのフォルダ、あるいはアーカイブ・ファイルを選択して [参照] をクリックし、ワークスペースでプロジェクトのフォルダを選択します。
5.

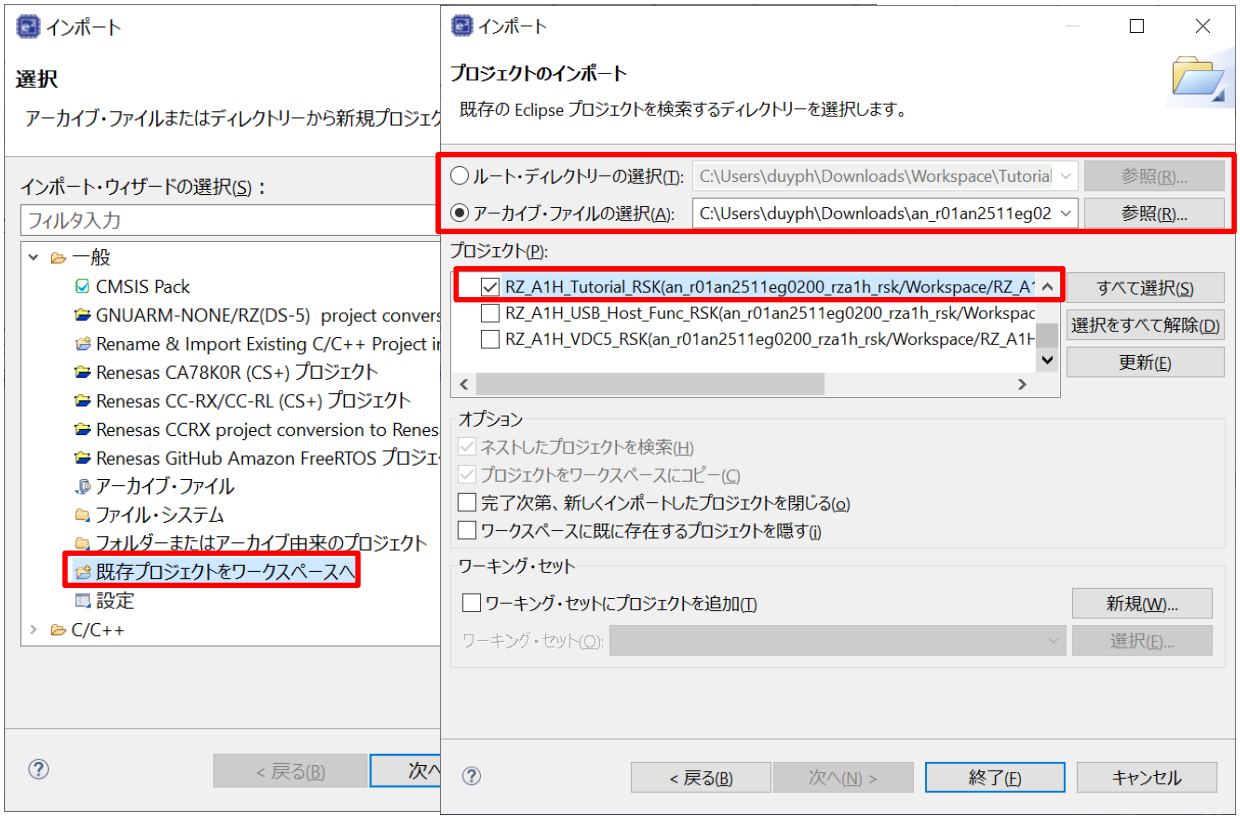

図 3-2 プロジェクトファイルに含むフォルダあるいはアーカイブ・ファイルの選択

5. 古いe² studio (V6.0以前) で生成したプロジェクトは、更新する必要はありません。
‘プロジェクトの更新が必要です’ のポップアップメッセージをクリックします。(次ページへ続きます。)

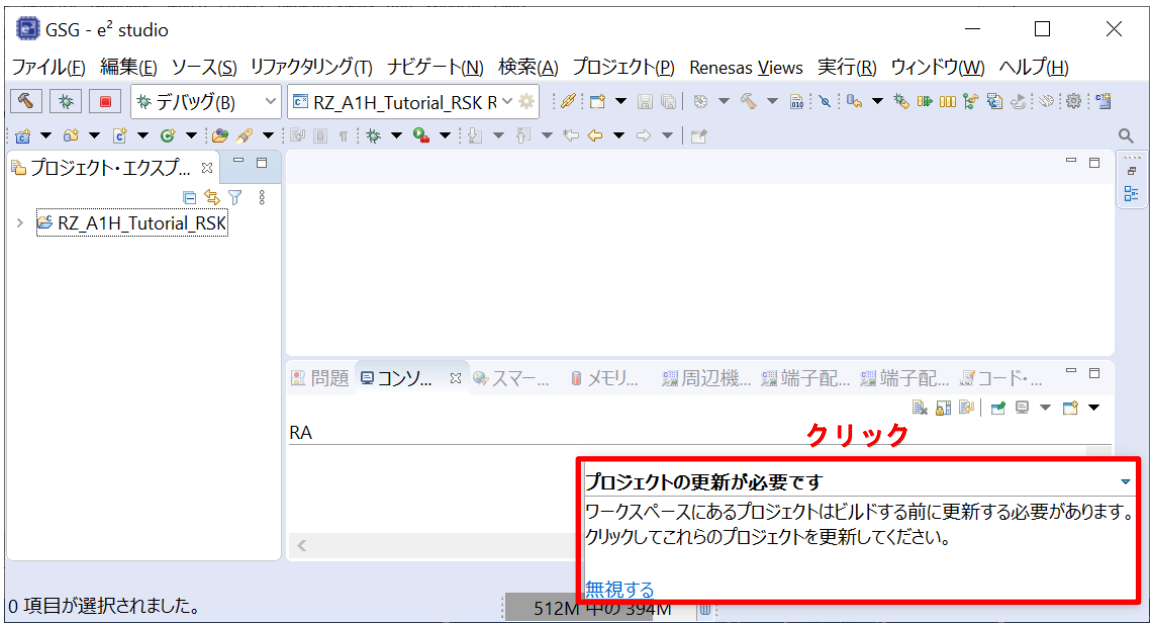

図 3-3 プロジェクトの更新

プロジェクトの右クリックメニューで、[古い e2 studio プロジェクトを更新...] を実行します。

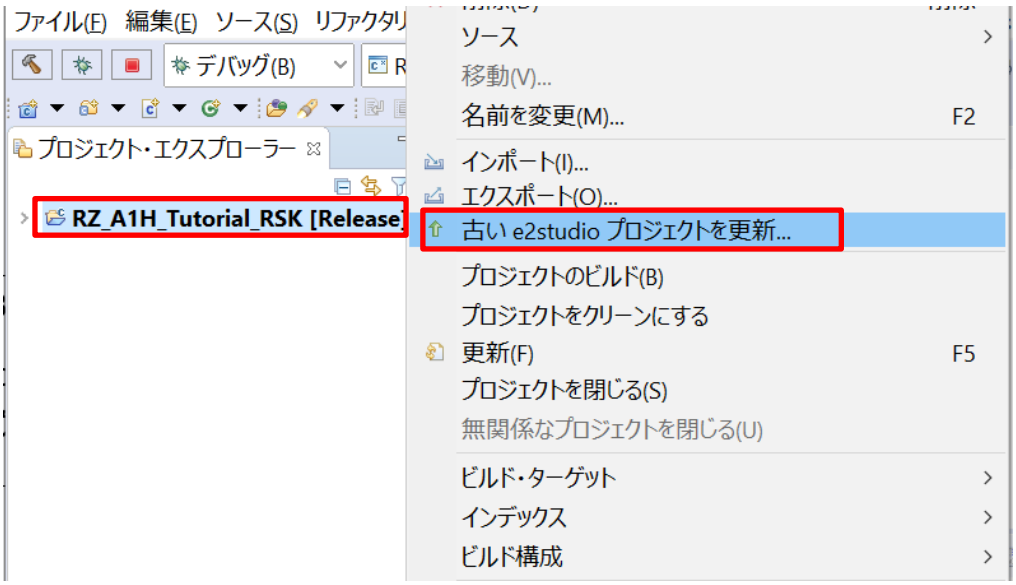

図 3-4 プロジェクトの自動更新

起動構成を変換する必要があります。(図 3-5)。

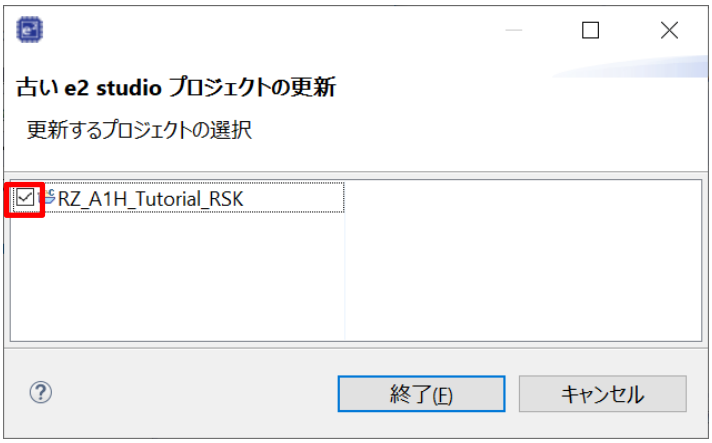

図 3-5 [変換が必要です] メッセージ

3.2 新規プロジェクトの作成

新規プロジェクトを作成する場合、RZ/A2M グループと他の RZ ファミリグループのプロジェクト作成方法が異なります。それぞれのデバイスについて、プロジェクトを作成する方法を説明します。

3.2.1 RZ/A2M のプロジェクトの作成

RZ/A2M Evaluation Board Kit を使用する場合の例を説明します。RZ/A2M で作成されるプロジェクトのタイプは、"Loader Project"または"Application Project"を選択することが可能です。RZ/A2M では、この 2 つの両方のプロジェクトを作成し、ビルド、デバッグ（ダウンロード）する必要があります。

"Loader Project"はローダプログラム作成用のプロジェクトであり、"Application Project"はユーザアプリケーションプログラム作成用のプロジェクトです。"Loader Project"は、RZ/A2M Evaluation Board Kit に搭載されているシリアルフラッシュメモリに高速にアクセスできるように設定した後、"Application Project"に分岐します。（ローダプログラムがダウンロードされていない場合は、アプリケーションプログラムを実行することができません。）

お客様のシステムで使用する場合には、ご使用のシリアルフラッシュメモリに合わせて、ローダプログラムの内容を変更してください。

Windows の [スタート] メニューから e² studio を起動し、ワークスペースディレクトリを指定します。このワークスペースに作成したプロジェクトが追加されます。

新規にプロジェクトを作成するには、以下の手順を実行してください。

1. [ファイル] → [新規] → [C/C++ Project] の順にクリックすると、新規プロジェクト作成ウィザードが起動します(下図)。

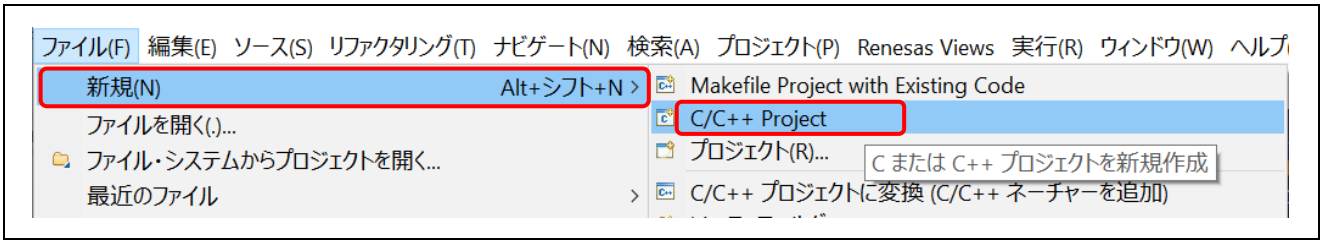

図 3-6 新規プロジェクト作成ウィザード

2. 新規プロジェクトのテンプレートを選択します(例えばRZデバイスの場合は「GCC for Renesas RZ C/C++ Executable Project」)。[次へ(N)>] をクリックすると次の画面に進みます。

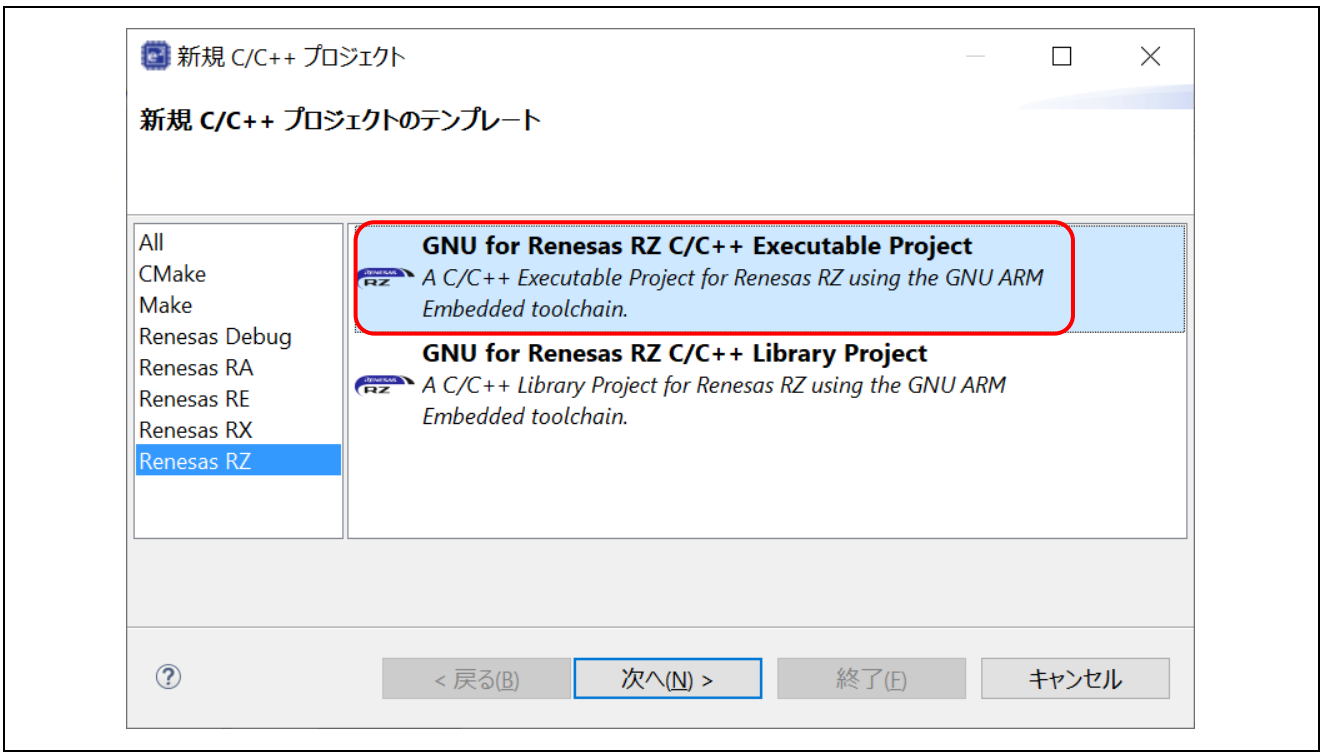

図 3-7 プロジェクトテンプレートの選択

3. 最初にLoaderプロジェクトを作成します。プロジェクト名"Loader"を入力し、[次へ(N)>] で進めます。

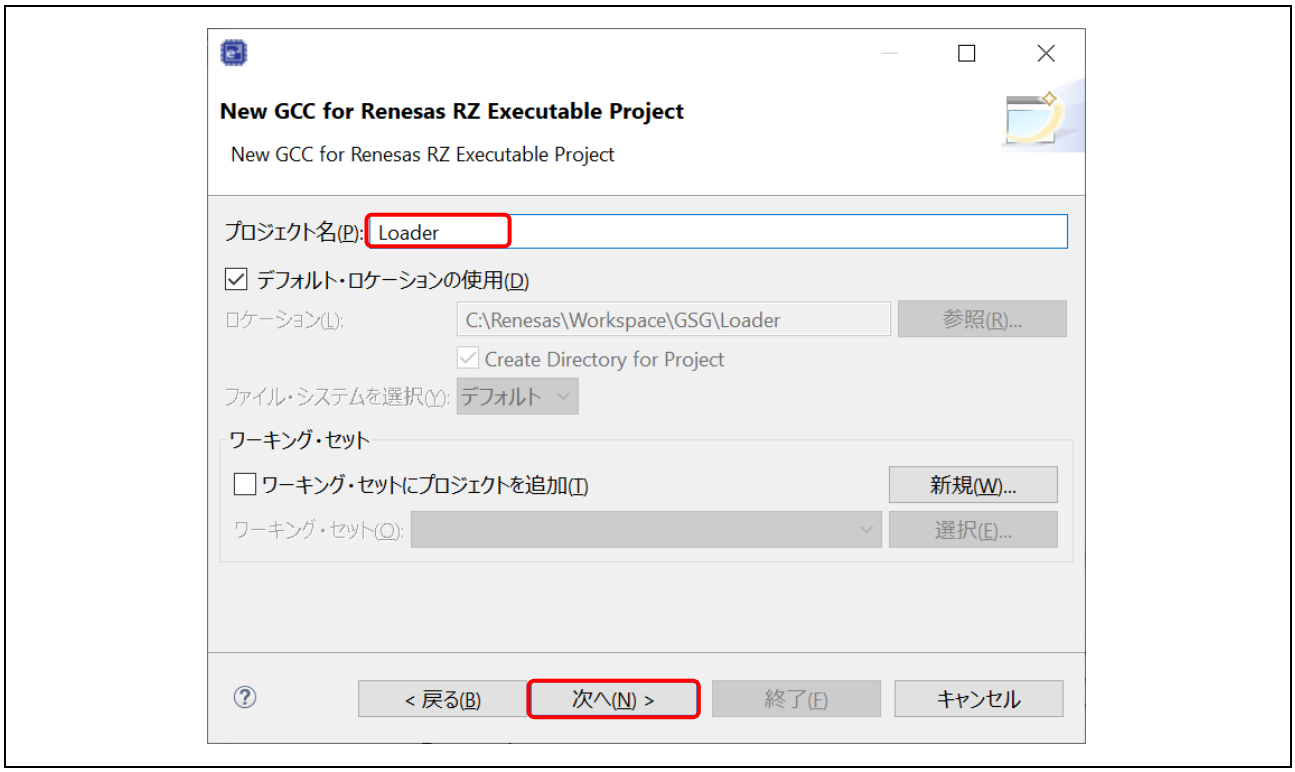

図 3-8 プロジェクト名の指定

4. 以下の設定を選択します。

- 言語：“C”
- ツールチェーン：“GCC ARM Embedded”
- ツールチェーン・バージョン：“6.3.1.20170620”
- ターゲット・デバイス：“R7S921053”
- プロジェクトタイプ：“Loader Project”

[Hardware Debug 構成を生成] で、“J-Link ARM”を選択し、[次へ(N)>] で進めます。

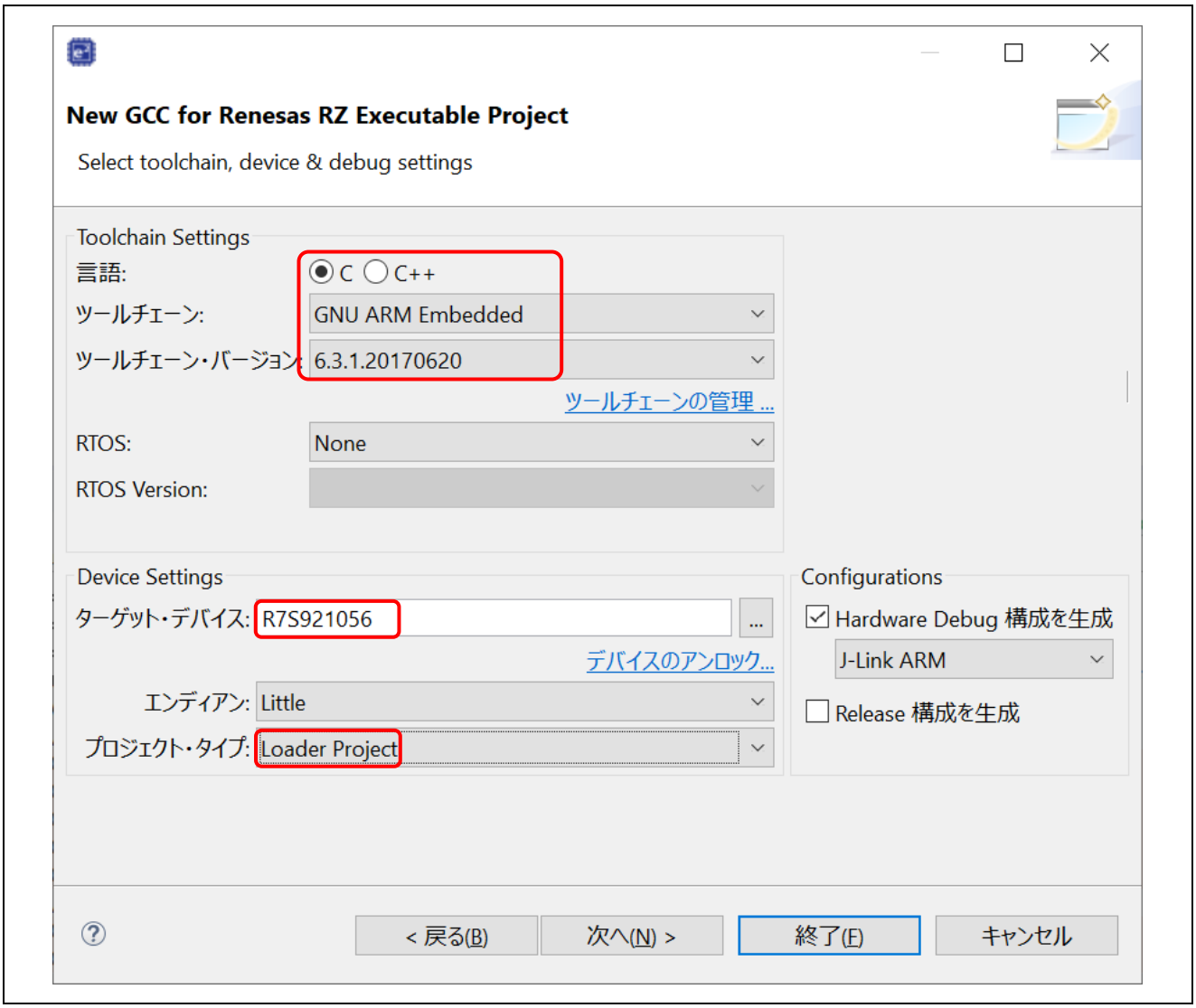

図 3-9 Select toolchain, device & debug settings：RZ/A2M Evaluation Board Kit の場合

5. “Loader Project”はSmart Configuratorに対応していないため、そのまま[次へ(N)>] ボタンで次の画面に進みます。

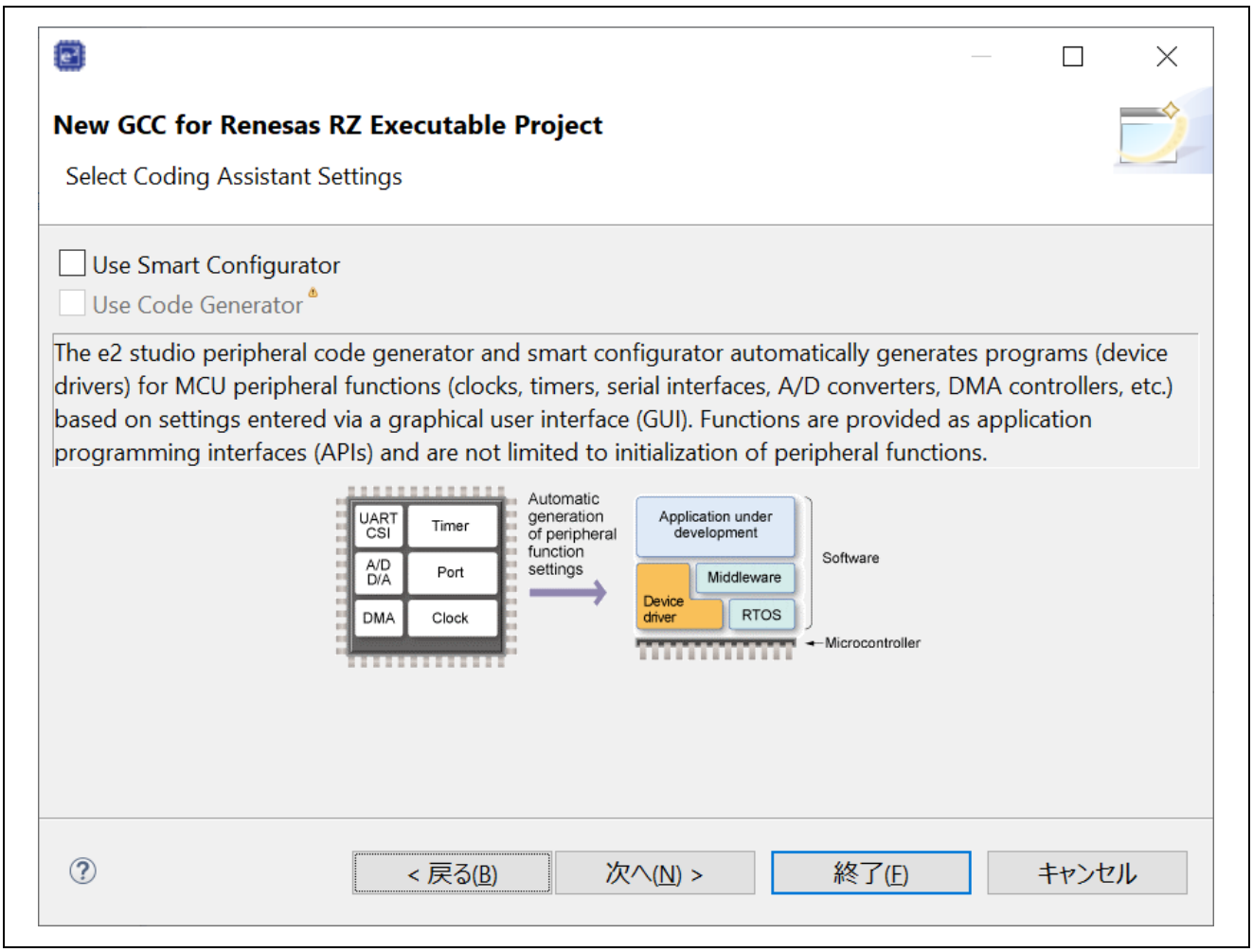

図 3-10 コーディング・アシスタントツールの選択

6. 初期設定ではUSB、RTCは使用しない設定としています。“Loader Project”ではそのまま[次へ(N)>] ボタンで次の画面に進みます。

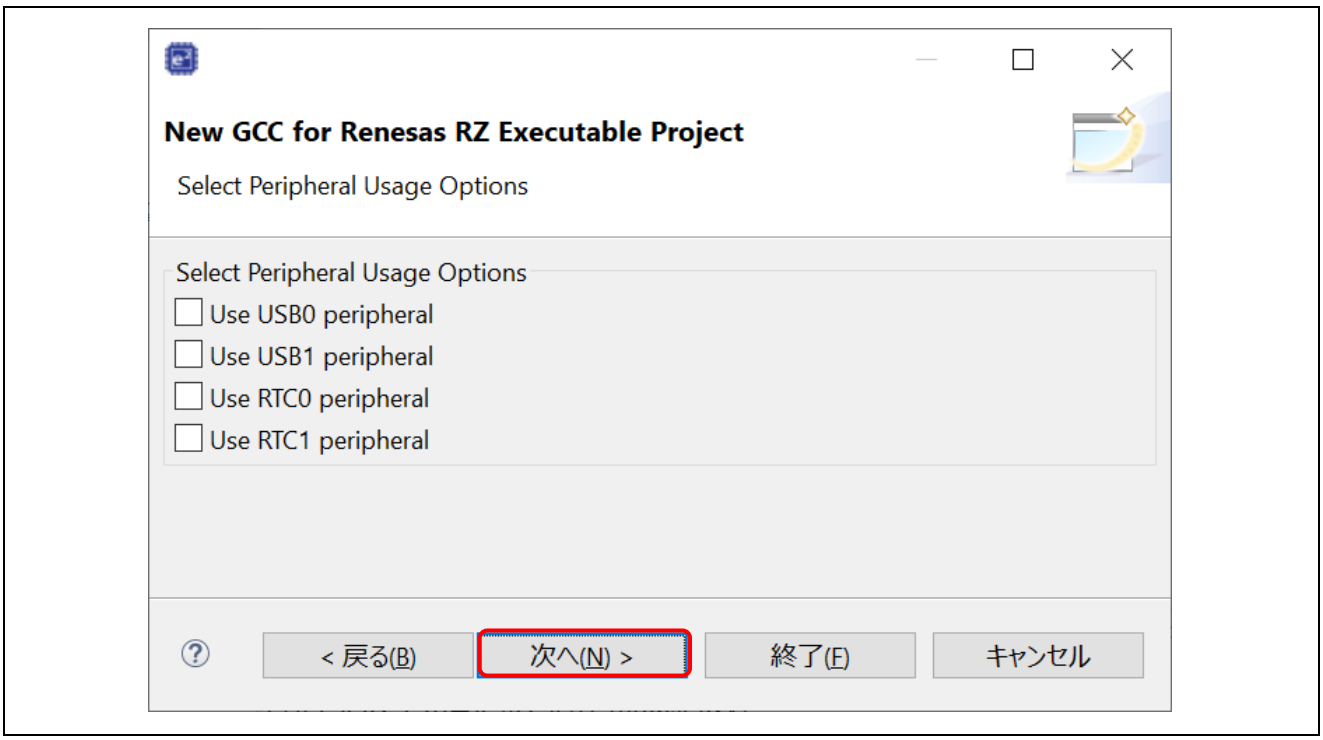

図 3-11 Peripheral 追加オプションの選択

7. CPU追加オプションはデフォルトの設定を変更せず、[次へ(N)>]ボタンを押してください。

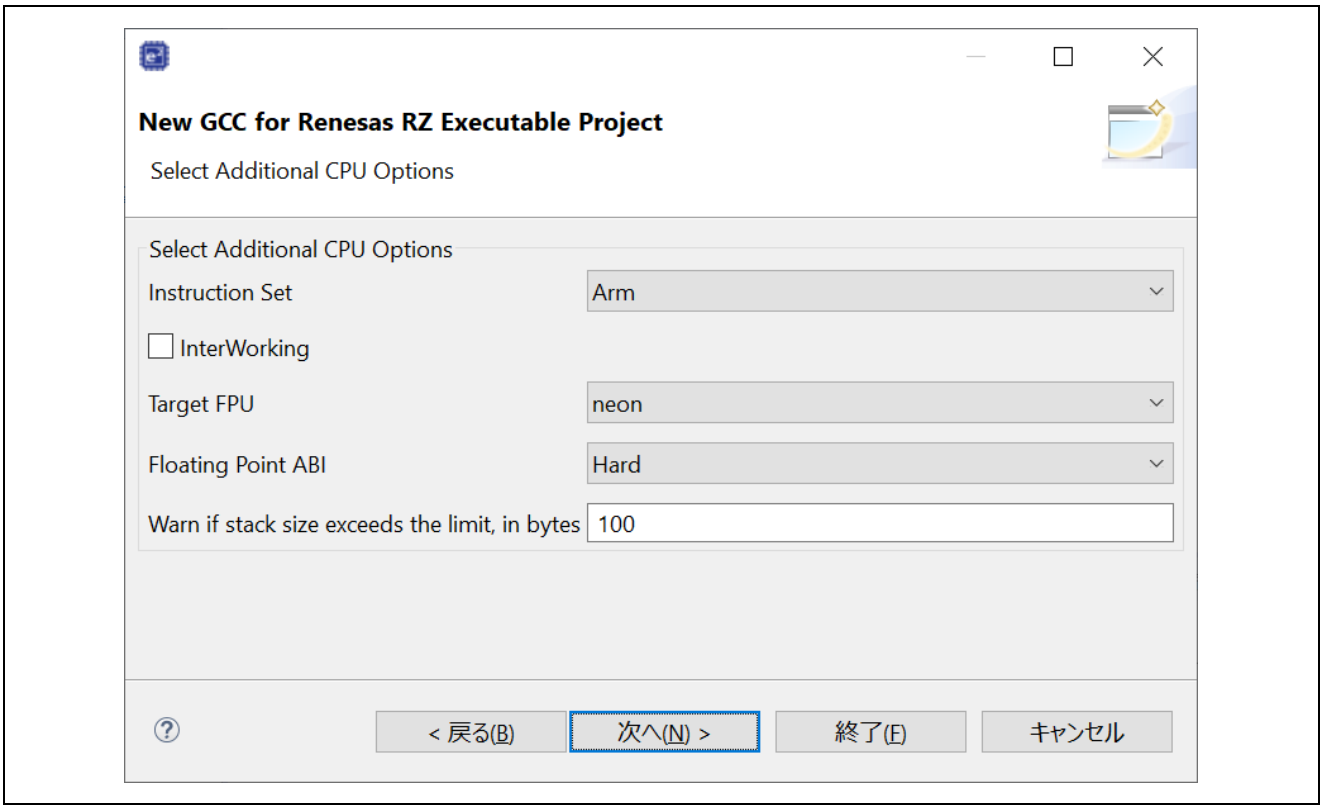

図 3-12 CPU 追加オプションの選択

8. ‘Project-Built’を選択し、 [次へ(N)>]ボタンを押してください。

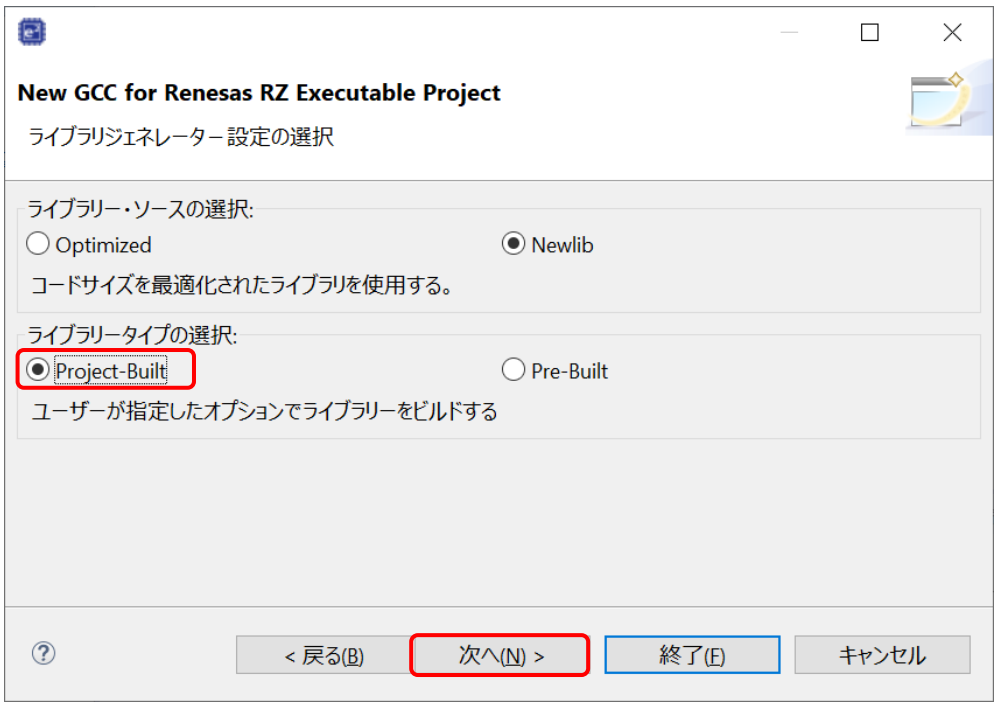

図 3-13 ライブラリ・ジェネレータ設定の選択

9. プロジェクトの要約が表示されます。最後に[終了(F)]ボタンを押すとプロジェクトが作成されます。

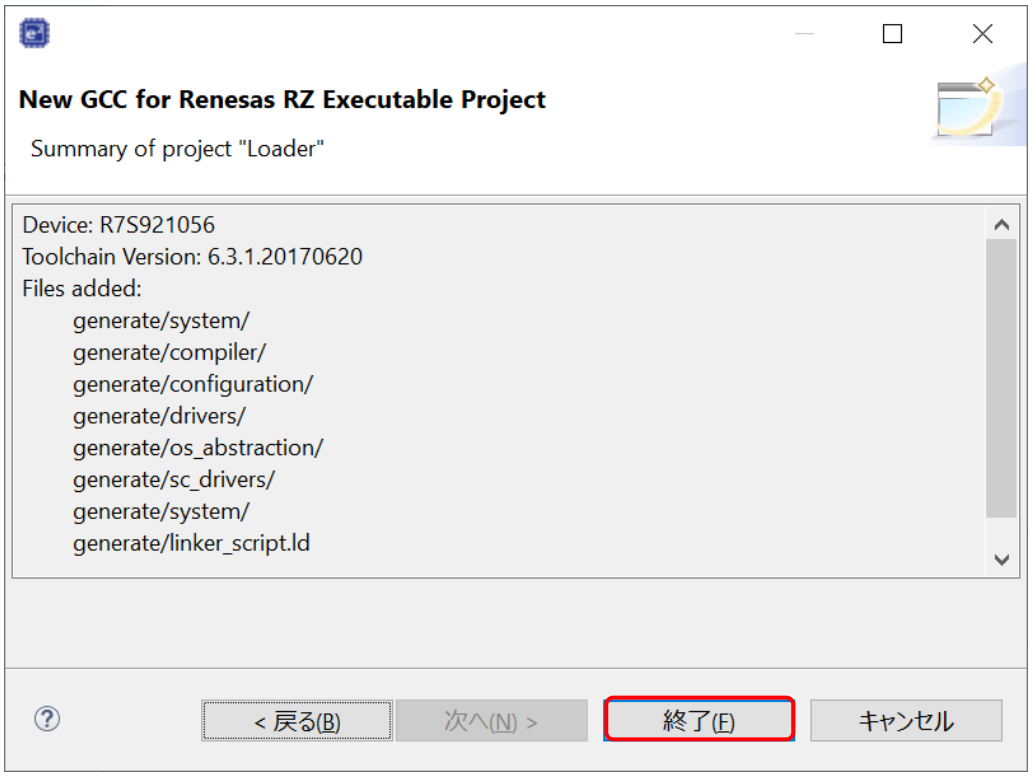

図 3-14 新規プロジェクト作成ウィザード (要約)

10. 「Loader」という新しいCプロジェクトが作成されます。

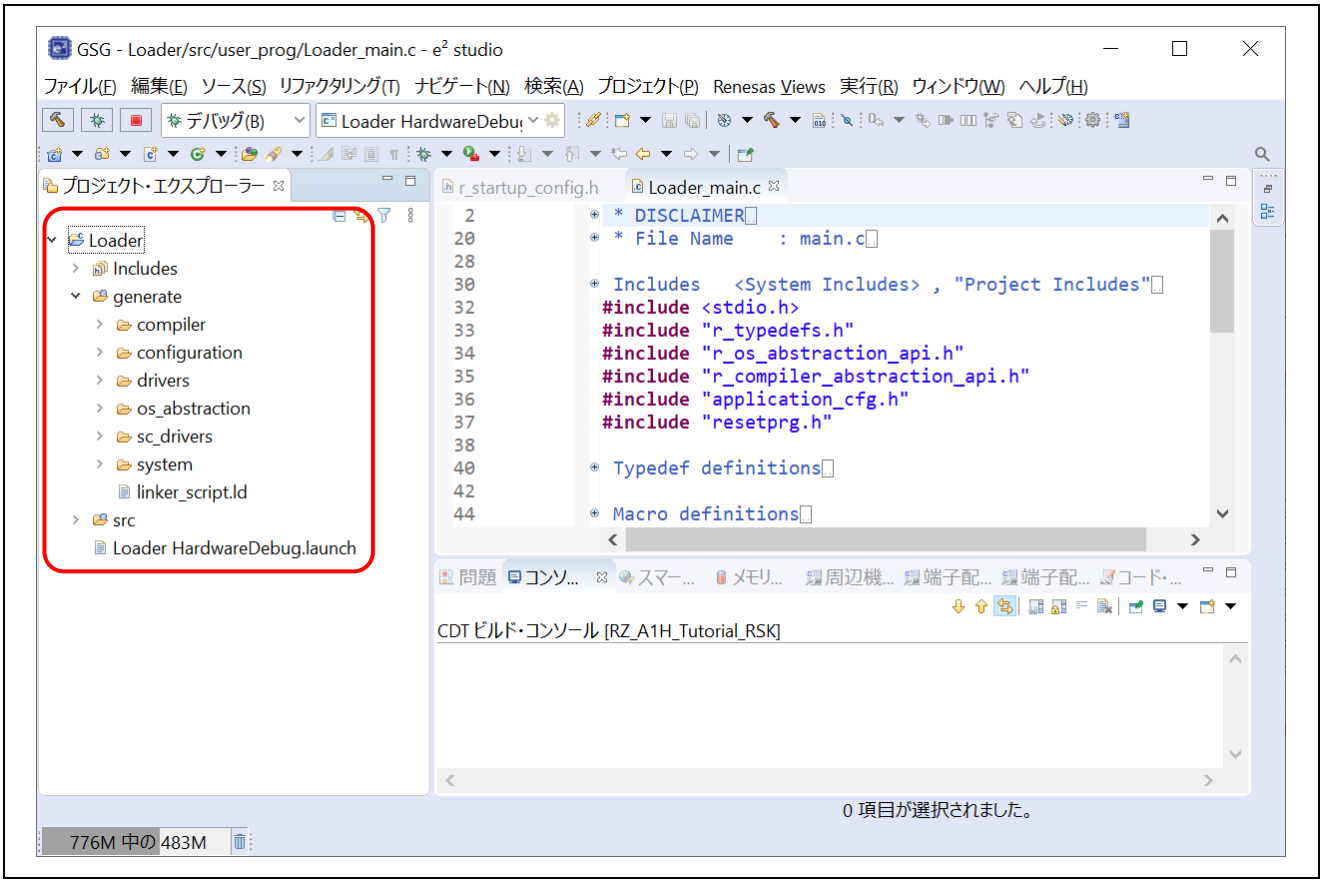

図 3-15 作成された新規 C プロジェクト

11. 次にApplicationプロジェクトを作成します。 [ファイル] → [新規] → [C/C++ Project] の順にクリックすると、新規プロジェクト作成ウィザードが起動します(下図)。

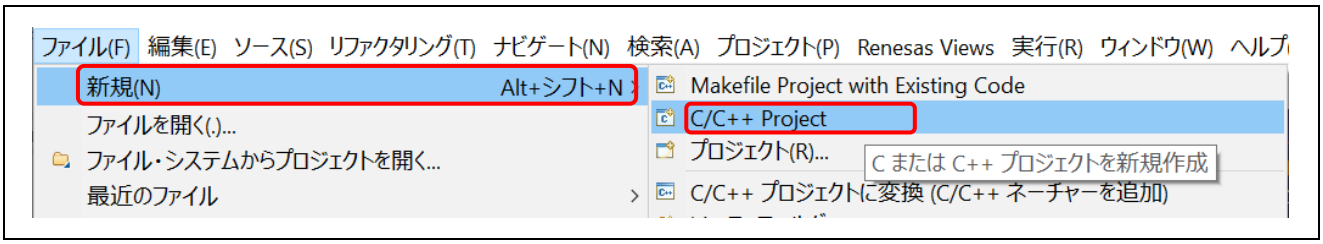

図 3-16 新規プロジェクト作成ウィザード

12. 新規プロジェクトのテンプレートを選択します。[次へ(N)>] をクリックすると次の画面に進みます。

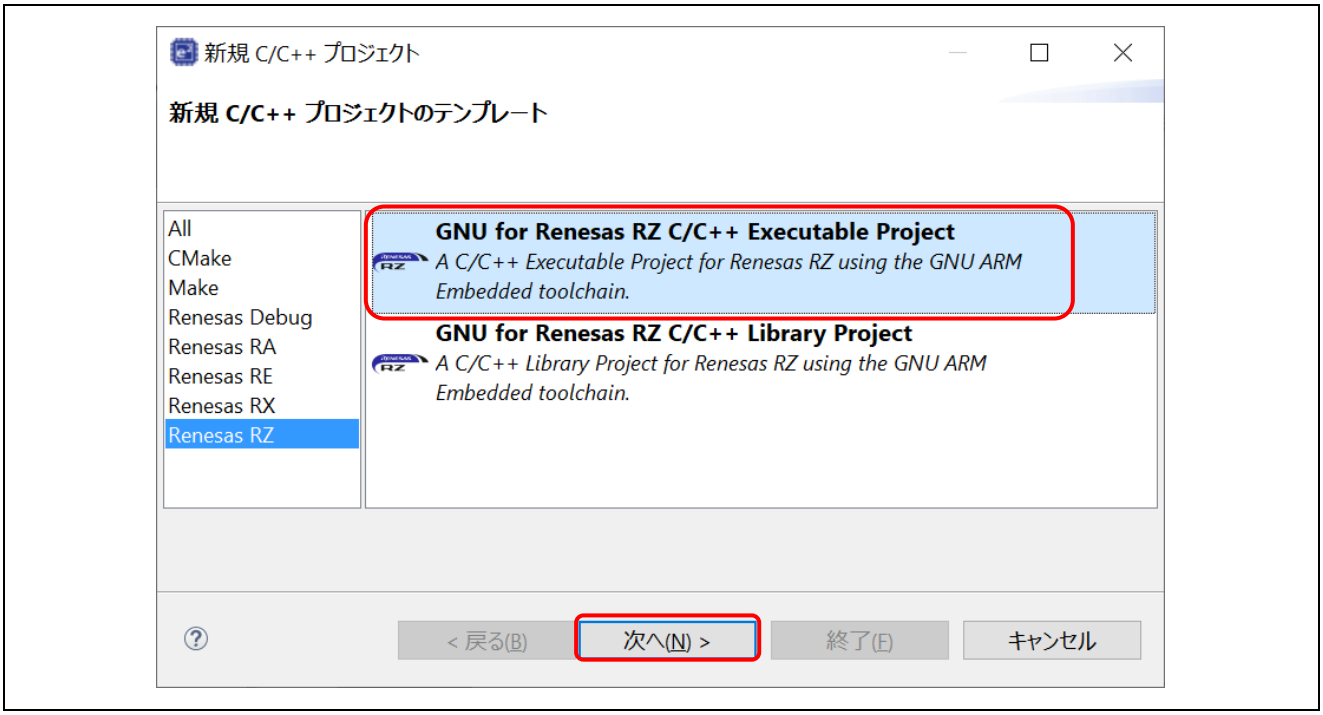

図 3-17　プロジェクトテンプレートの選択

13. プロジェクト名"Application"を入力し、[次へ(N)>] で進めます。

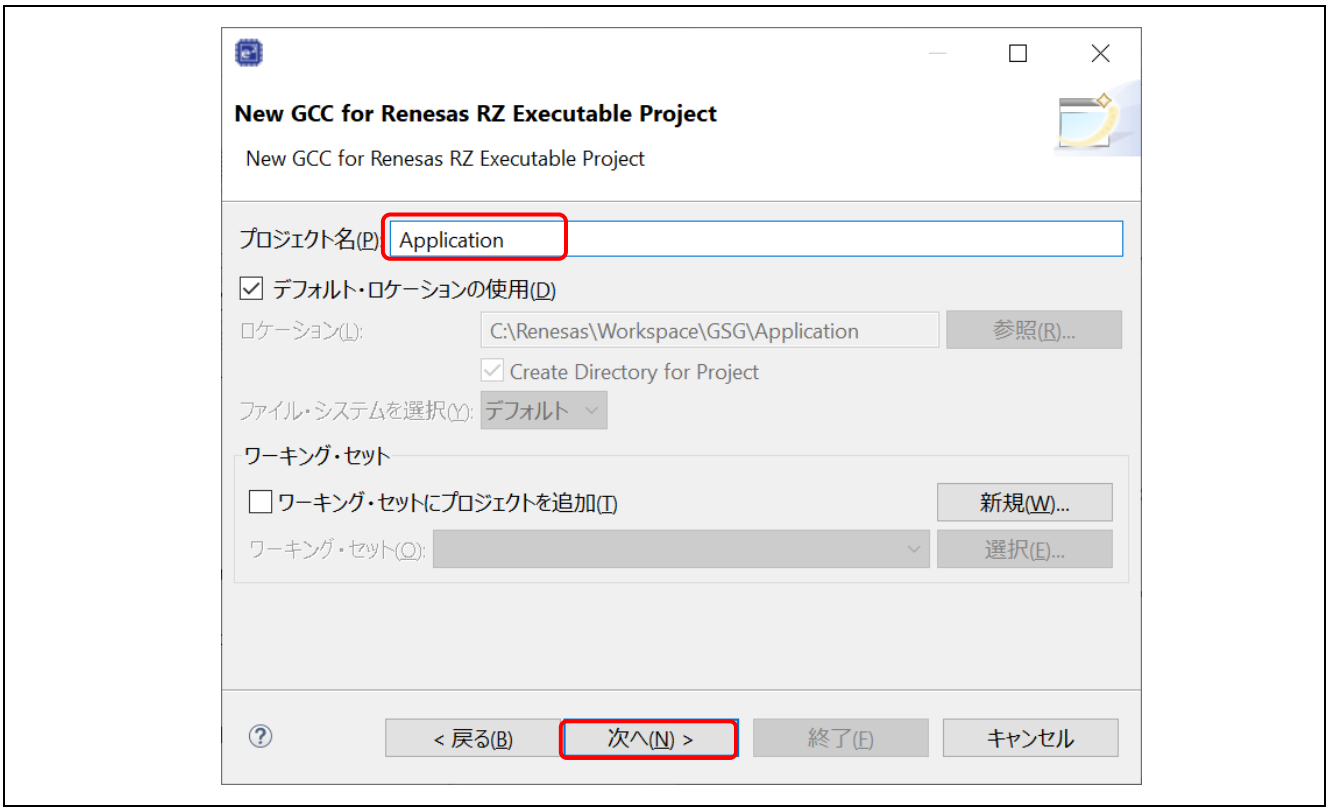

図 3-18　プロジェクト名の指定

14. 以下の設定を選択します。

- 言語：“C” or “C++”
- ツールチェーン：“GCC ARM Embedded”
- ツールチェーン・バージョン：“6.3.1.20170620”
- ターゲット・デバイス：“R7S921053”
- プロジェクトタイプ：“Application Project”

[Hardware Debug 構成を生成] で、“J-Link ARM”を選択し、[次へ(N)>] で進めます。

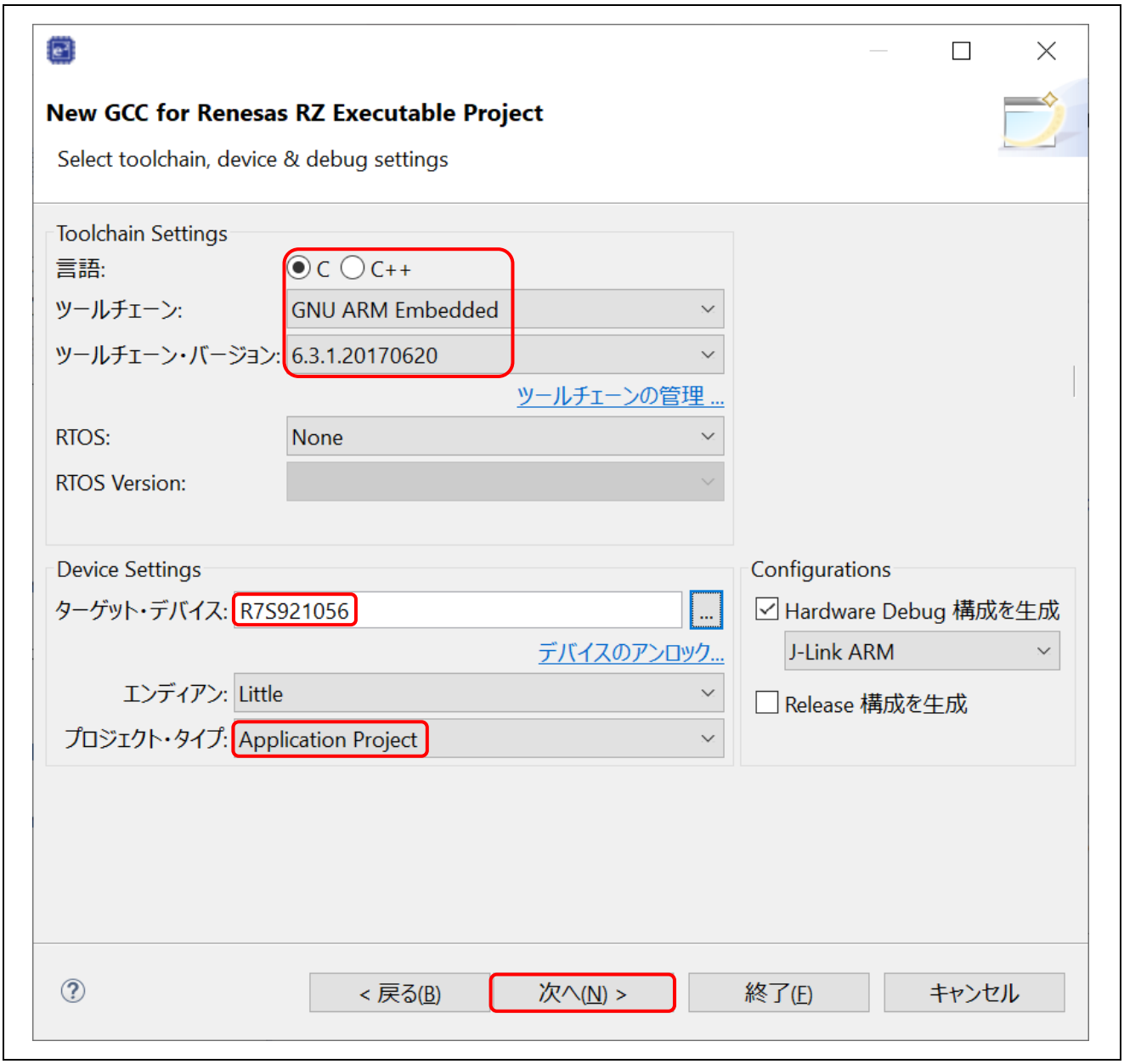

図 3-19 Select toolchain, device & debug settings：RZ/A2M Evaluation Board Kit の場合

15. 必要に応じて、コーディング・アシスタント機能を使用することができます。

注： ［スマート・コンフィグレータ］はコード生成とコンフィグレータを1画面で統一的に扱えるようにしたインタフェースを提供します。クロック設定、割り込み設定、ピン設定が全て含まれます。

“Use Smart Configurator”を選択して、[次へ(N)>] ボタンで次の画面に進みます。

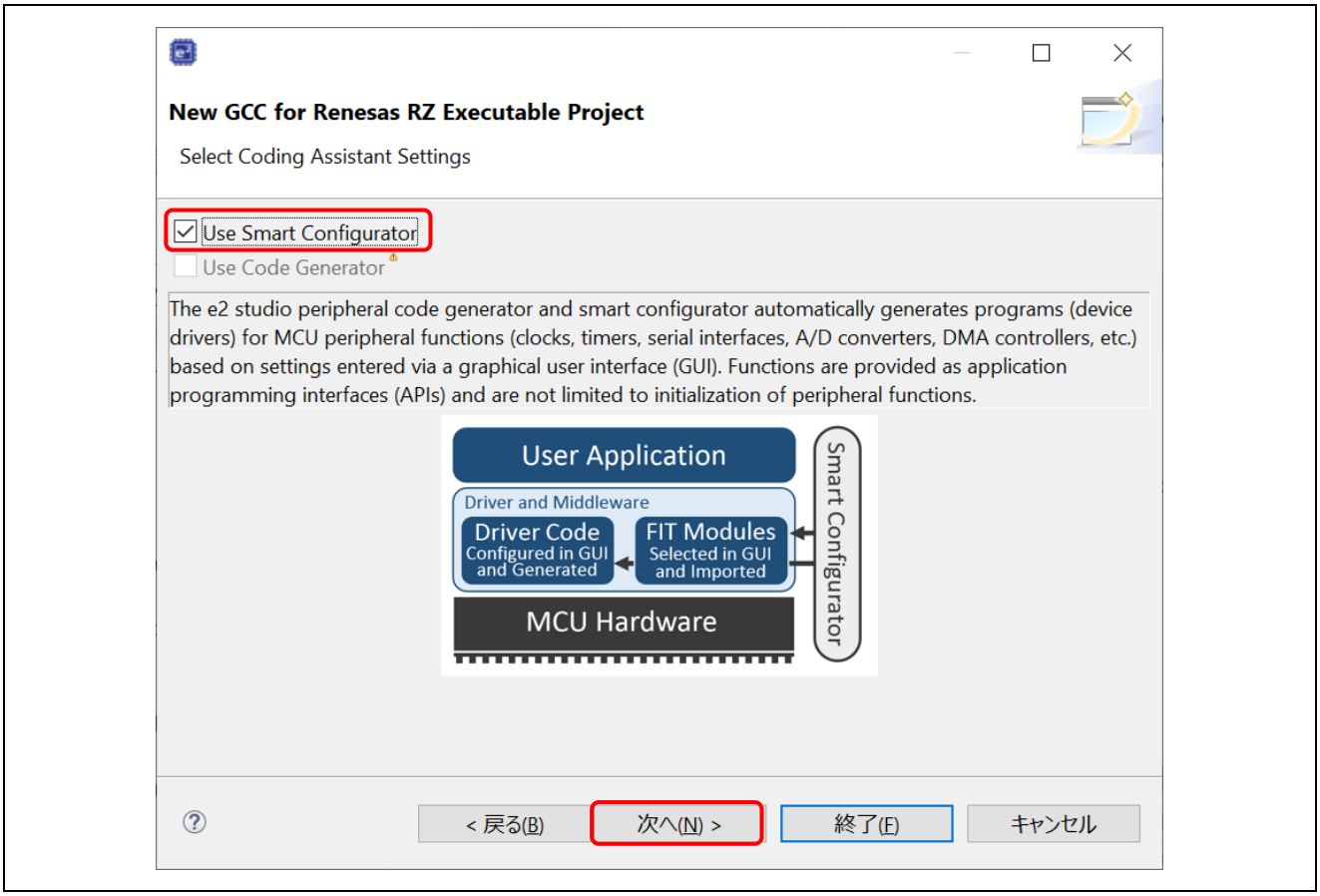

図 3-20 コーディング・アシスタントツールの選択

16. 初期設定ではUSB、RTCは使用しない設定としています。USBおよびRTCをご使用の場合には、使用する周辺モジュール/チャネルを選択してください。

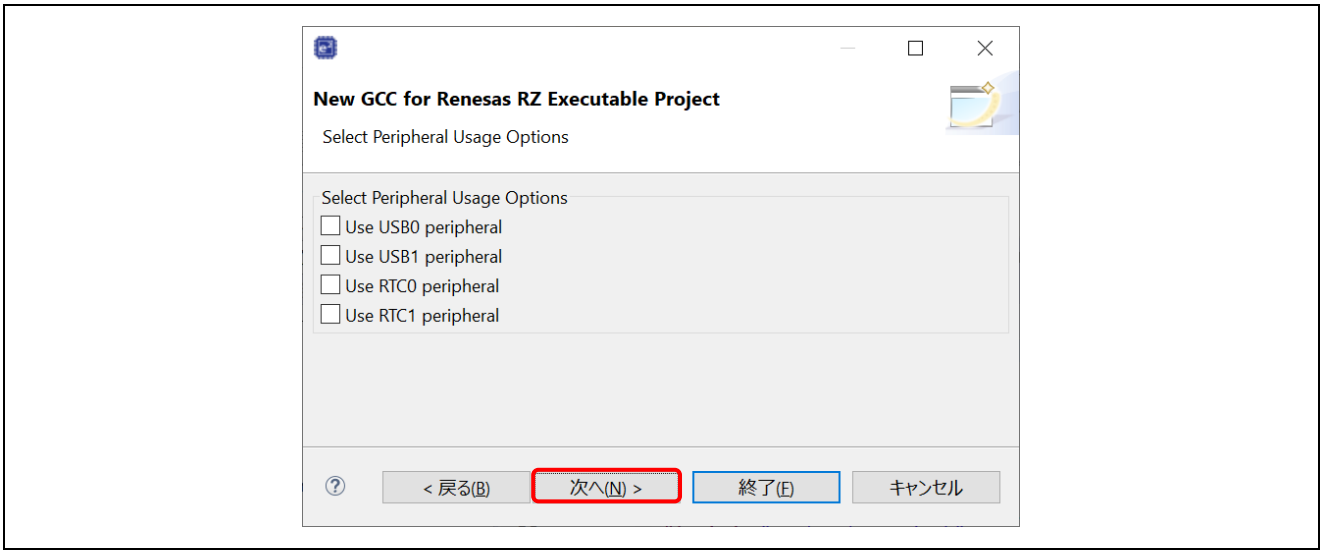

図 3-21 Peripheral 追加オプションの選択

17. CPU追加オプションはデフォルトの設定を変更せず、[次へ(N)>]ボタンを押してください。

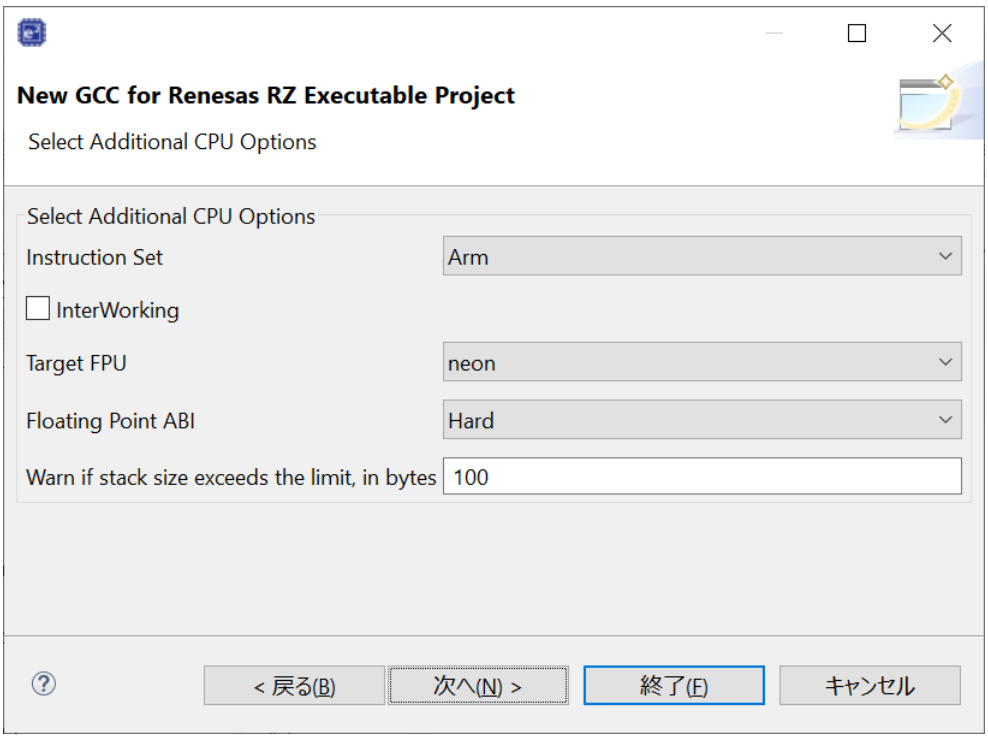

図 3-22　CPU 追加オプションの選択

18. ライブラリータイプの選択において、 'Project-Built'を選択してください。選択後、[次へ(N)>]ボタンを押してください。

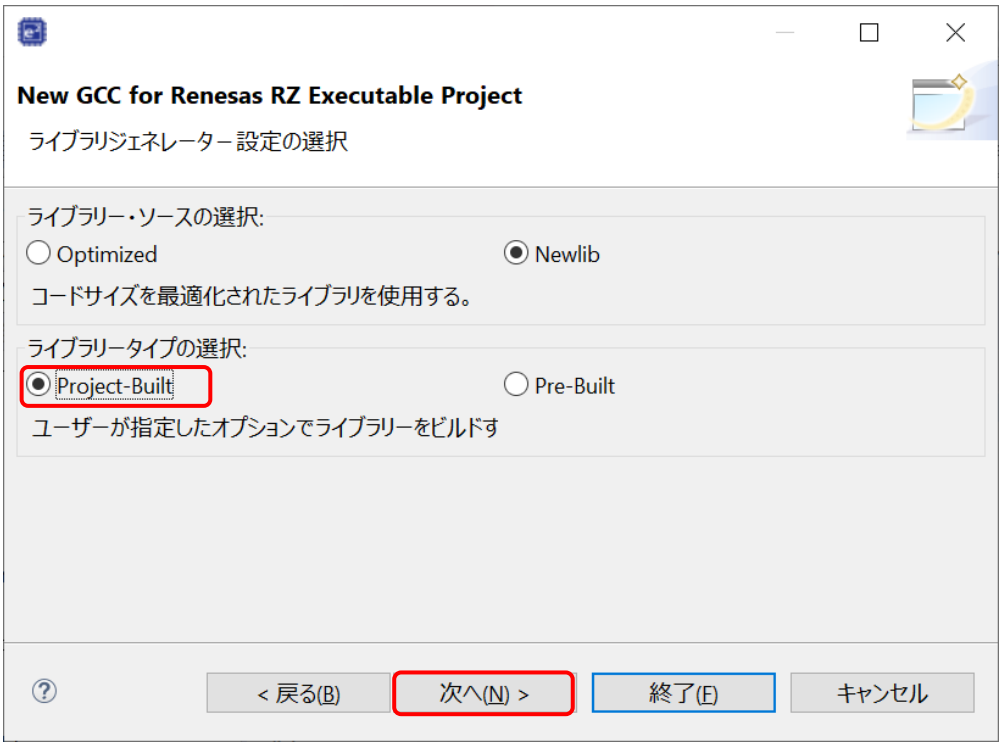

図 3-23　ライブラリ・ジェネレータ設定の選択

19. プロジェクトの要約が表示されます。最後に[終了(F)]ボタンを押すとプロジェクトが作成されます。

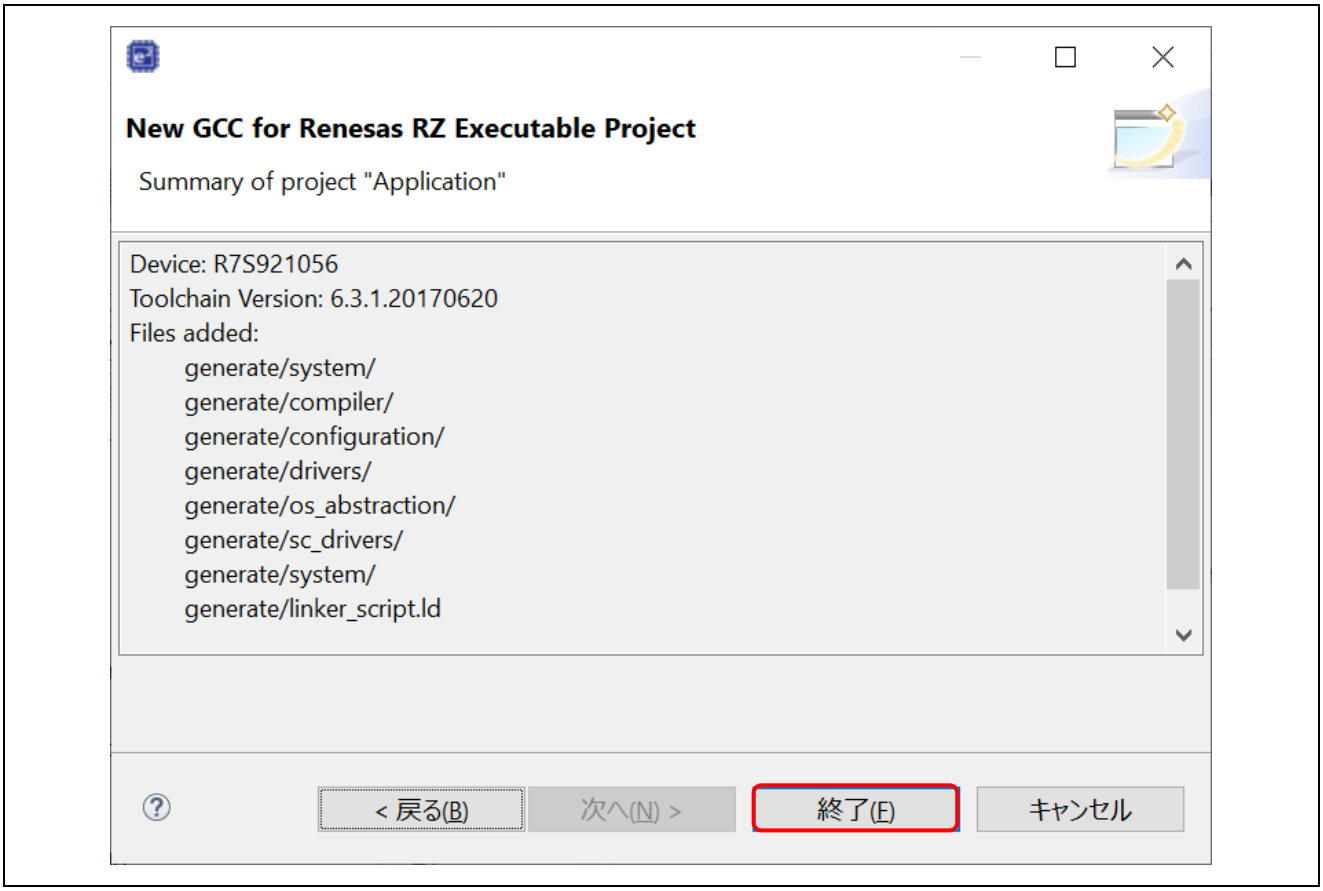

図 3-24　新規プロジェクト作成ウィザード (要約)

20. “関連付けられたパースペクティブを開く”を選択します。

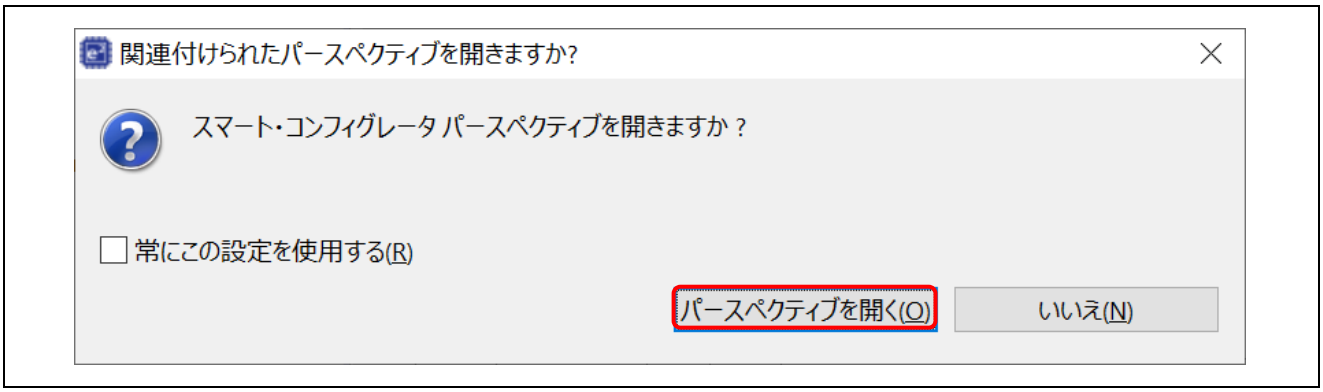

図 3-25　パースペクティブ選択画面

21. 「Application」という新しいCプロジェクトが作成されます。

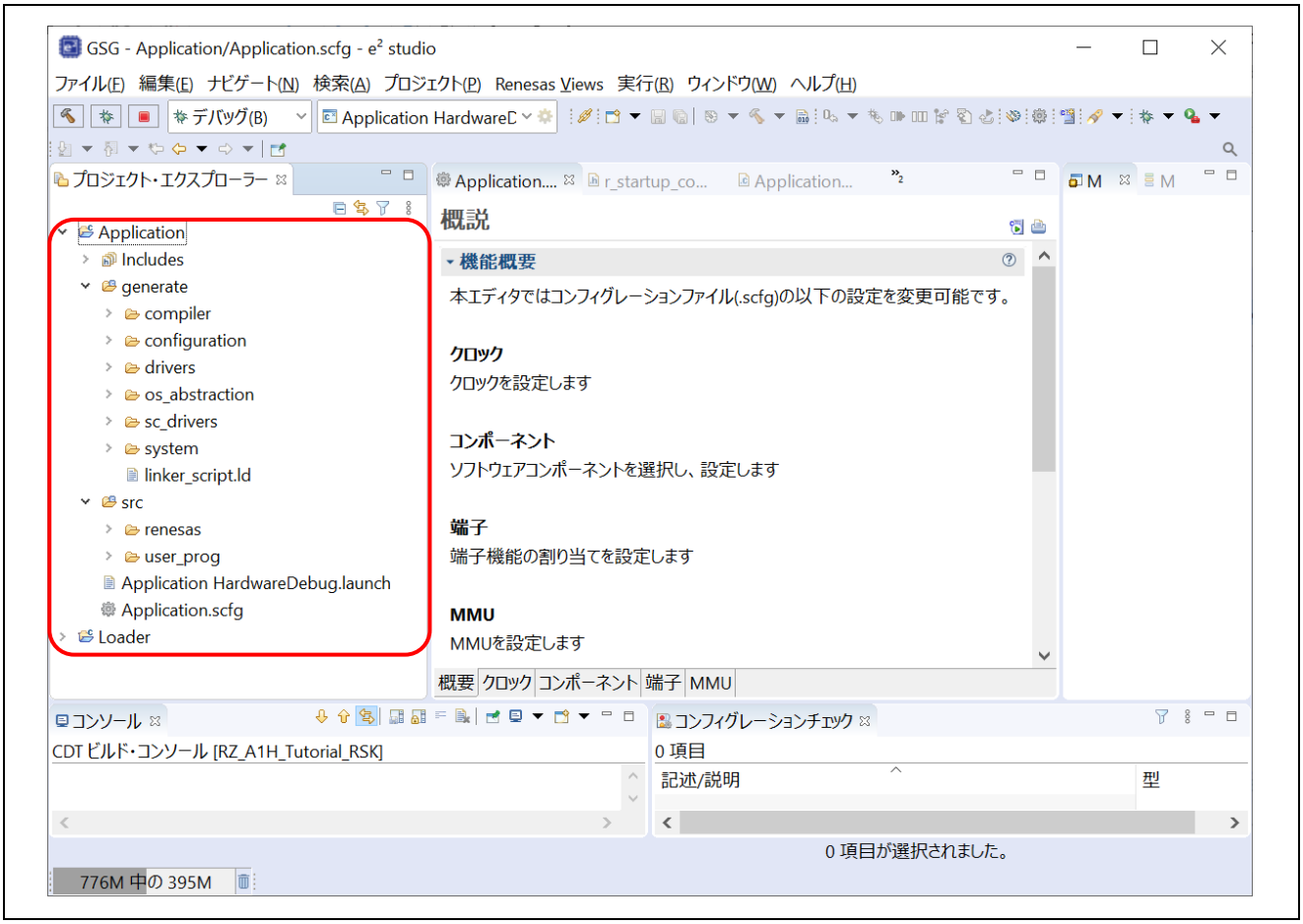

図 3-26　作成された新規 C プロジェクト

このプロジェクトは、“Application.c” というアプリケーションファイルと、自動的に生成されるファイル一式から構成されています。[プロジェクト・エクスプローラー] パネルではすべてのプロジェクトおよびソースファイルを Windows エクスプローラーと同様のフォルダ階層として表示します。

プロジェクトをバックアップする際の注意事項：

- 「.」(ドット)で始まる名前のファイルやフォルダ（.project など）にはプロジェクトの設定情報が含まれますので、バックアップを取る際にはこれらのファイルやフォルダも含めてプロジェクトのフォルダ全体を圧縮するなどしてください。
- 他のプロジェクトのファイルを参照する設定など、プロジェクト間で共有される設定を保存するためには、ワークスペース全体をバックアップする必要があります。

3.2.2 RZ ファミリ向けのプロジェクト作成（RZ/A1、RZ/T）

RZ/A1H Renesas Starter Kit を使用する場合の例を説明します。

作成されたプロジェクトは、RZ/A1 の大容量内蔵 RAM で動作します。フラッシュメモリで動作させる場合には、お客様にてフラッシュメモリの仕様やメモリ配置に合わせてプロジェクトの内容を変更してください。

RZ/T のプロジェクトの生成方法は RZ/A1H とほぼ同じです。

e² studio は *RZ/G* 向けには Linux アプリケーション開発用の「RZ Linux C/C++ Project」をサポートします。この機能は e² studio をインストールする時、「Support for RZ/G」の選択が必要です。詳細は図 2-2 をご参照ください。

Windows の [スタート] メニューから e² studio を起動し、ワークスペースディレクトリを指定します。このワークスペースに作成したプロジェクトが追加されます。新規にプロジェクトを作成するには、以下の手順を実行してください。

1. [ファイル] → [新規] → [C/C++ Project] の順にクリックすると、新規プロジェクト作成ウィザードが起動します(下図)。

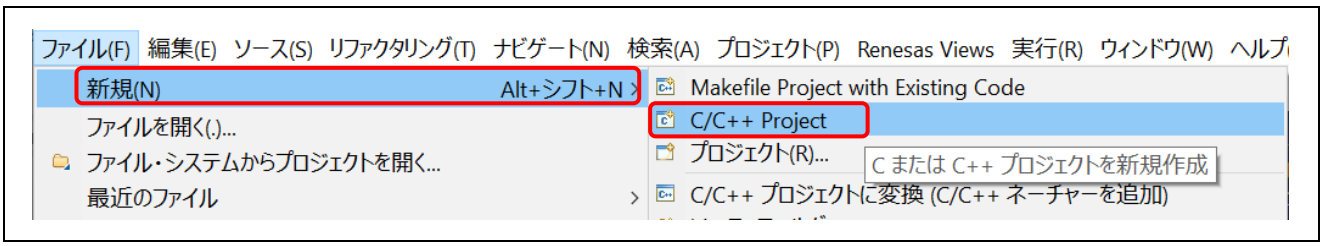

図 3-27 新規プロジェクト作成ウィザード

2. 新規プロジェクトのテンプレートを選択します(例えばRZデバイスの場合は「GCC for Renesas RZ C/C++ Executable Project」)。[次へ(N)>] をクリックすると次の画面に進みます。

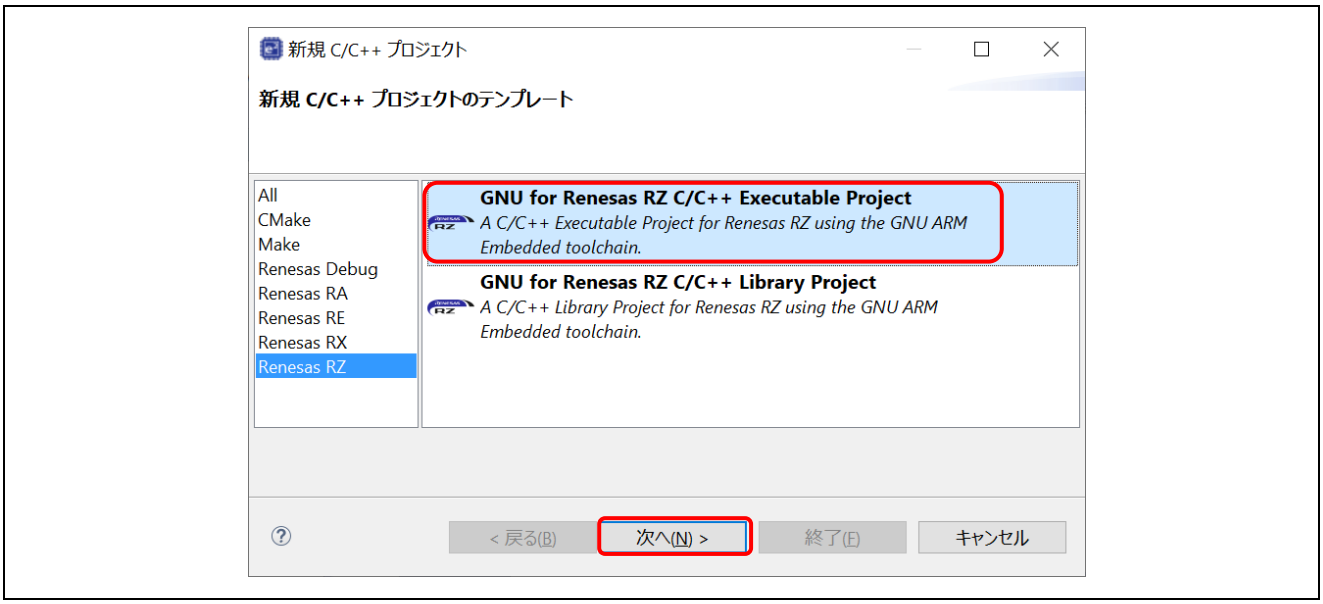

図 3-28 プロジェクトテンプレートの選択

3. プロジェクト名を入力し、[次へ(N)>] で進めます。

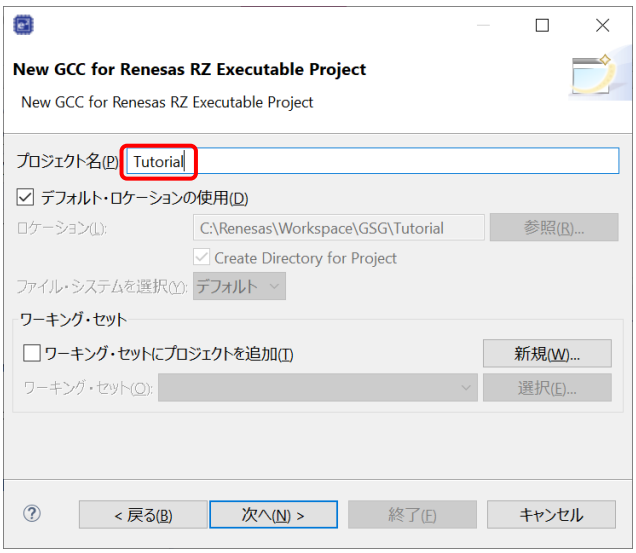

図 3-29 プロジェクト名の指定

- 言語：“C”
- ツールチェーン：“GCC ARM Embedded”
- ツールチェーン・バージョン：“4.9.3.20170620”
- ターゲット・デバイス：“R7S721001”

  [Hardware Debug 構成を生成] で、“J-Link ARM”を選択し、[次へ(N)>] で進めます。

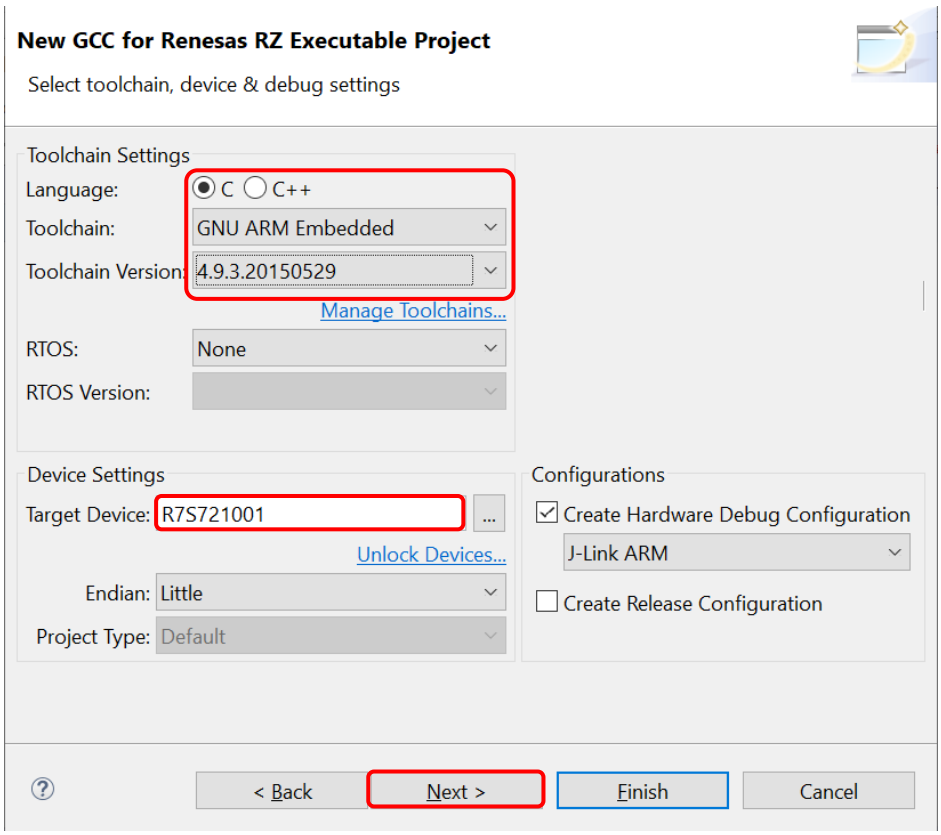

図 3-30 Select toolchain, device & debug settings：RZ/A1H Renesas Starter Kit の場合

4. 必要に応じて、コード生成サポート機能を選択してください。

   注意：

   - コード生成プラグイン（Peripheral Code Generator）は GUI 操作でデバイスドライバや周辺機能のコードの生成機能をサポートします。周辺機能を操作する関数を API として提供する他、様々な機能を提供します。
   - スマート・コンフィグレータ（Smart Configurator）はコード生成とミドルウェアの設定機能を一つの GUI にまとめたツールです。スマート・コンフィグレータはクロック、割り込みとピンを設定するためのビューも提供します。

5. スマート・コンフィグレータ、コード生成プラグインは一部のデバイスはサポートされていない場合があります。
6. [次へ(N)>] ボタンで次の画面に進みます。

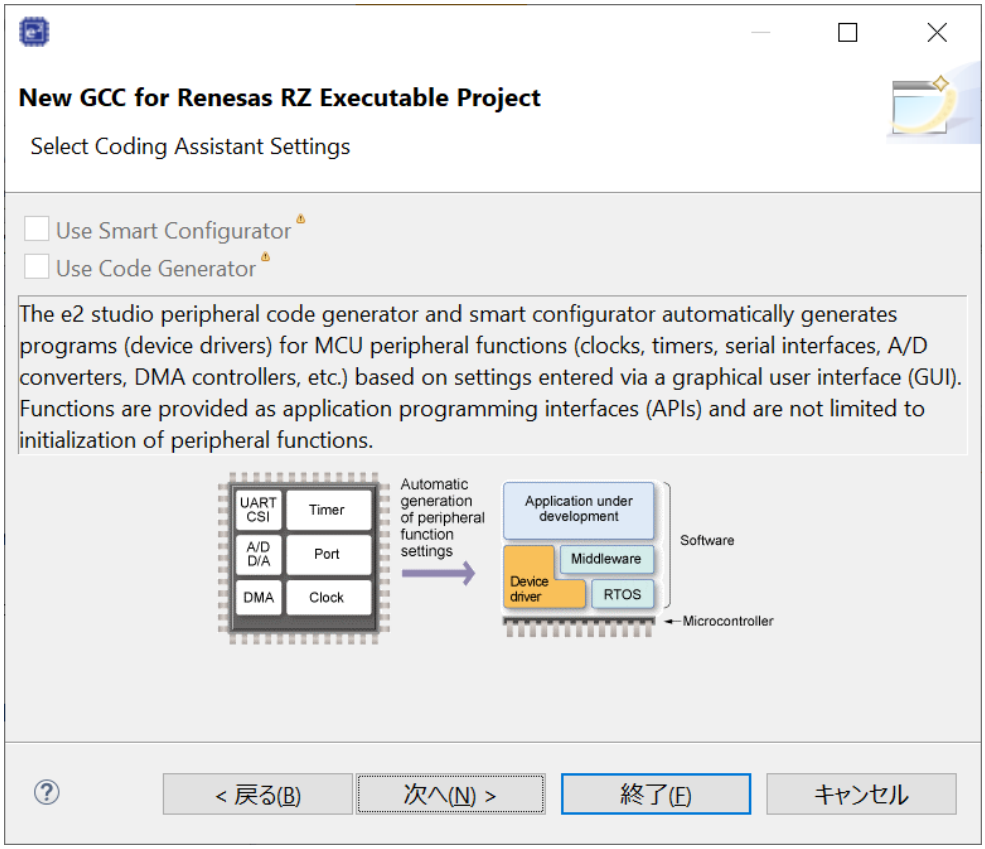

図 3-31　コーディング・アシスタントツールの選択

7. CPU追加オプションはデフォルトの設定を変更せず、[次へ(N)>]ボタンを押してください。

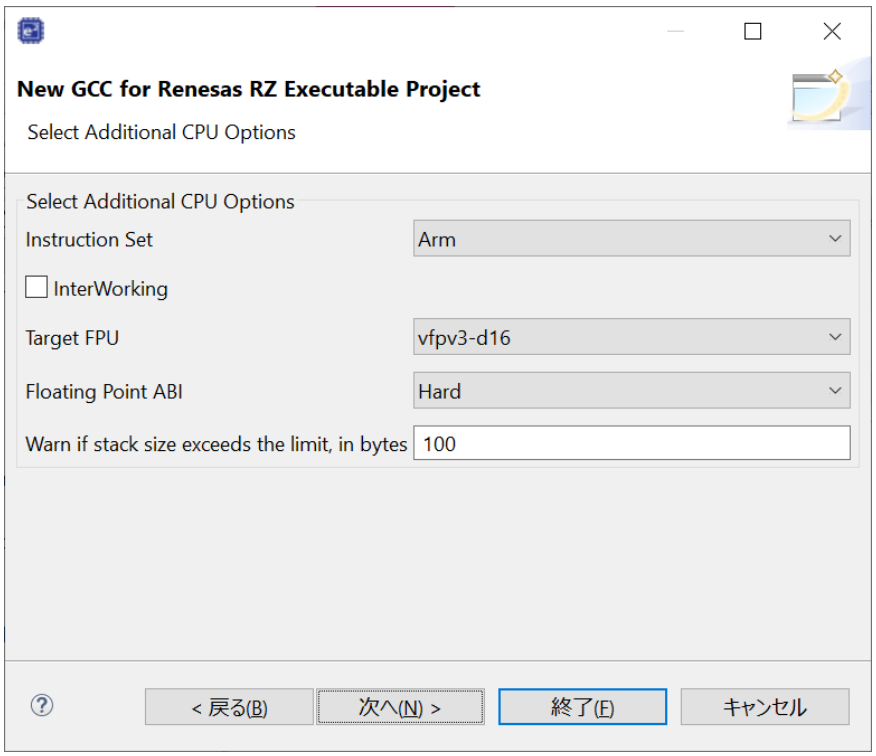

図 3-32 CPU 追加オプションの選択

8. 選択したコンパイラが'Libgen Update for GNU ARM Embedded Toolchains'によって更新されていない場合、ライブラリ・ジェネレータに関する警告メッセージが次のウィザードに表示されます。これは、プロジェクトビルドエラーの原因となる場合があります。2.2.2節を参照し、'Libgen Update for GNU ARM Embedded Toolchains'を実行してください。

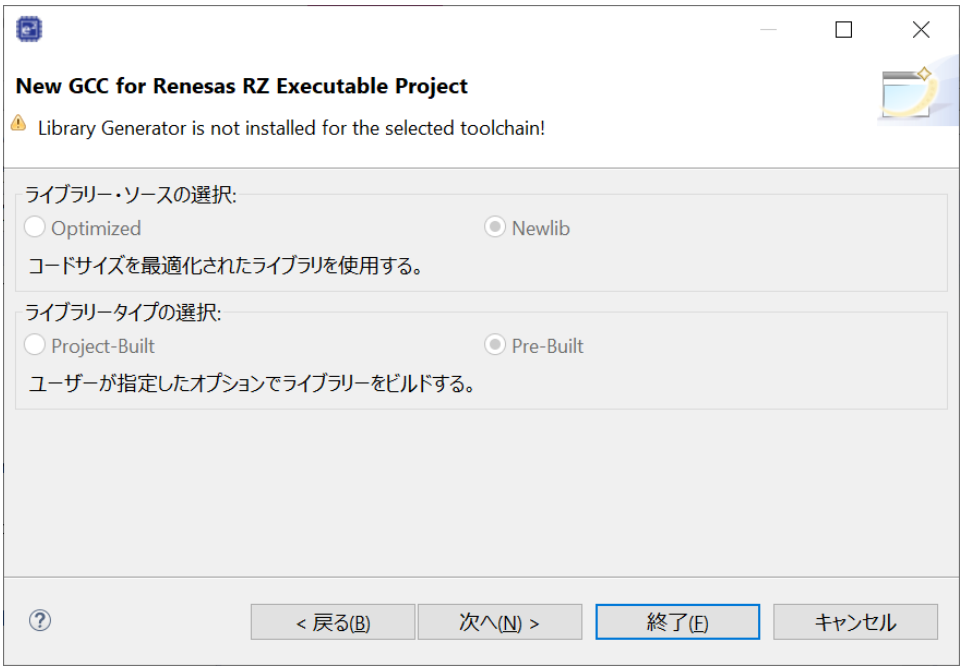

図 3-33 ライブラリ・ジェネレータの設定：警告メッセージ

9. プロジェクトの要約が表示されます。最後に[終了(F)]ボタンを押すとプロジェクトが作成されます。

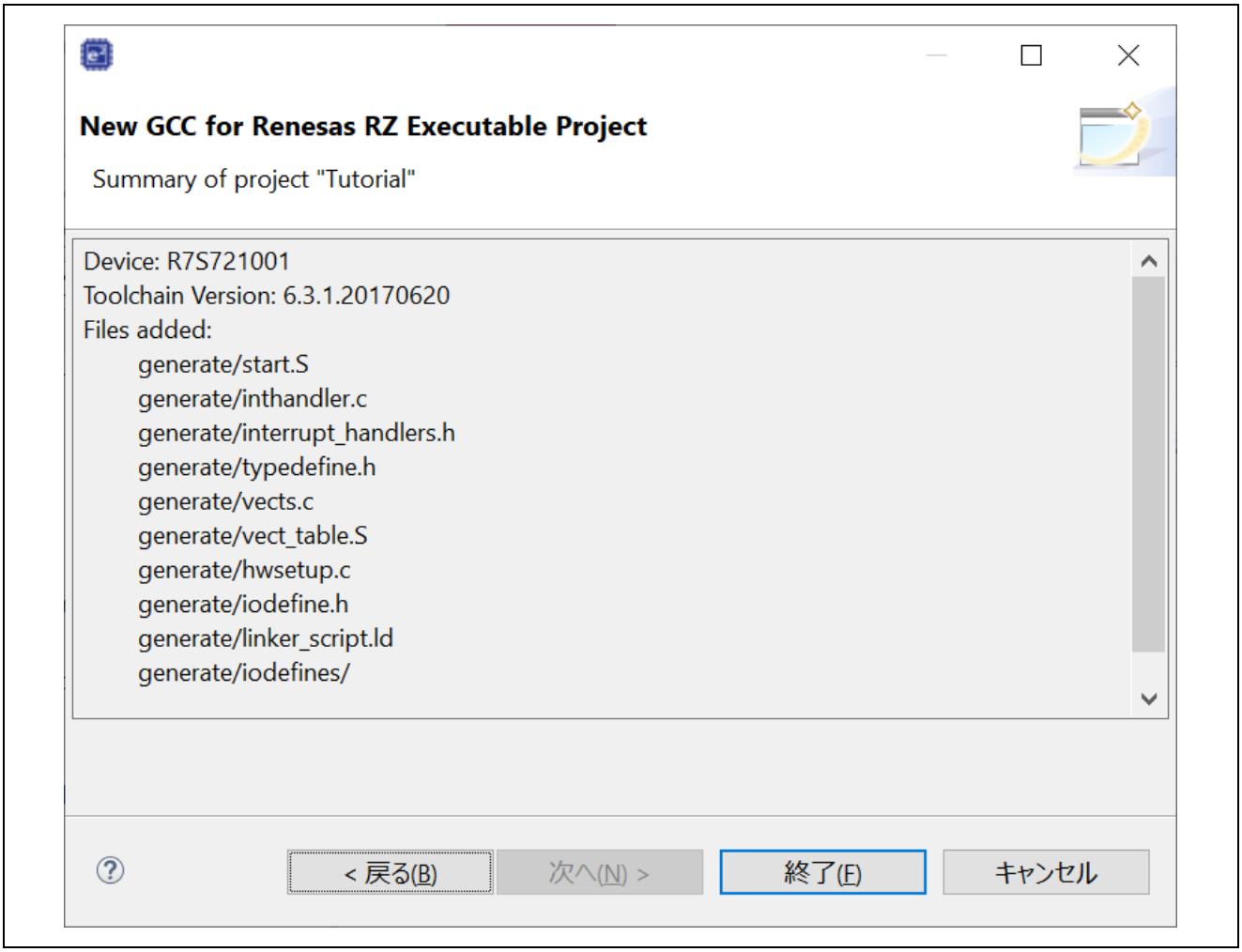

図 3-34 新規プロジェクト作成ウィザード (要約)

10. 「Tutorial」という新しいCプロジェクトが作成されます。

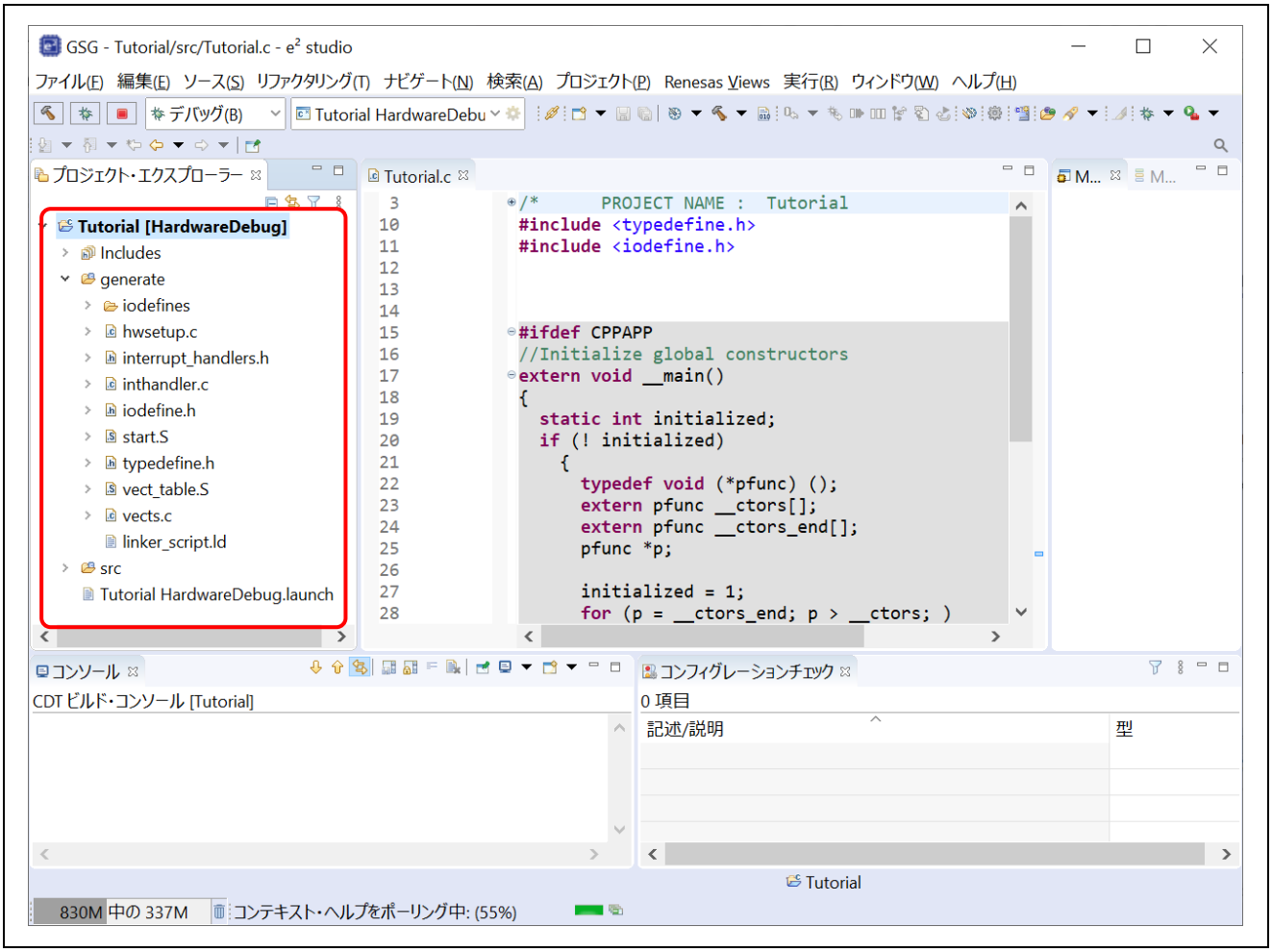

図 3-35 作成された新規 C プロジェクト

このプロジェクトは、“Tutorial.c” というアプリケーションファイルと、自動的に生成されるファイル一式から構成されています。[プロジェクト・エクスプローラー] パネルではすべてのプロジェクトおよびソースファイルを Windows エクスプローラーと同様のフォルダ階層として表示します。

プロジェクトをバックアップする際の注意事項：

- 「.」(ドット)で始まる名前のファイルやフォルダ（.project など）にはプロジェクトの設定情報が含まれますので、バックアップを取る際にはこれらのファイルやフォルダも含めてプロジェクトのフォルダ全体を圧縮するなどしてください。
- 他のプロジェクトのファイルを参照する設定など、プロジェクト間で共有される設定を保存するためには、ワークスペース全体をバックアップする必要があります。

4. ビルド

この章では、e² studio 統合開発環境の主要なビルド機能と設定方法について説明します。

4.1 ビルドオプションの設定

プロジェクト作成時のデフォルトのビルドオプションでビルド・実行する事は可能です。オプションを変更したい場合には、以下で説明する設定画面をお使いください。

1. プロパティ設定画面の起動

   まずプロジェクトのプロパティ設定画面を起動します。e² studio のプロジェクト・エクスプローラで、プロジェクト名の箇所を右クリックしてコンテクスト・メニューの「プロパティ」を選択するか、プロジェクトを選択した状態で「プロジェクト」メニューの「プロパティ」を選択するか、[Alt]+[Enter]を押してください。

   プロジェクト・エクスプローラのどこを選択してプロパティの設定画面を表示したかで設定の及ぶ範囲が変わります。ワークスペース、プロジェクト、フォルダ、個別のソースコードそれぞれのレベルでの設定が可能です。プロジェクトを選択しない状態でプロパティ設定画面を起動するとワークスペース全体のプロジェクトやファイルに対し一括して設定を行うことができます。

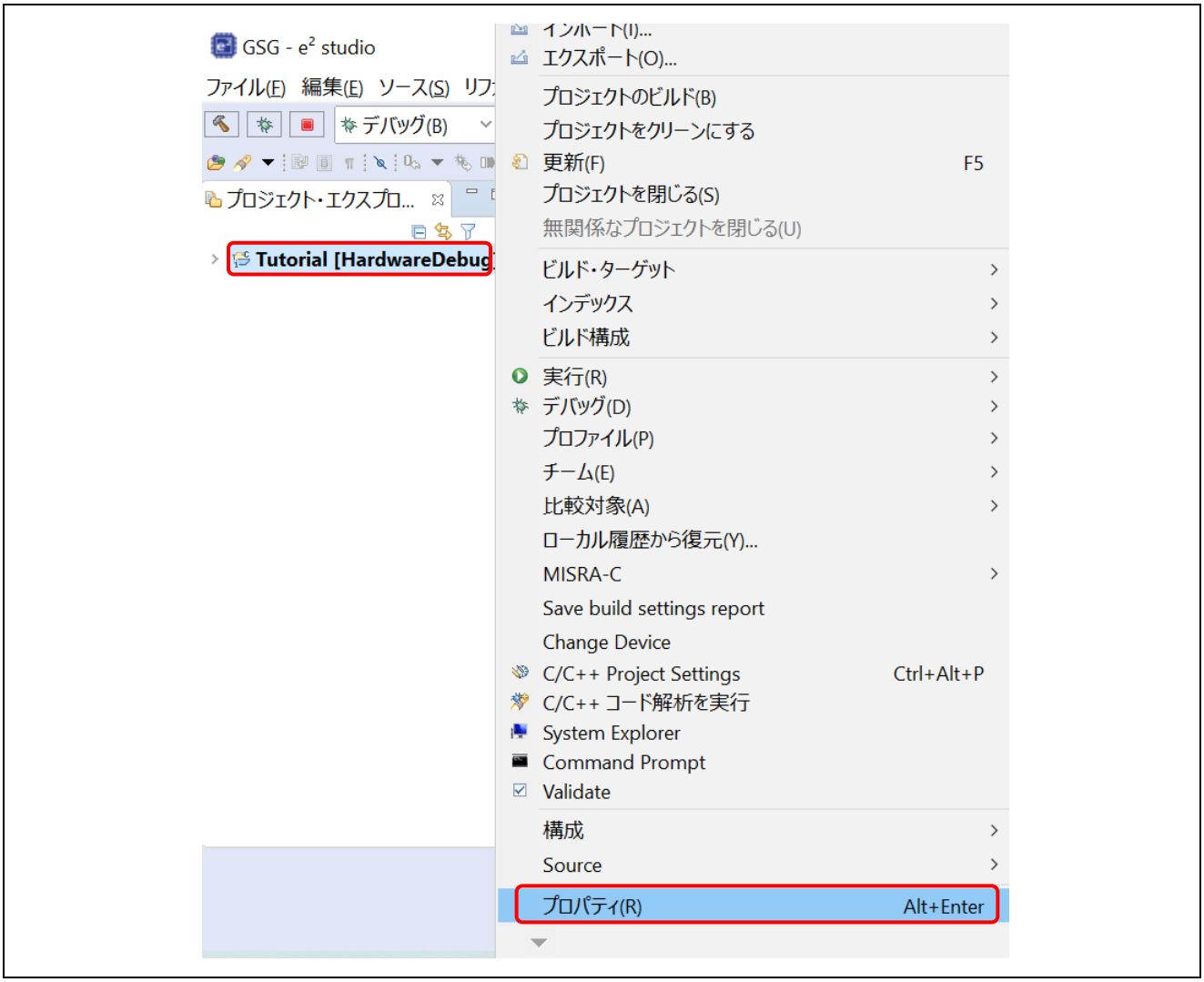

図 4-1 プロジェクトのプロパティ設定画面を起動する

2. ツールチェーン・バージョンの確認

[C/C++ビルド] → [設定] の「Toolchain」タブをクリックすると、ツールチェーンとそのバージョンが表示されます。作成・インポート済のプロジェクトでもここの設定でツールチェーン・バージョンを変更できます。

ツールチェーン管理画面で有効にしたバージョンが選択肢として表示されます。

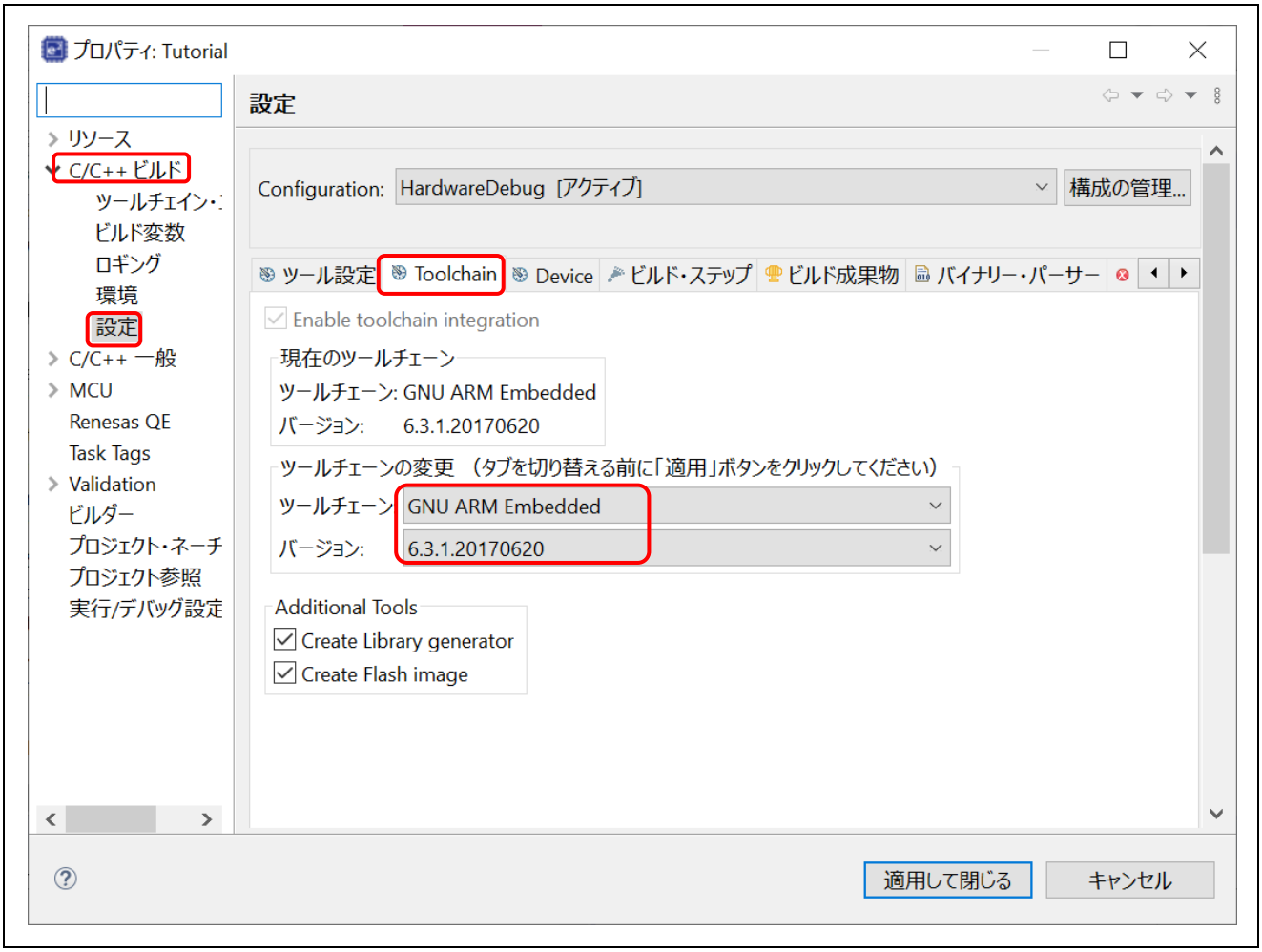

図 4-2 ツールチェーン・バージョンの確認・変更画面

3. ビルド環境変数

環境変数は e² studio を経由してビルドコマンドに引き渡されます。環境変数の内容を確認するには、[C/C++ ビルド] → [環境]を選択してください。

プロジェクトで独自の環境変数を設定することも可能です(下記の例ではテンポラリフォルダを指定)。

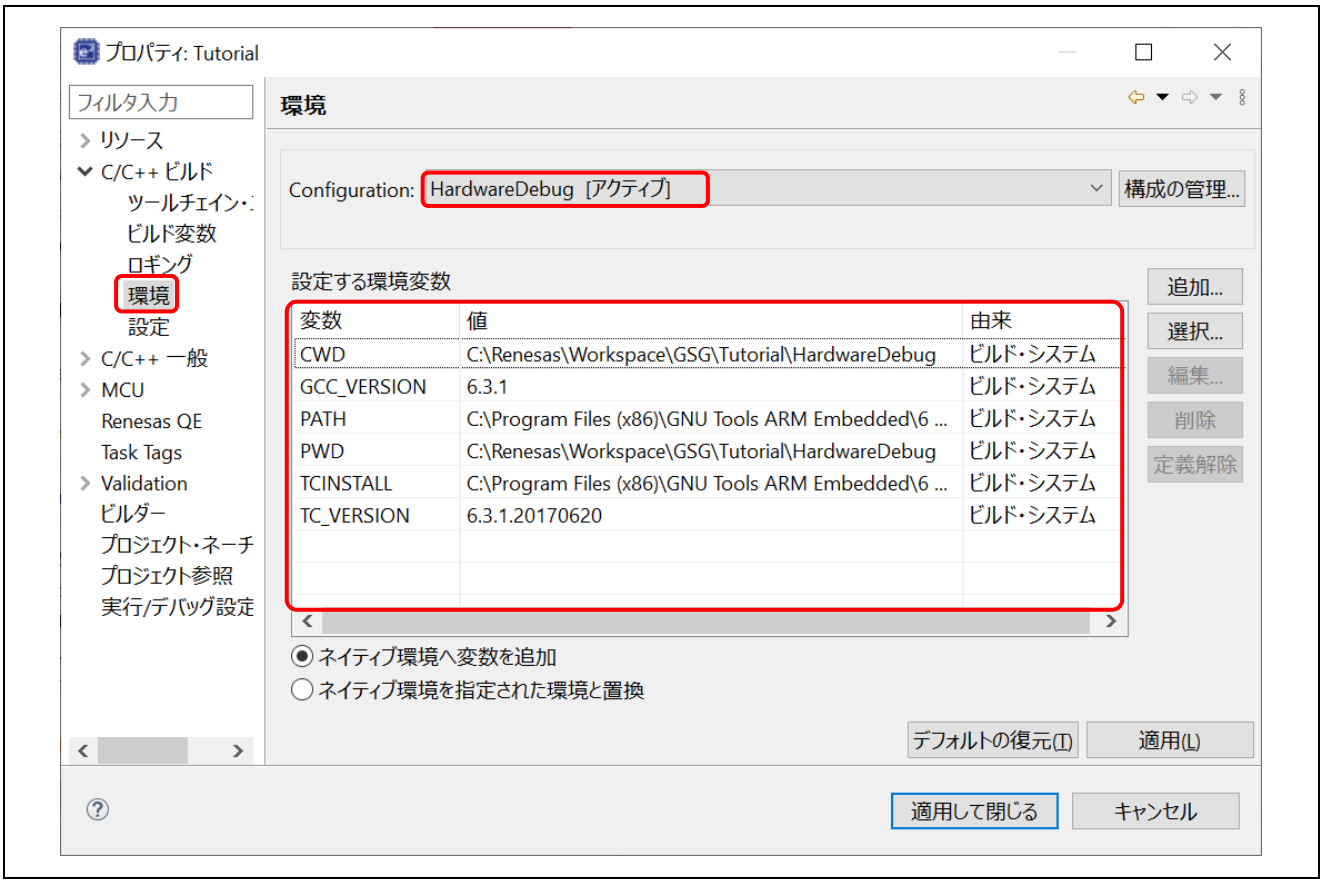

図 4-3 環境変数設定画面

4. ビルドオプションの設定

ビルドオプションによって、パス名を含むすべてのツールチェーン設定を維持することができます。図 4-3 に示すとおり、現在のビルド構成は、「HardwareDebug [アクティブ]」です。

ビルドオプションの詳細は、各コンパイラ製品のユーザーズマニュアル(ヘルプ)を参照してください。
インストールディレクトリの doc フォルダ内にあります。
(例：C:¥Program Files (x86)¥GNU Tools ARM Embedded¥4.9 2015q3¥share¥doc)

注： 上記画面とは別に「Tool chain エディター」設定画面もありますが、これは Renesas 製ビルドプラグインがサポートしていないコンパイラ製品で設定が必要になるものです。通常は使用しないでください。

4.2 プロジェクトのビルド

以下いずれかの方法でプロジェクトがビルドできます。

1. プロジェクト・エクスプローラでプロジェクトを右クリックして「プロジェクトのビルド」を選択する。
2. プロジェクト・エクスプローラでプロジェクトを選択状態にし、プロジェクトメニューの「プロジェクトのビルド」を選択する。
3. 2.でプロジェクトメニューの代わりに「ファイル」ツールバーの ボタンを押す。
4. 2.でプロジェクトメニューの代わりに、Ctrl+Bを押す。

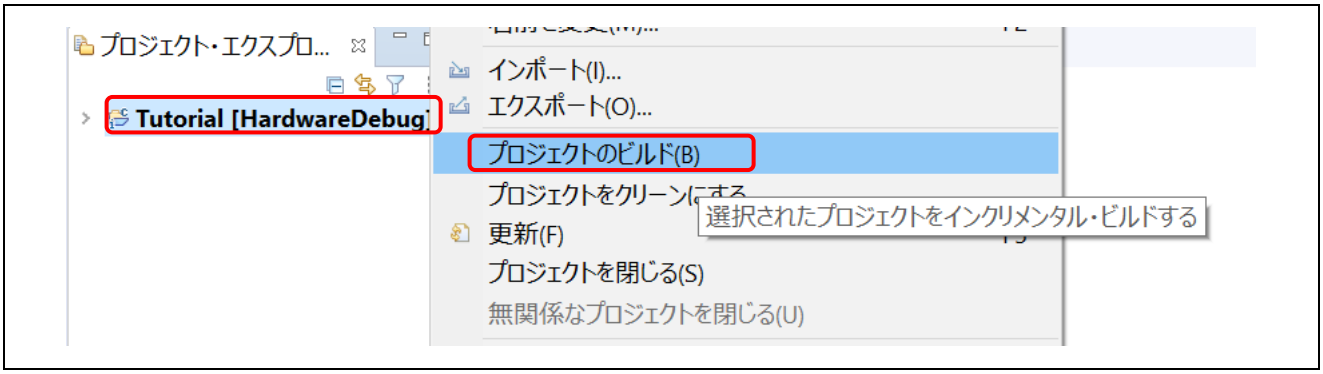

図 4-4 プロジェクトメニューによるビルド

5. ビルドが成功すると、[コンソール]に “Build Finished.” というメッセージが表示されます。ビルドの最後に${CWD}ディレクトリに出力されるファイルは、“makefile”、“<プロジェクト名>.elf”、“<プロジェクト名>.map”、“<プロジェクト名>.hex” で構成されます。
6. “<プロジェクト名>.elf” は、デバッグに使用するELF/DWARFフォーマットの標準ロードモジュールです。

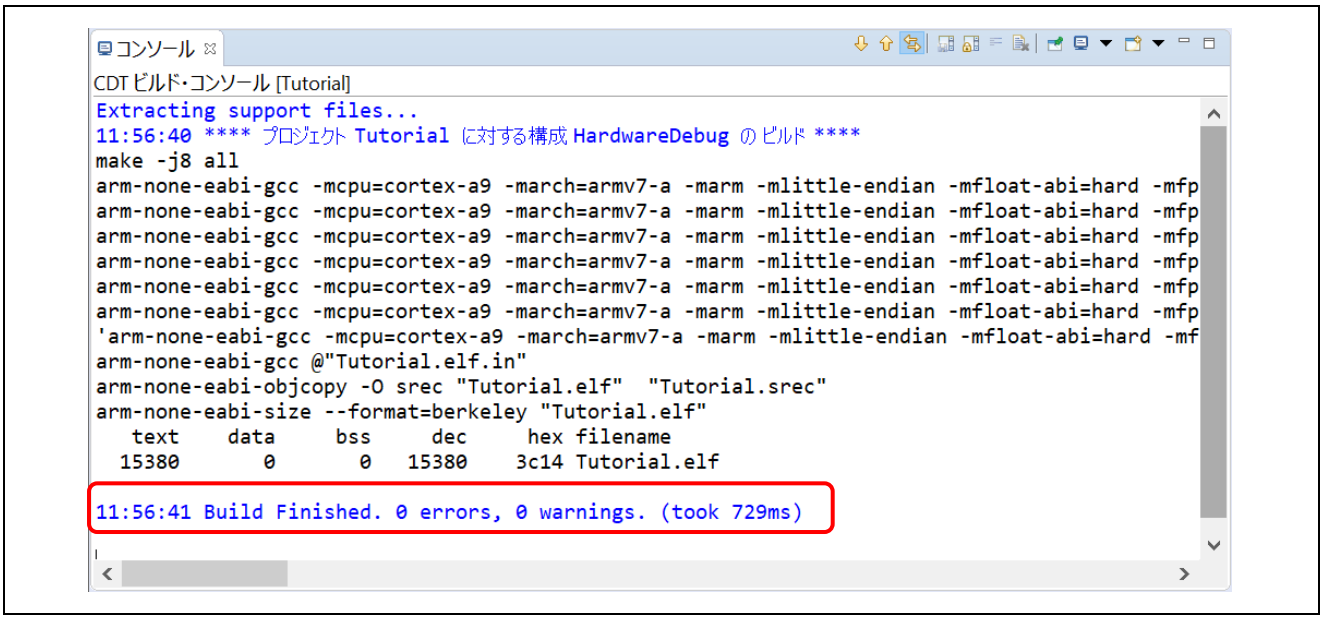

図 4-5 ビルドコンソールの表示例 (ビルド成功時)

4.3 ビルド構成のレポート

プロジェクトレポート機能により、プロジェクトとビルド構成を e² studio からファイルに出力して、プロジェクトやビルド環境の設定を容易にチェックし、比較することができます。

1. [プロジェクト・エクスプローラー] を右クリックしてコンテクスト・メニューを開きます。
2. [Save build settings report] を選択してビルド構成レポートを保存します。

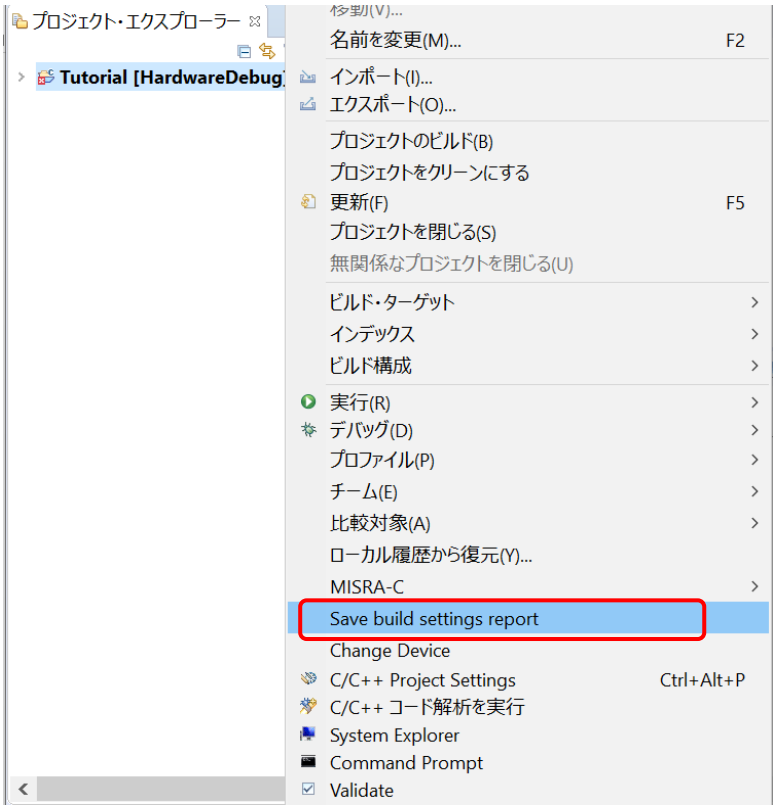

図 4-6 プロジェクトレポート

5. デバッグ

この章では、e² studio 統合開発環境のデバッグ構成と主要なデバッグ機能の使い方について説明します。以下では、Segger J-Link エミュレータと RSK RZ/A1H ボードを動作環境とする “Tutorial” プロジェクト(4.2 節)を例に説明します。

デバッグに使用する特定のパースペクティブを開くには、e² studio の “Tutorial” プロジェクトワークスペースを開き、[デバッグ] パースペクティブをクリックします。

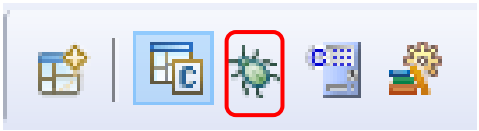

図 5-1 [デバッグ] パースペクティブへの切り替え

パースペクティブでワークベンチのウィンドウのレイアウト表示(開発ツール関連)を定義します。それぞれのパースペクティブは、ビュー、メニュー、ツールバーの組み合わせで構成され、それによりユーザは特定のタスクを実行できます。

例えば、[C/C++] パースペクティブは、ユーザが C/C++プログラム開発を行うために必要なビューを持ち、[デバッグ] パースペクティブは、プログラムのデバッグに必要なビュー表示をすることができます。ユーザが [C/C++] パースペクティブ表示中にデバッガに接続しようとする場合、e² studio は [デバッグ] パースペクティブに切り替えるようユーザを促します。

一つのワークベンチのウィンドウのなかに、一つまたは複数のパースペクティブを表示することができます。ユーザはそれらをカスタマイズしたり、新しいパースペクティブを追加することができます。

注意： デバッグについての詳細は、e² studio メニューの
[ヘルプ] → [ヘルプ目次] → [e2 studio ユーザーガイド]
→ [デバッグに関する機能]
を参照してください。

5.1 既存デバッグ構成の変更

初めてデバッグを行う際には、一度だけデバッグ構成を設定する必要があります。既存のデバッグ構成は以下のように変更できます。

1. [プロジェクト・エクスプローラー] の “Tutorial” プロジェクトをクリックします。
2. [実行] → [デバッグの構成...] あるいは アイコン(下向き矢印) → [デバッグの構成] の順にクリックし、[デバッグ構成] ウィンドウを開きます。

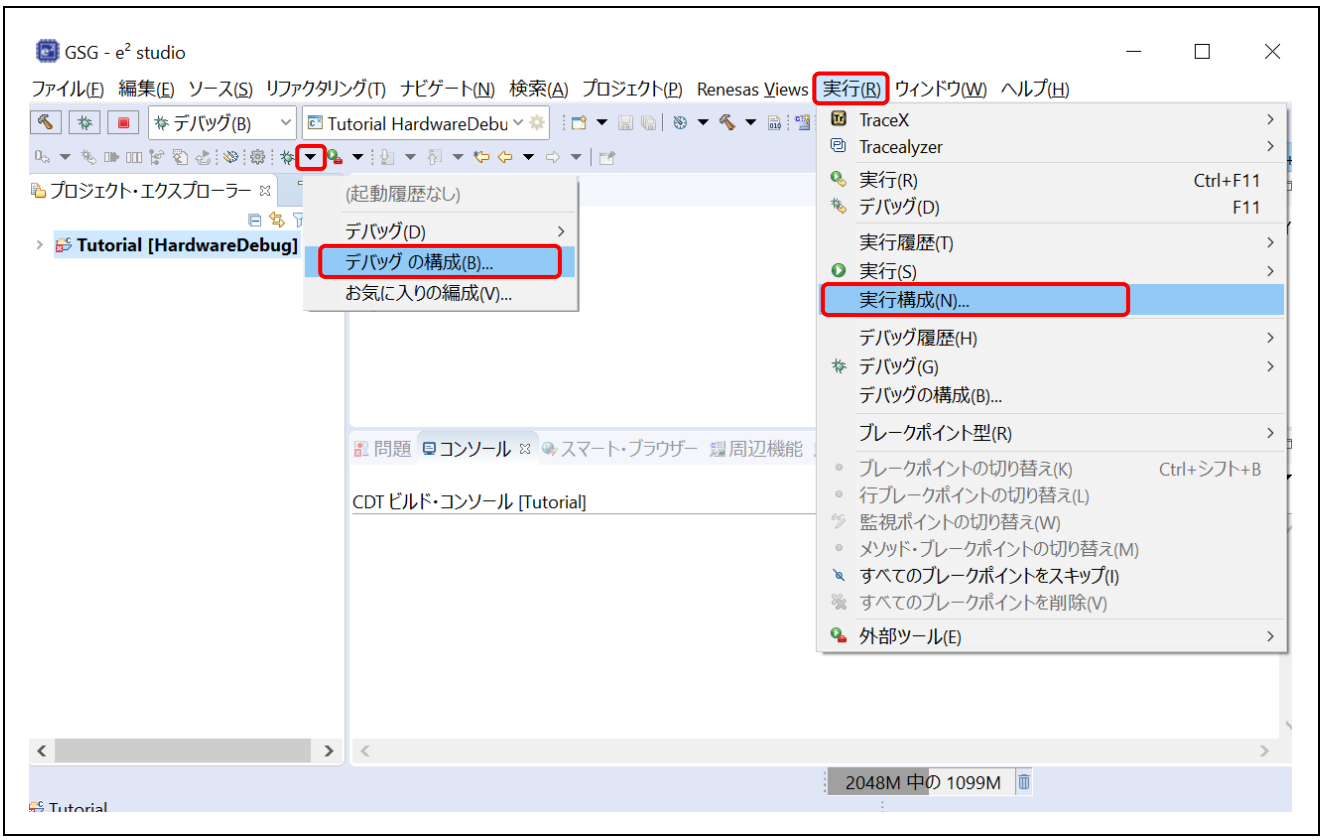

図 5-2 [デバッグ構成]ウィンドウを開く

3. [デバッグ構成] ウィンドウで、“Renesas GDB Hardware Debugging” デバッグ構成の表示を展開し、既存のデバッグ構成をクリックしてください（例：“Tutorial HardwareDebug”）。[メイン] タブを選択し、デバッグ対象のロードモジュールが指定されているかを確認してください(例： “Tutorial.elf”)。

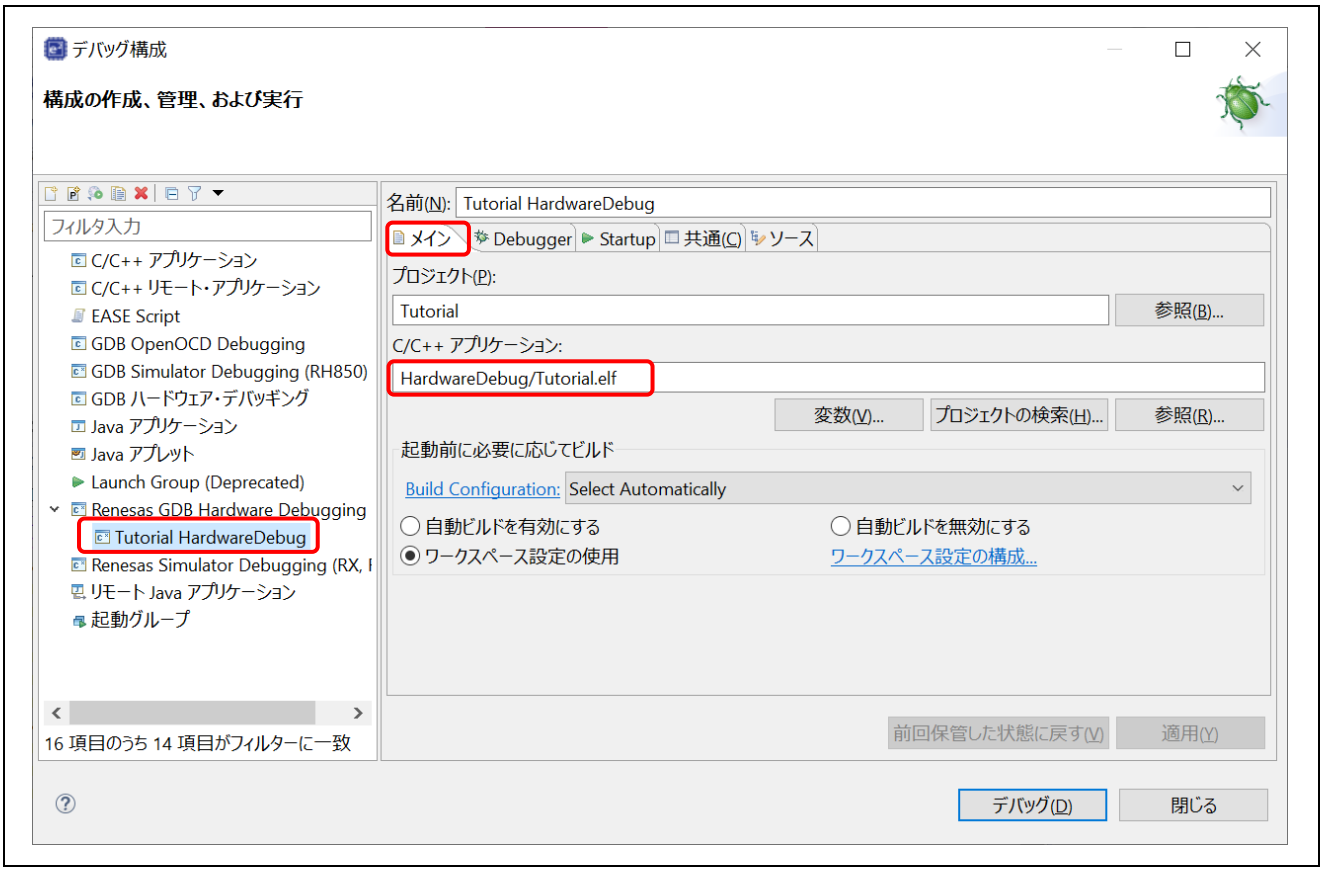

図 5-3　　ロードモジュールの選択

4. [Debugger] タブに切り替え、エミュレータの種類とデバイス名を選択してください。この例ではDebug hardwareに “J-Link ARM”、Target Deviceに “R7S721001” を設定しています。

図 5-4　　ターゲットデバイスの選択

5. [Debugger] タブの中の [Connection Settings] サブタブでは、Segger J-Linkエミュレータ、および RSK RZ/A1Hボードの設定を元に、以下を設定します。

- Interface
  - Type = “SWD”
  - Speed = “Auto”

注意: 以上の設定内容は例として示したものです。誤った設定は誤作動やハードウェアの故障の原因となりますので、接続の前に設定内容がボードの仕様と合っているかを慎重にご確認ください。

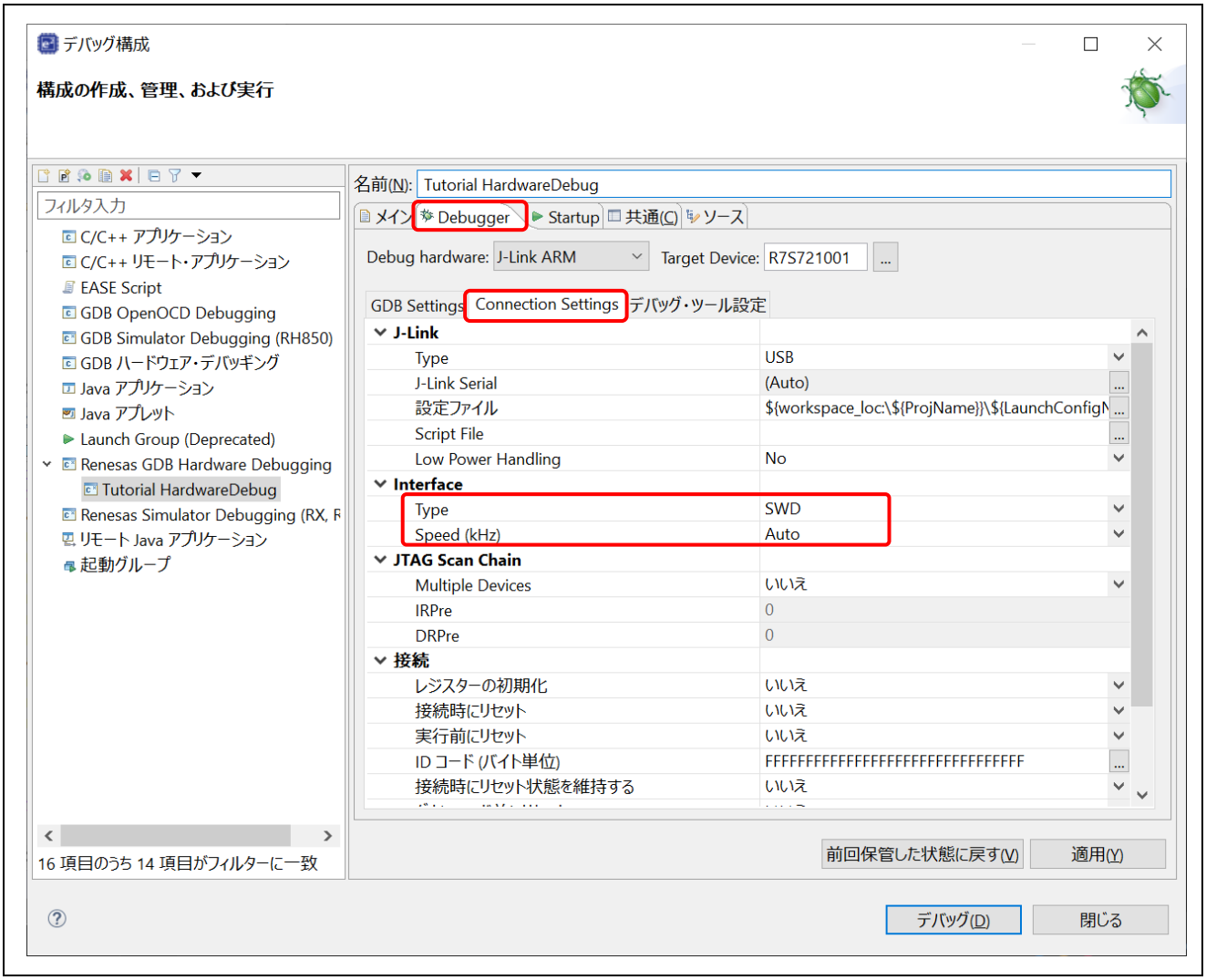

図 5-5 接続設定の変更

6. サブタブ [デバッグ・ツール設定] では、RSK RZ/A1H ボードに基づいて、以下を設定してください。
   - メモリー
     - エンディアン ="リトル・エンディアン"

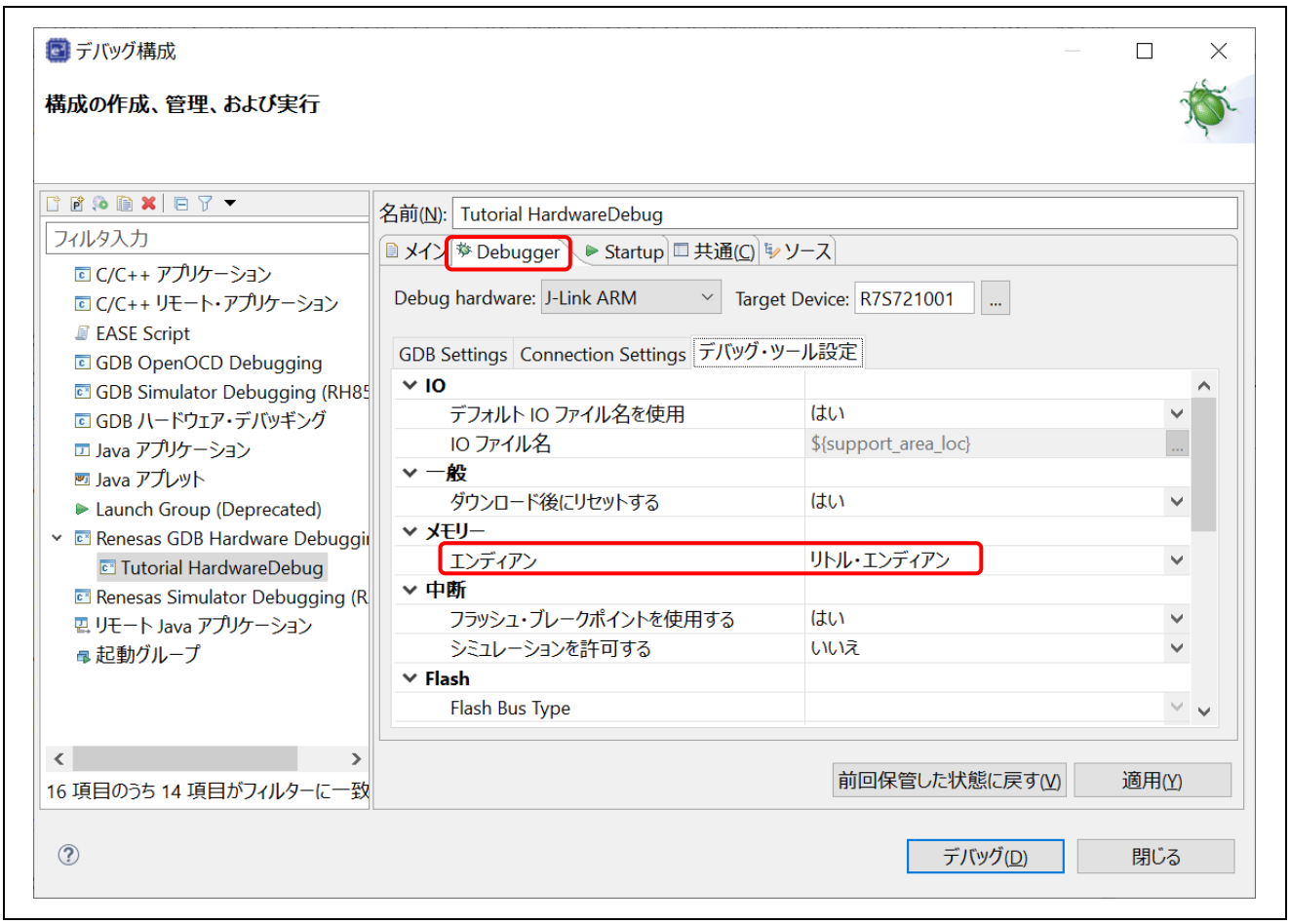

図 5-6 デバッグ・ツール設定の変更

7. [適用] ボタンを押すと設定が保存されます。[デバッグ] ボタンを押すと、デバッガが起動し、Segger J-Link エミュレータ、および RSK RZ/A1H ボードへの接続が開始されます。
8. 正しく接続できた場合は、図に示すような [Debug] ビュー画面を選択します。接続後はエントリポイント(この例では "start.S" の "PowerON_Reset()")でプログラムの実行が一旦中断されます。

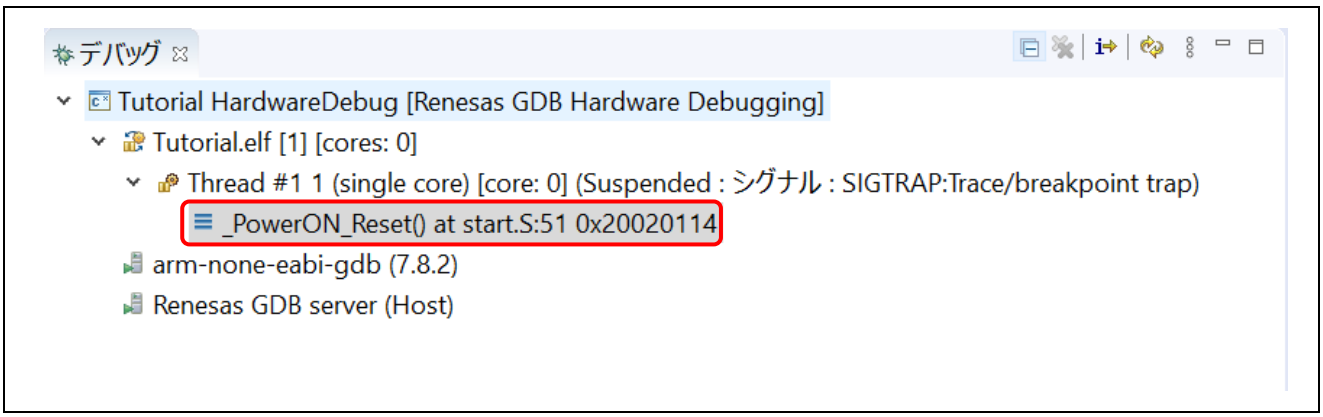

図 5-7 [Debug] ビューのユーザターゲット接続

5.2 新規デバッグ構成の作成

別な種類のエミュレータを使用したいなどで、デバッグ構成を追加する場合に、簡単な方法は既存の構成を複製する方法です。以下の手順で行います。

1. [プロジェクト・エクスプローラー] の “Tutorial” プロジェクトをクリックします。[デバッグ構成] ウィンドウを開きます(呼び出し方は「5.1. 既存デバッグ構成の変更」参照)。
2. [デバッグ構成] ウィンドウで、デバッグ構成 (例: “Tutorial HardwareDebug”) を選択し アイコンをクリックします (現在選択している起動構成をコピーします)。新規デバッグ起動構成 (例: “Tutorial HardwareDebug (1)”) が作成されます。設定を確認するために、“名前” テキストボックスに別の名前を入力して [適用] ボタンをクリックすることもできます。

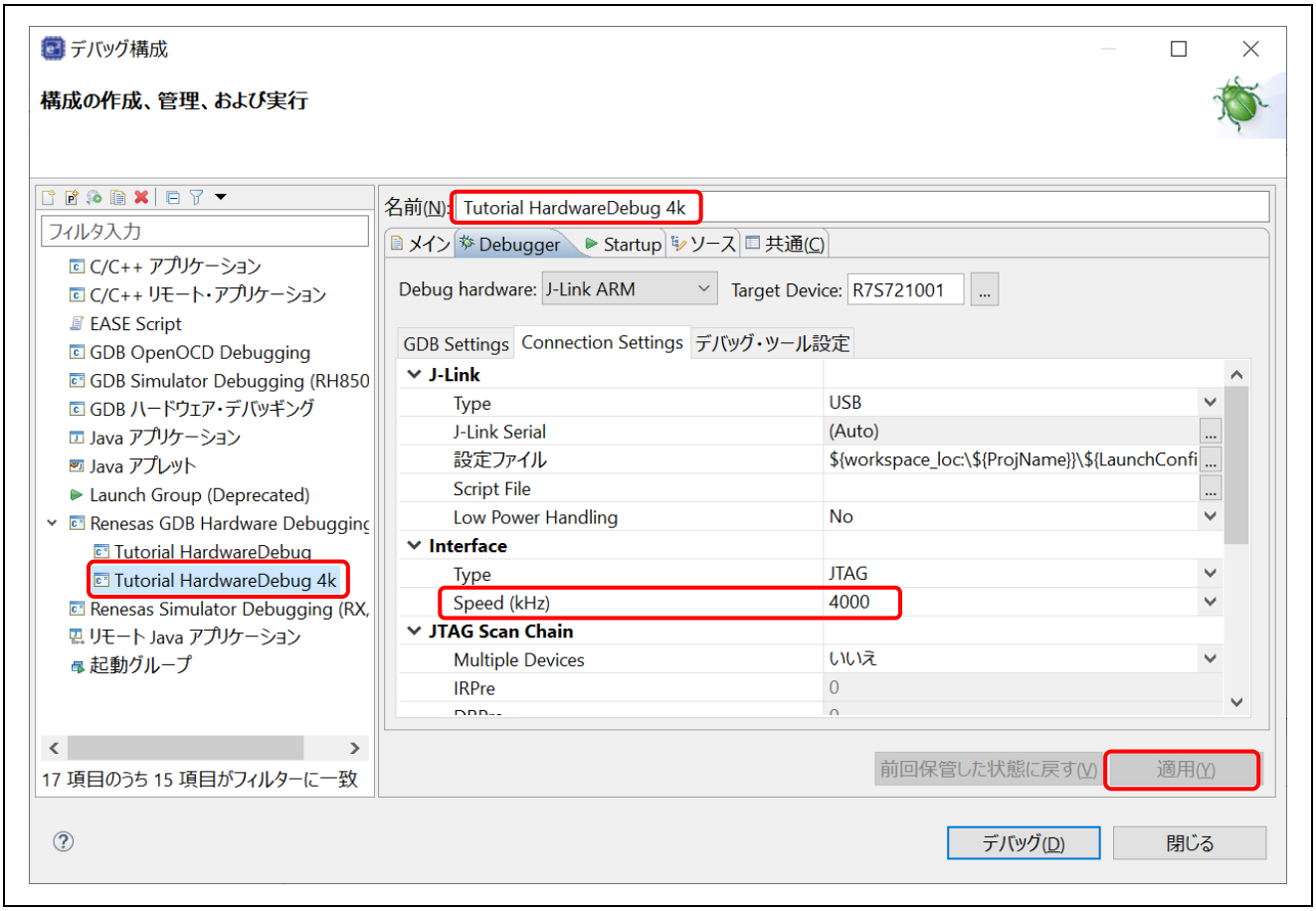

図 5-8 選択したデバッグ起動構成の複製

3. デバッグ構成は、5.1節で述べたように作成されます。例えば、インタフェースの “Speed” を “4000” に変更します。

4. “[local]” の付いた構成を起動してください。“*” 印は、その構成がまだどのプロジェクトにも適用されていないことを示します。[共通] タブでプロジェクト名を指定してください。

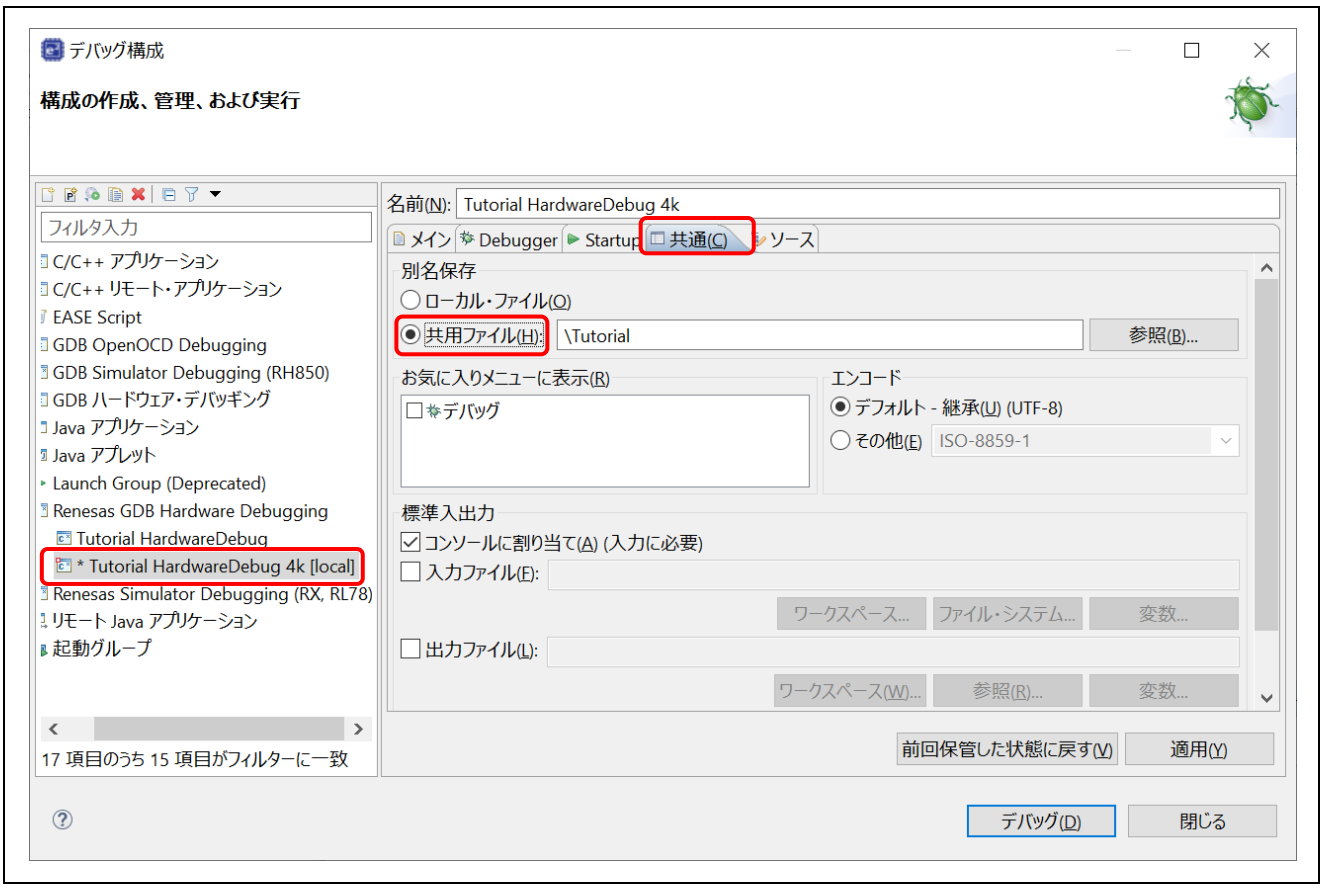

図 5-9 特定プロジェクトへの起動構成の適用

5.3 Launch Bar

本節では、V6.0.0 以降で追加された Launch Bar(下図のツールバー)の使い方を説明します。
Launch Bar は e² studio メインウィンドウのツールバーエリア内に表示されます。
この簡単なインタフェースは、デバッガで起動対象とするターゲットに対して、ビルドボタンやデバッグボタンが用意されています。

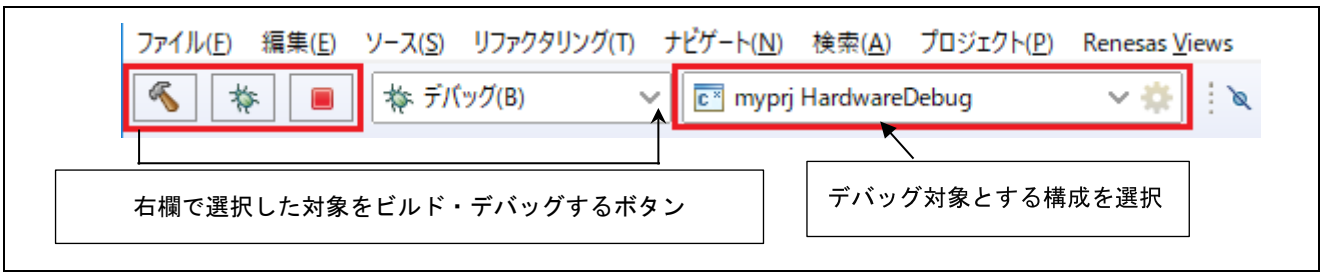

図 5-10 Launch Bar の表示例

Launch Bar は以下の動作を行います：

- ビルド操作：ボタンを押すと、右欄で選択されたデバッグ対象のロードモジュールをビルドします。

注意： 「プロジェクト管理」ツールバーにも同様のボタンがありますが、そちらはプロジェクト・エクスプローラで選択およびアクティブ状態にしたビルド構成に対してビルド操作を行うものです。状況によっては Launch Bar のビルドボタンを押した時とはビルドされるものが異なる場合がありますのでご注意ください。

- デバッグ操作： ボタンでデバッグの開始／終了(ブレークではなくデバッグセッションの終了)を行います。

Launch Bar の機能が不要な場合は、「ウィンドウ」メニュー → 「設定」→ 「実行/デバッグ」→ 「起動中」の「Launch Bar」の設定でツールバーやボタンを非表示にすることができます。

5.4 基本的なデバッグ機能

本節では、e² studio がサポートする典型的なデバッグビューを説明します。

- 標準的な GDB デバッグ (Eclipse フレームワークによってサポートされている) : ブレークポイント、式、レジスタ、メモリ、逆アセンブル、変数
- 標準的な GDB デバッグの拡張 : イベントポイント、IO レジスタ、トレース

[デバッグ・ツールバー] を開くには、プルダウンメニューボタンをクリックして [デバッグ・ツールバーを表示] を選択します。以下に示すのは、[デバッグ] ビューの便利なツールバーです。

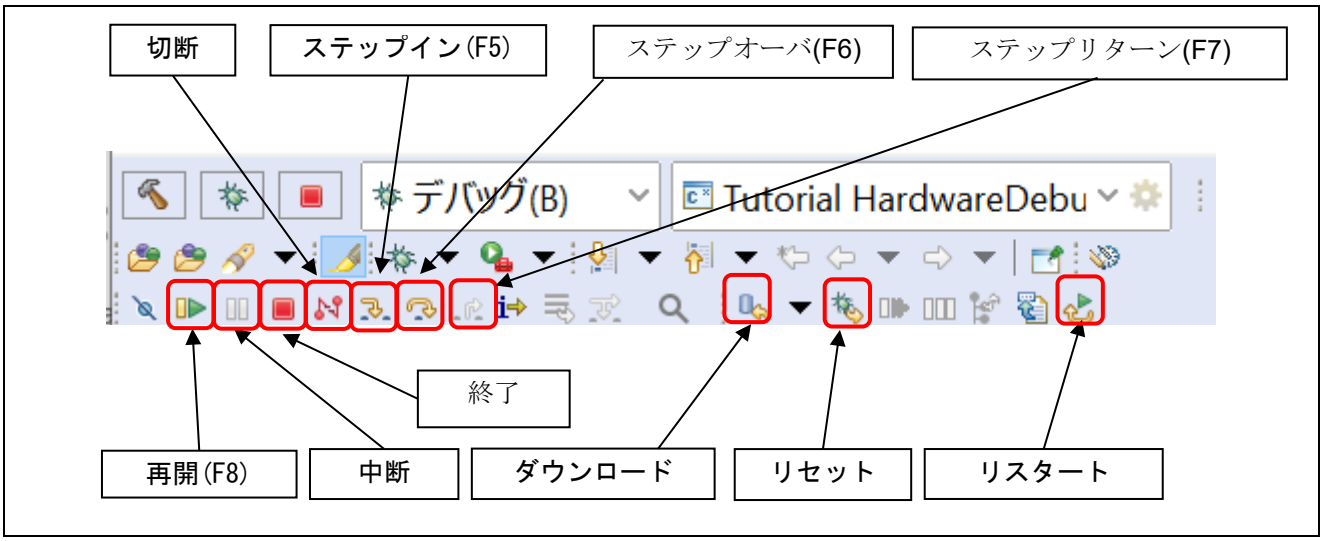

図 5-11 デバッグビューの便利なツールバー

プログラムを実行するには ボタンをクリックするか [F8] キーを入力します。

プログラムは、ブレークポイントで、あるいは ボタンをクリックすることで一時停止します。一時停止中は以下の操作が可能です。

- ボタンまたは [F5] キーは、現在実行中のプログラム行にある次の関数呼び出しへステップイン実行します（関数内に入って 1 ステップ実行）。
- ボタンまたは [F6] キーは、現在実行中のプログラム行にある次の関数呼び出しをステップオーバ実行します（1 行実行するが関数内には入らない）。
- ボタンで、実行を再開します。

デバッグセッションの停止は、選択したデバッグセッション／プロセスを ボタンで終了するか、選択したプロセスとデバッガを ボタンで切断します。

他に以下のような操作が可能です。

- ボタンは、エントリポイントからプログラムを再実行します（ → の順にクリックするのと同じ）。
- ボタンは、プログラムをパワーオンリセットのエントリポイントにリセットします。
- ボタンは、ターゲットシステムにバイナリファイルを再びダウンロードします。

注意： 以降の章で機能について説明するため、ルネサスのウェブサイトから、RZ/A1Hのサンプルコードをダウンロードしてください。

1. ルネサスのウェブサイトから、Starter Kit Sample Code for RZ/A1H をダウンロードします。
   https://www.renesas.com/sg/en/software/D6000665.html.
2. 第3.1章を参照しインポートしてワークスペースに「RZ_A1H_Tutorial_RSK」プロジェクトをインポートしてください。
3. プロジェクトのプロパティを開き、画面の左側で、[C/C++ ビルド] → [設定] を選択します。[Toolchain] タブで、プロジェクトの GNU ARM Embedded toolchain を選択します。[Apply and Close] をクリックしてください。

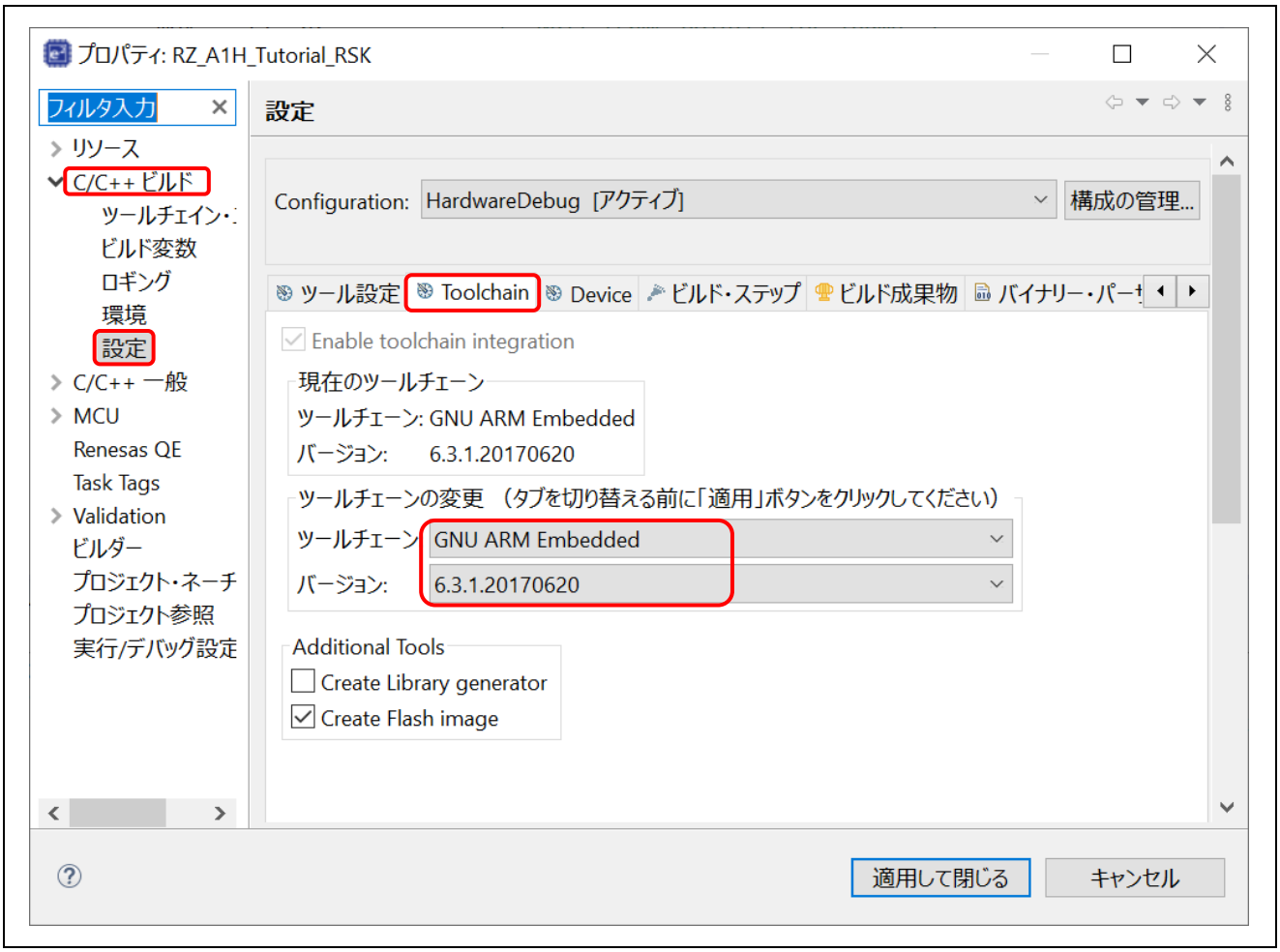

図 5-12 プロジェクトツールチェーンの更新

4. [Device] タブに切り替えて、使用中のデバイスがボードに合っているかを確認してください。合っていない場合は、インポートしたプロジェクトを右クリックして “Change Device” を選択し、正しいデバイスを選択します。

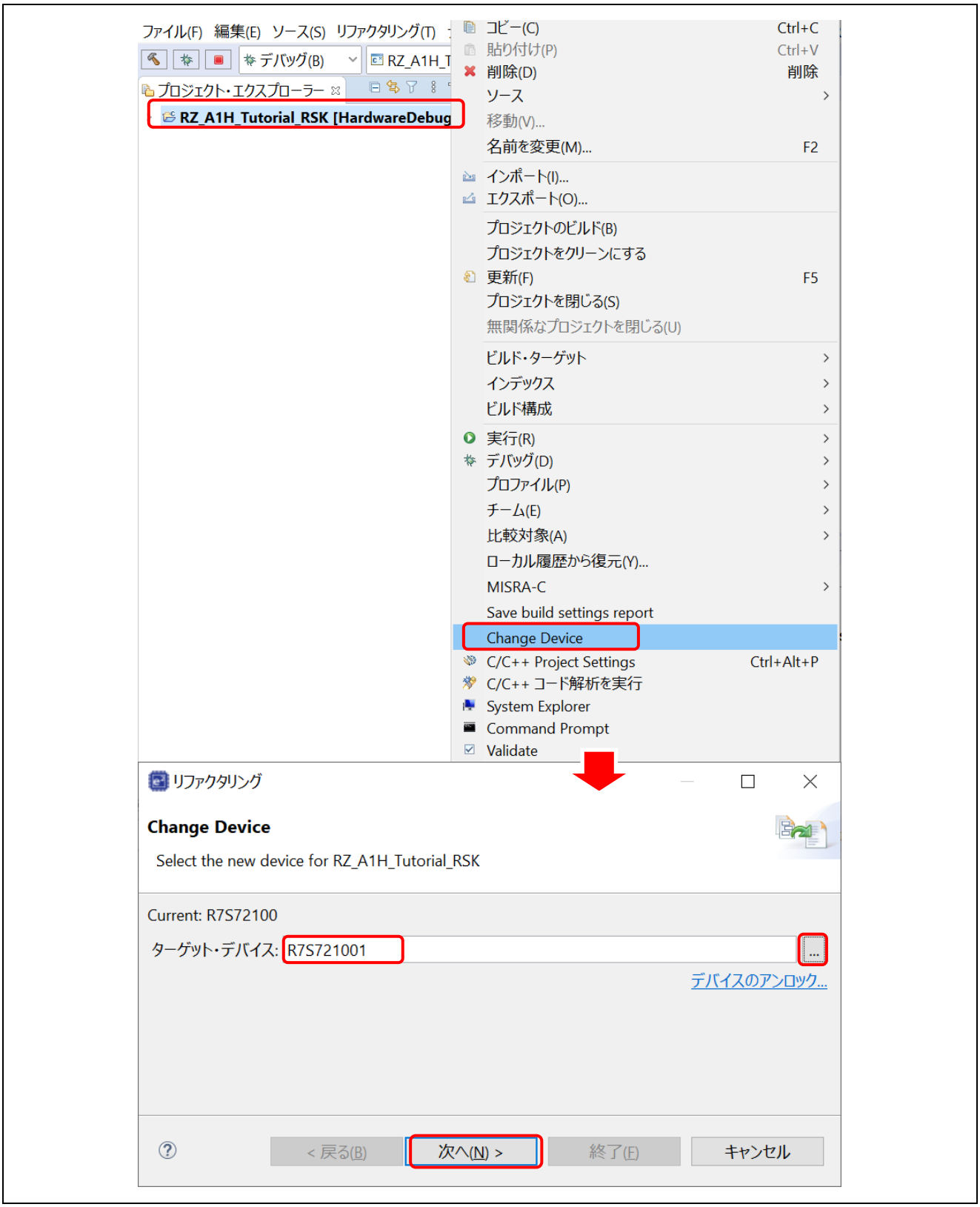

図 5-13 デバイスの変更

5. 新規デバイスを選択し[次へ]をクリックするとファイルが変更、または再生成されていることを確認します。

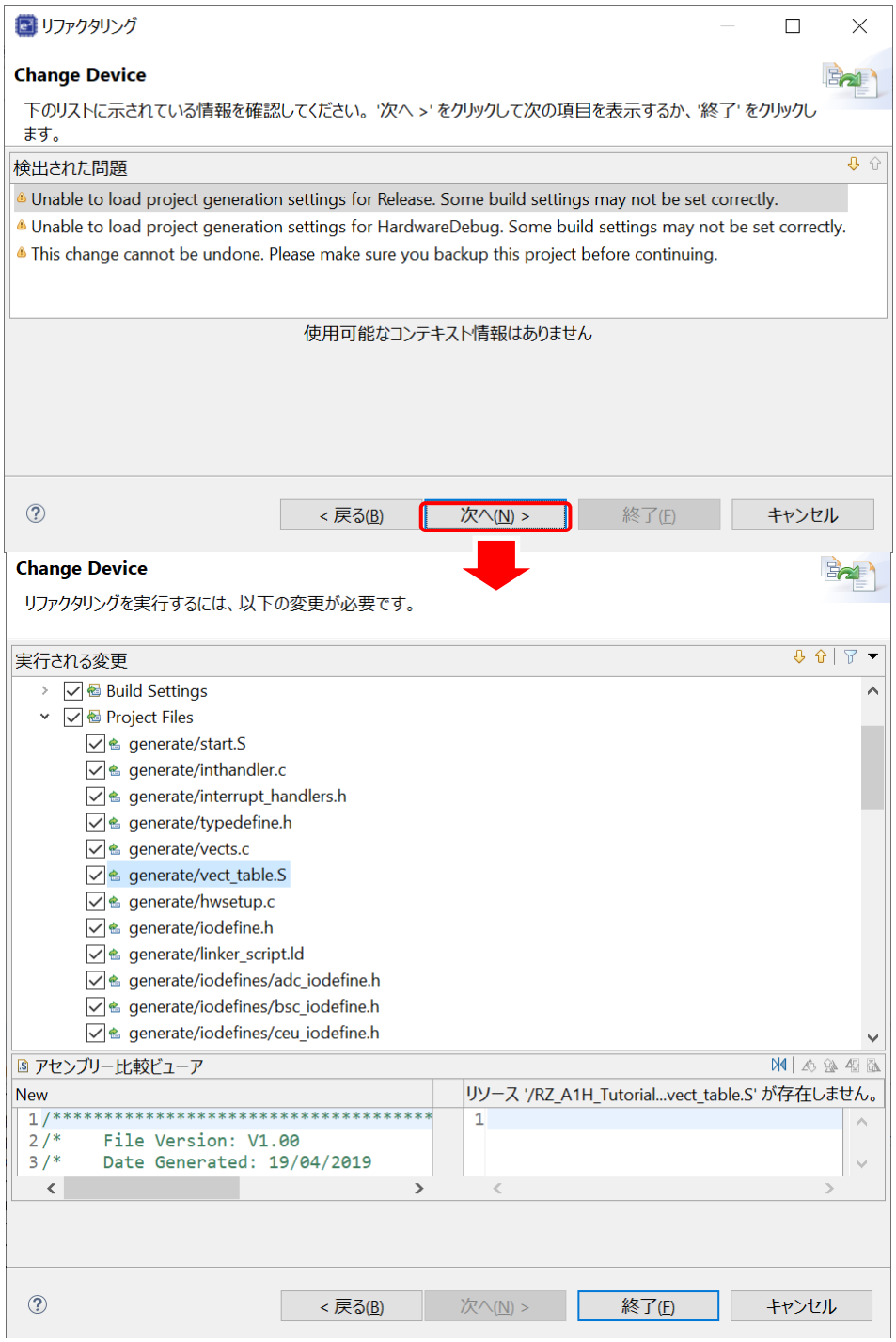

図 5-14 デバイス変更後に再生成した（変更した）ファイルの比較

6. プロジェクトのプロパティを開き、画面の左側で、[C/C++ ビルド] → [設定] を選択します。[ツール設定] タブに切り替えて、[Cross ARM C Linker] → [General] を選択し、コンパイラが生成したスクリプトファイルを、スクリプトファイルの一覧に追加します。[Apply and Close] をクリックしてください。

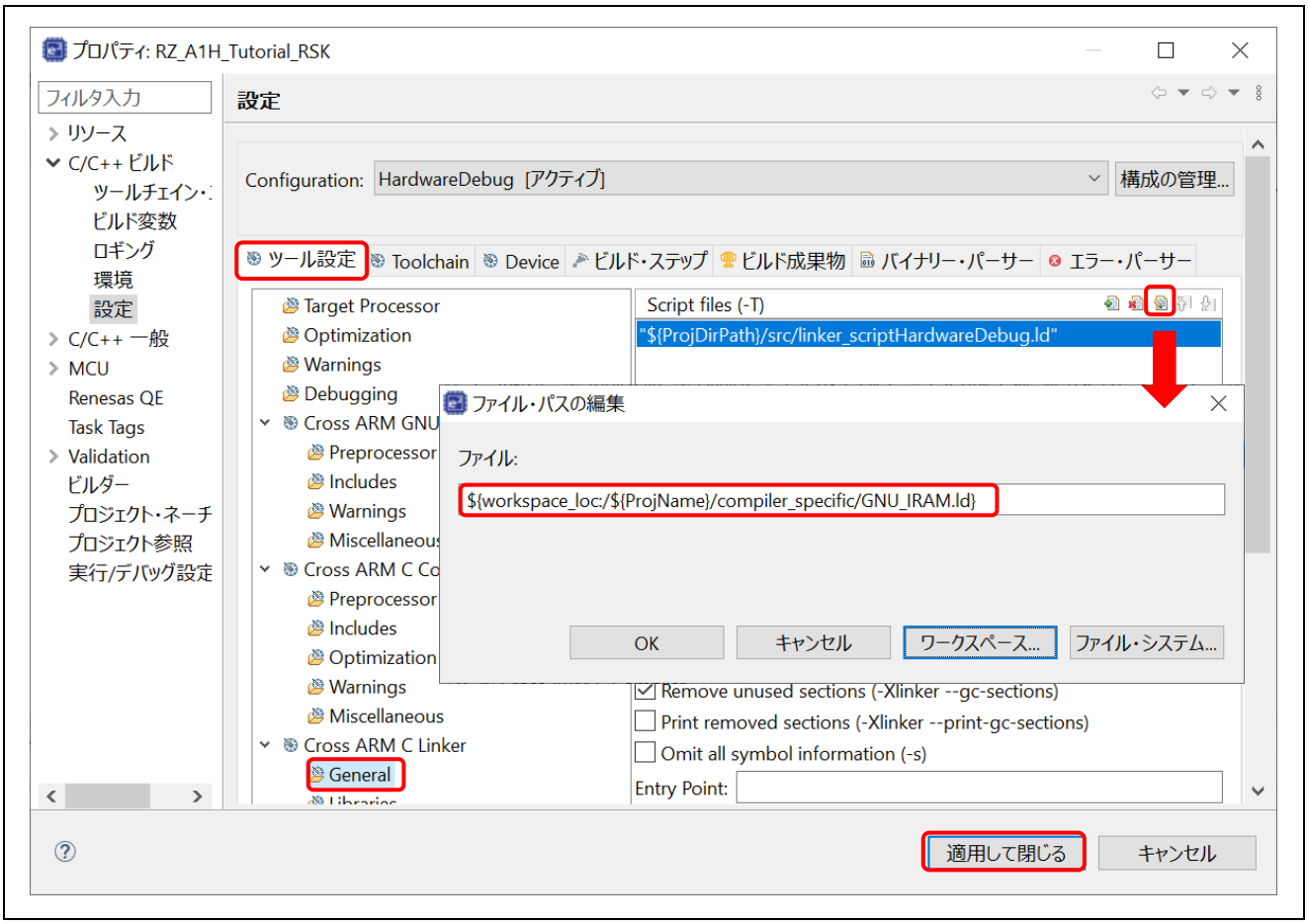

図 5-15 コンパイラが生成したスクリプトファイルの追加

7. プロジェクトをビルドし、成功したことを確認してください。

5.4.1 ブレークポイントビュー

ブレークポイントビュー(Breakpoints view)には、プログラムの実行可能な行に設定されたブレークポイントが表示されます。ブレークポイントがデバッグ中にイネーブルになった場合、当該行が実行される前に中断されます。e² studio ではソフトウェアブレークポイントとハードウェアブレークポイントを区別して設定できます。ダブルクリックによりブレークポイントを設定すると、デフォルトのブレークポイント型が適用されます。ハードウェアブレークポイントの場合、ブレークポイント用のハードウェアリソースが残っていないときは設定できません。その場合は、ソフトウェアブレークポイントに変更するようエラーメッセージが表示されます。

ブレークポイントの設定方法は 2 通りあります。

方法 1： エディタ画面上での設定

1. ソースコードの左余白を右クリックして [Toggle Software Breakpoint] または [Toggle Hardware Breakpoint] を選択してハードウェアブレークポイント またはソフトウェアブレークポイント を設定します。

方法 2： 「ブレークポイント」ビュー

ブレークポイント設定方法 2 では、

1. ソースコードの左余白を右クリックしてコンテクスト・メニューを開き、[ブレークポイント型] → [e2 studio Breakpoint]（デフォルトのハードウェアブレークポイント）または [ブレークポイント型] → [C/C++ ブレークポイント]（ソフトウェアブレークポイント）を選択します。
2. ソースコードの左余白をダブルクリックして、ハードウェアブレークポイントまたはソフトウェアブレークポイントを設定します。
3. [ビューの表示] → [ブレークポイント] またはアイコン をクリックし（あるいはショートカットキー [ALT]+[Shift]+[Q], [B] を使い）、[ブレークポイント] ビューを開いて、設定したハードウェアブレークポイントまたはソフトウェアブレークポイントを表示します。

ブレークポイントを無効にする際は、選択したブレークポイントのみを無効にするか、あるいはすべてのブレークポイントを無効にするか選択できます。

1. [ブレークポイント] ビューで、ブレークポイントを有効または無効に切り替えることができます。無効になったソフトウェアブレークポイントは白い丸が表示されます。無効になったハードウェアブレークポイントは白い丸が表示されます。
2. すべてのブレークポイントを無効にするには、[ブレークポイント] ビューにあるアイコン をクリックします。斜線を重ねた青い丸( )が [ブレークポイント] ビューとエディタ画面の両方に表示されます。

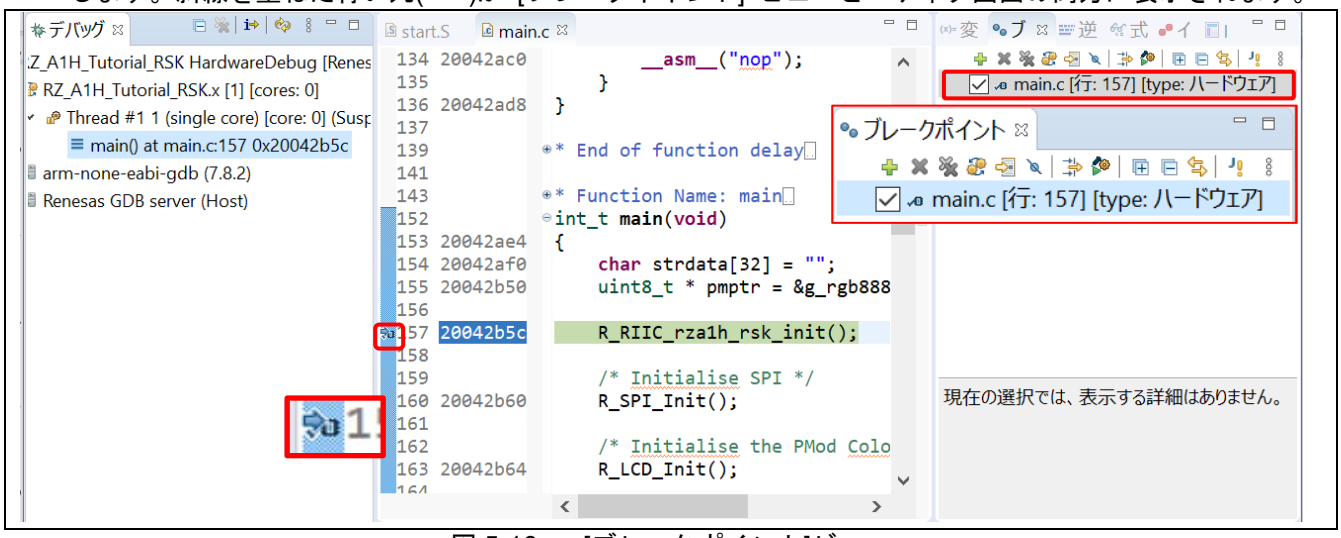

図 5-16 [ブレークポイント]ビュー

5.4.2 式ビュー

式ビュー(Expressions view)では、デバッグ中のグローバル変数、静的変数、ローカル変数の値をモニタできます。「リアルタイムリフレッシュ」を有効にすると、デバッガ実行中に設定された周期で変数の値が更新されます。

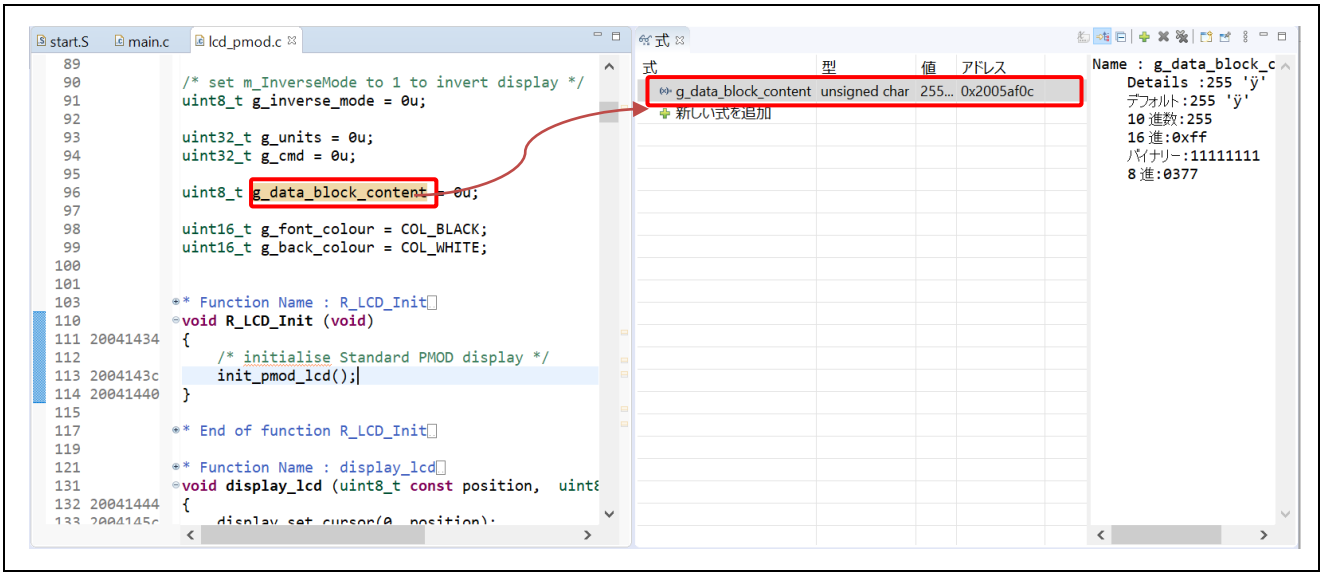

図 5-17 [式]ビュー

グローバル変数を見るには、

1. [ビューの表示] → [式] あるいはアイコン をクリックし、[式] ビューを開きます。
2. “main.c” の49行目で、グローバル変数 (例えば “gPeriodic_Delay”) を [式] ビューへドラッグ&ドロップします。（または、グローバル変数を右クリックして “監視式を追加(A)...” メニューアイテムを選択し、[式] ビューに追加します。）
3. [式] ビューで、“リアルタイム・リフレッシュ” メニューアイテムを右クリックして選択します。これで、プログラムの動作中、表示した値はリアルタイムに更新されます。“R” は、このグローバル変数がリアルタイムで更新されることを意味します。
4. “リアルタイム・リフレッシュ” を無効にするには、“リアルタイム・リフレッシュを無効にする” メニュー項目を右クリックして選択します。

5.4.3 レジスタービュー

レジスタービュー(Registers view)は汎用レジスタについての情報を表示します。プログラムを停止すると、変化のあった値を強調表示します。

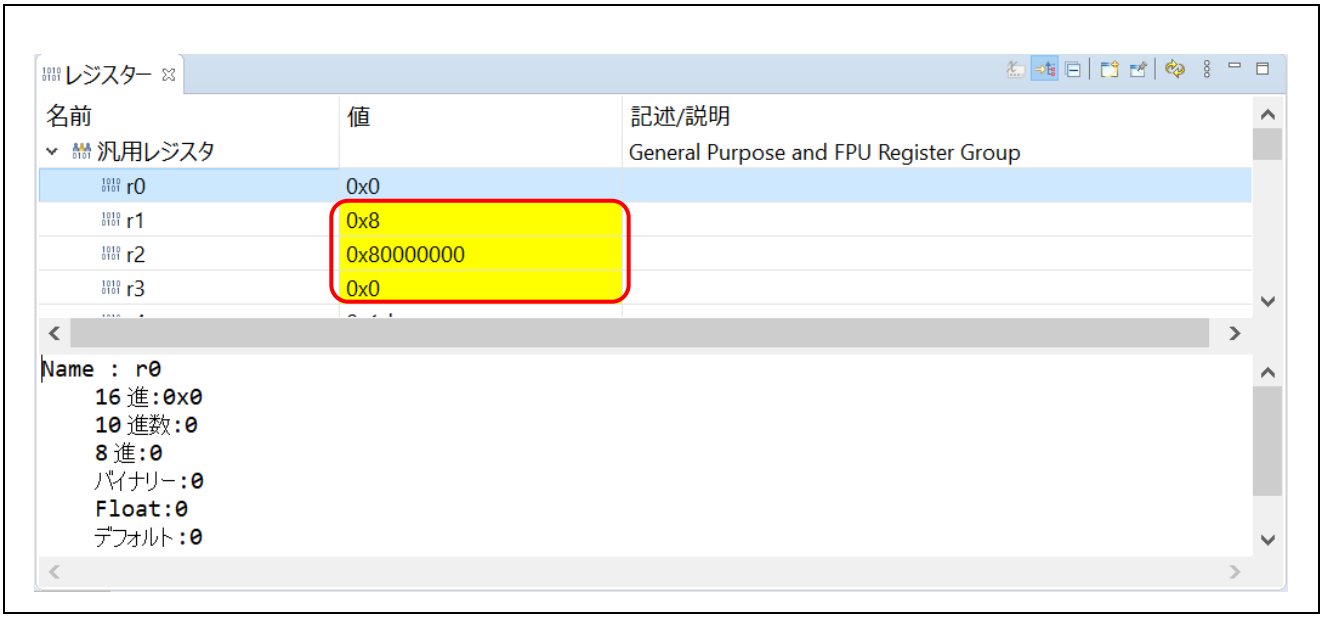

図 5-18 [レジスター]ビュー

汎用レジスタ “r0” を表示するには、

1. [ビューの表示] → [レジスター] あるいはアイコン をクリックし、[レジスター] ビューを開きます。
2. “r0” をクリックすると、他の基数フォーマットで値を表示します。

プログラムを停止すると、前回の表示以降で変化のあった値が [レジスター] ビューの中で強調表示（黄色など）されます。

5.4.4 メモリービュー

メモリービュー(Memory view)では、ユーザは“メモリーモニター”でメモリを表示し編集することができます。各モニターは“ベースアドレス”と呼ばれる格納位置によって特定される記憶場所を表します。各メモリーモニターの中のメモリデータは異なる“メモリレンダリング”で表示することができます。メモリレンダリングはあらかじめ設定したデータフォーマット（例えば、16 進数、符号付き整数、符号なし整数、ASCII、イメージなど）です。

変数（例えば“g_data_block_content”）のメモリを表示するには、

1. [ビューの表示] → [メモリー] あるいはアイコン をクリックし、[メモリー] ビューを開きます。
2. アイコン をクリックし、[モニター・メモリー] ダイアログボックスを開きます。変数“g_data_block_content”のアドレスを入力します。

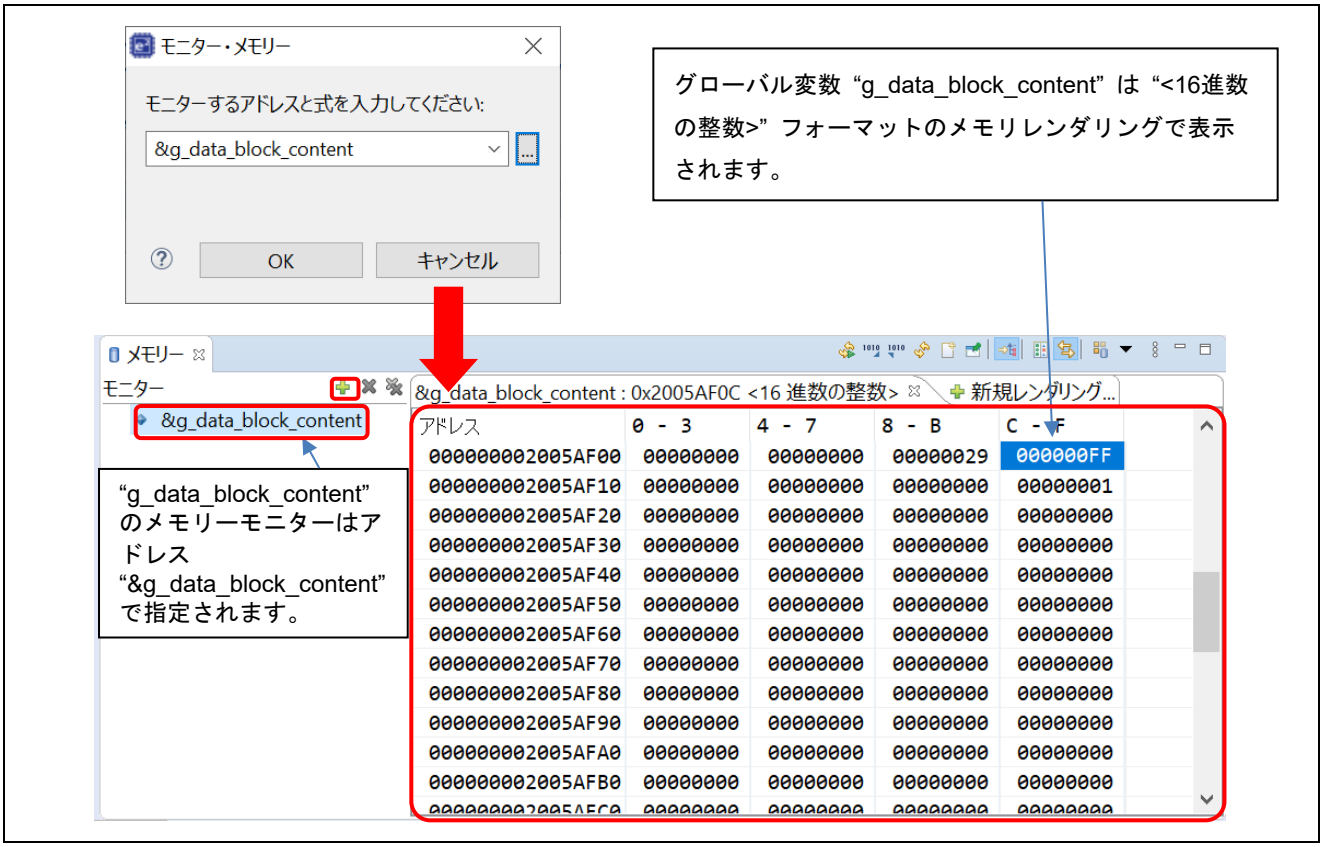

図 5-19 [メモリー] ビュー (1/2)

変数 “g_data_block_content” 用の新しいレンダリングフォーマット(例えば ASCII)を追加するには、

1. 新規レンダリング... タブをクリックし、“ASCII” を選択してレンダリングを追加します。
これで、“&g_data_block_content <16 進数の整数>” タブの横に新しい “&g_data_block_content < ASCII>” タブが作成されます。

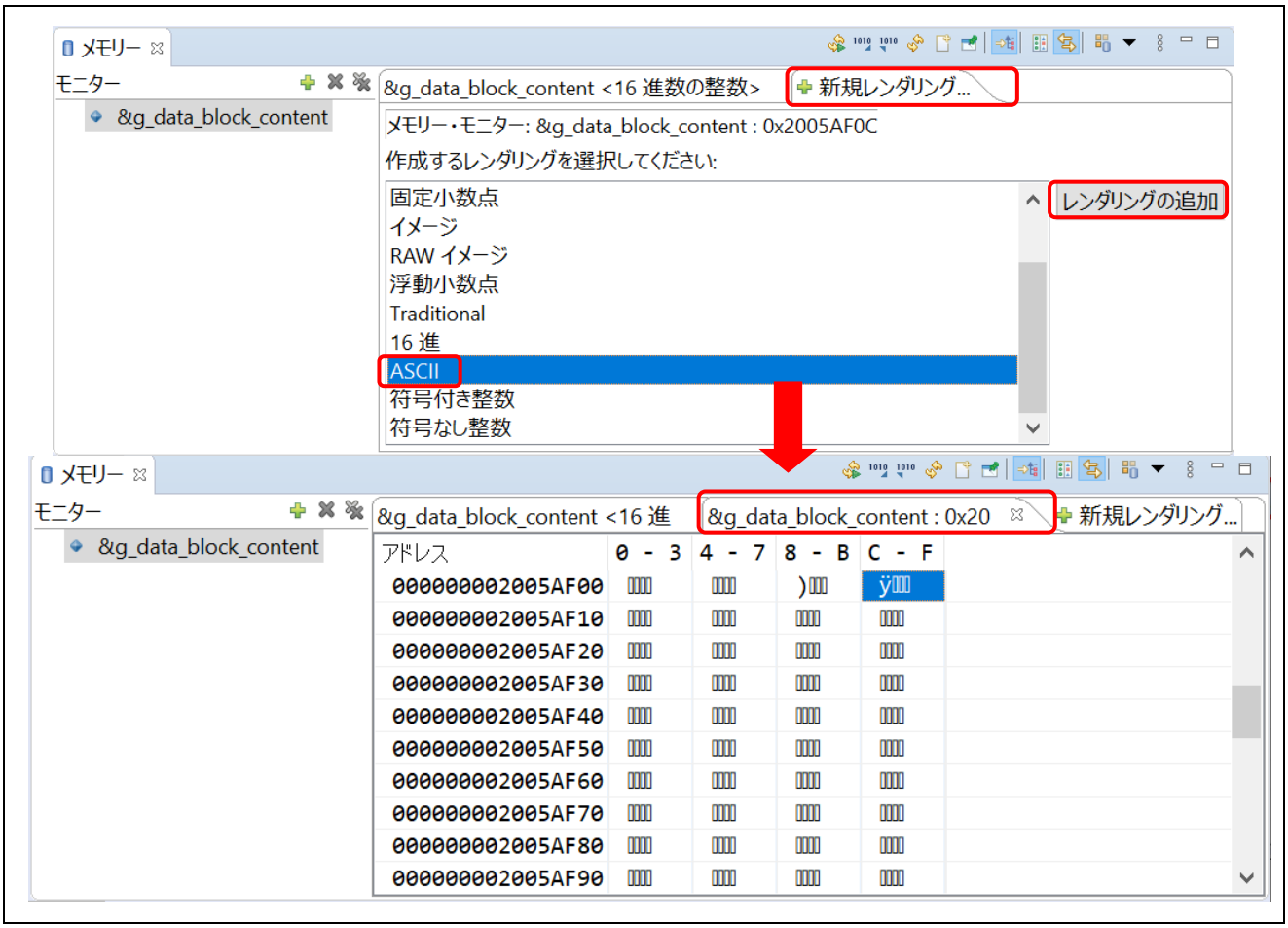

図 5-20 [メモリー] ビュー (2/2)

5.4.5 逆アセンブルビュー

逆アセンブルビュー(Disassembly view)は、ロードしたプログラムのソースコードとアセンブラ命令を混在して表示します。現在実行中の行は画面上で矢印のマーカで強調表示されます。逆アセンブルビューでは、アセンブラ命令へのブレークポイントの設定、ブレークポイントの有効化／無効化、逆アセンブル命令のステップ実行、プログラムの特定の命令へのジャンプが可能です。

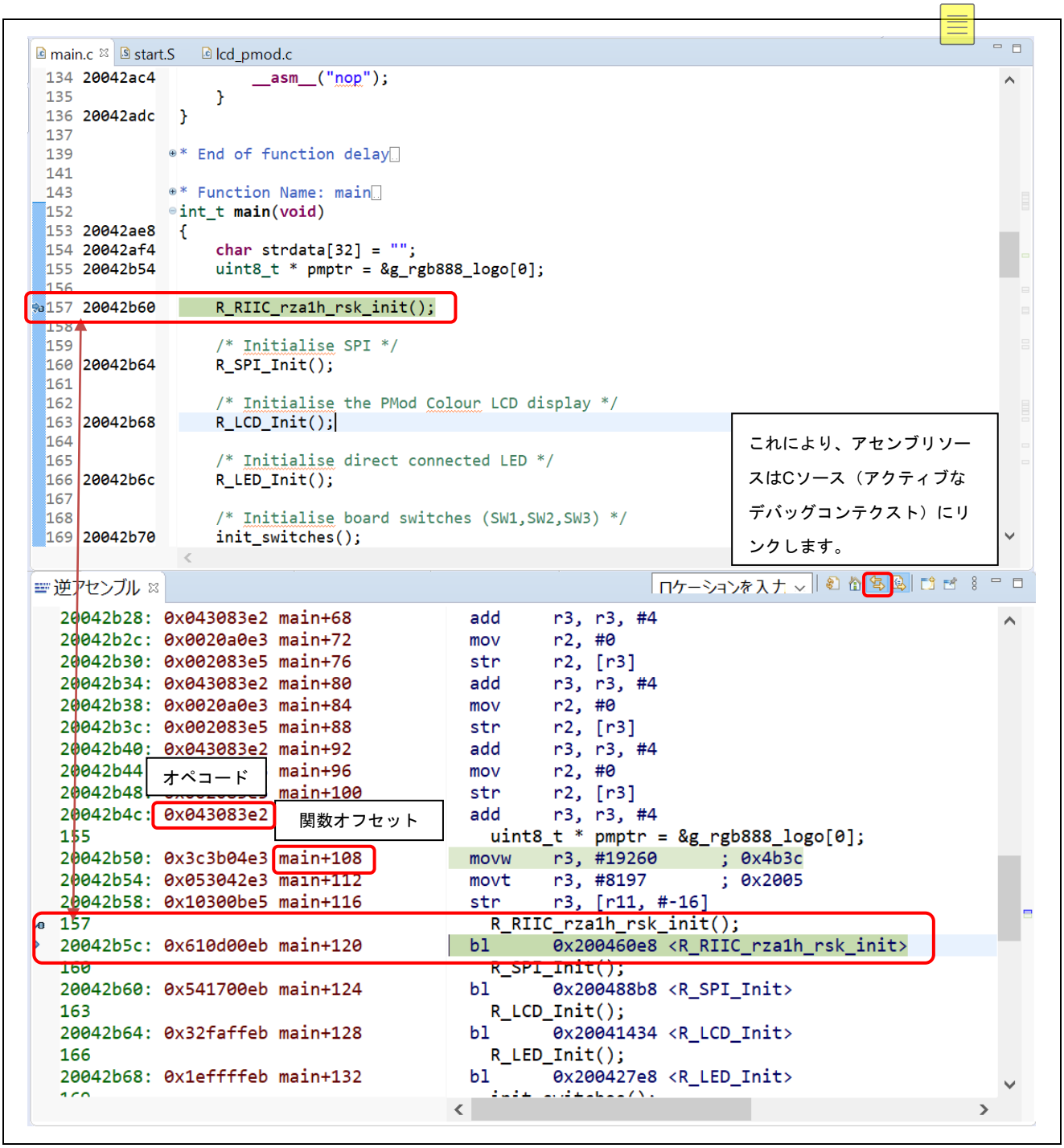

図 5-21 [逆アセンブル] ビュー

混合モードで C コードとアセンブリコードの両方を表示するには、

1. [ビューの表示] → [逆アセンブル] あるいはアイコン をクリックし、[逆アセンブル] ビューを開きます。
2. アイコン をクリックして、アセンブリソースとCソース（アクティブなデバッグコンテクスト）のリンクを有効にします。
3. [逆アセンブル] ビューのアドレス列で右クリックし、“オペコードを表示する” と “関数オフセットを表示する” を選択します。
4. コンテクスト・メニューを使用すればエディタ内でSource Addressesを有効にすることができます。

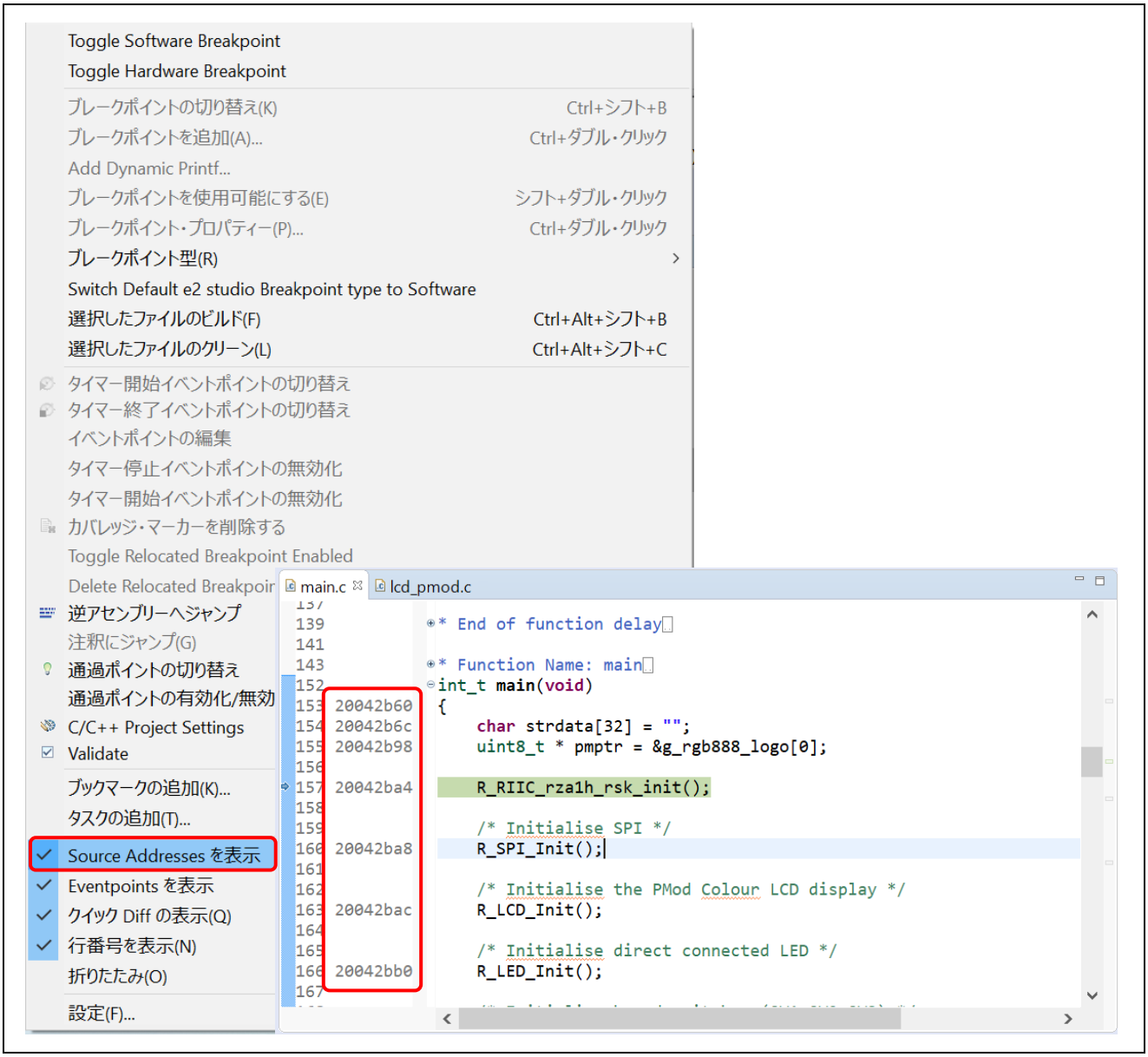

図 5-22 ソースアドレスメニュー

5.4.6 変数ビュー

変数ビュー(Variables view)は、現在実行中のスコープ内で表示可能な全てのローカル変数を表示します。

グローバル変数、あるいは、現在実行中のスコープ外の外部変数を見るには、‘式’ビューを参照してください。

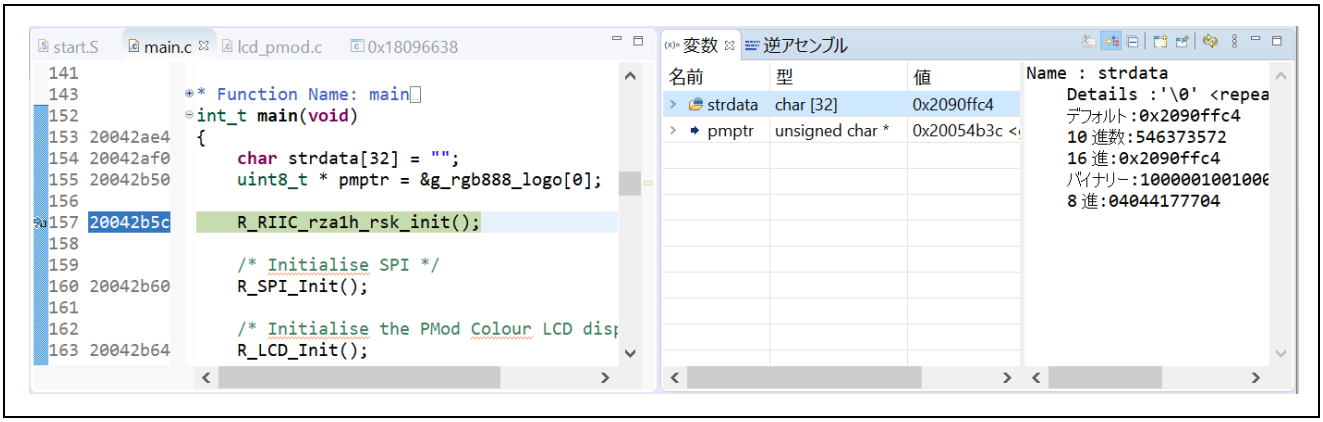

図 5-23 [変数]ビュー

ローカル変数（例えば、関数 “main()” の “strdata”）を見るには、

1. [ビューの表示] → [変数] またはアイコン (x)= をクリックし、[変数] ビューを開きます。
2. ステップ実行で関数 “main()” の中に入ると、[変数]ビュー内にローカル変数 “strdata” の値が表示されます。

注意: コンパイラやリンカの最適化処理により、変数の実体がなくなる(変数に対応するメモリを保持する必要がない場合や、変数や変数の演算に使用するレジスタが他の用途にも使われる場合など)と、変数ビューに値が表示されないことがあります。その際は逆アセンブリビューで変数に対応するレジスタやメモリがどれかを確認し、レジスタビュー等でその値を直接見てください。

5.4.7 イベントポイントビュー

イベントは、プログラム実行中にブレークあるいはトレース機能を実行するために設定された条件の組み合わせです。ユーザはイベントポイントビュー(Eventpoints view)で、異なる種類の定義されたイベント、たとえば、トレース開始、トレース終了、トレース・レコード、イベント・ブレーク、実行前 PC ブレーク、タイマー開始、およびタイマー終了、などを設定、表示することができます。

設定できるイベント数や設定条件は MCU によって異なります。以下に挙げる 2 種類のイベントがあります。

- 実行アドレス：エミュレータは CPU が特定のアドレスの命令を実行しようとしたことを検出します。これが “実行前 PC” ブレーク(例えば、イベントにより、条件は指定アドレスで命令の実行直前に成立する)、あるいは他のイベント(例えば、イベントにより、条件は指定アドレスで命令の実行直後に成立する)となります。
- データ・アクセス：エミュレータは指定された条件での指定アドレスあるいは指定アドレス範囲へのアクセスを検出します。**これにより、アドレスとデータを組み合わせた条件を設定することができます。**

イベントの組み合わせは（OR, AND（およびその組み合わせ），およびシーケンシャル）は 2 つ以上のイベントに使用できます。

変数をアクセスする条件（例えば、g_adc_result が新しい値として割り当てられたとき）で、グローバル変数にイベント・ブレークを設定するには、

1. [ビューの表示] → [イベントポイント] あるいはアイコン をクリックし、[イベントポイント] ビューを開きます。
2. “イベント・ブレーク” オプションをダブルクリックし、[編集イベント・ブレーク] ダイアログボックスを開きます。
3. [追加…] ボタンをクリックし、以下の操作を行います。

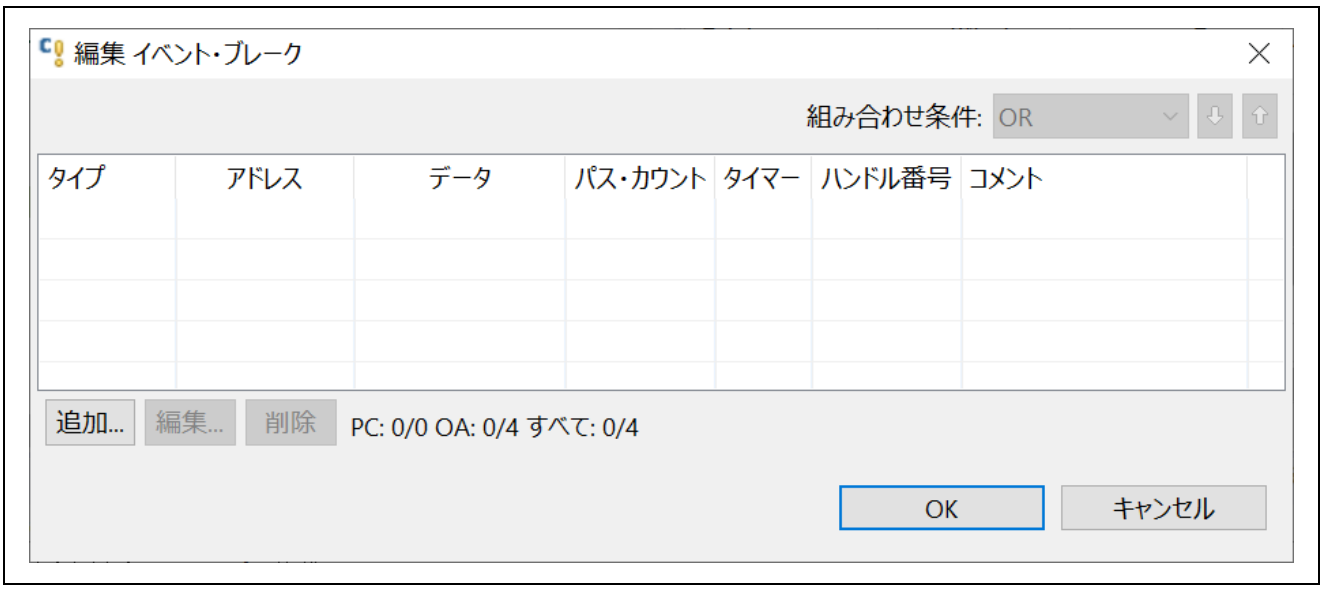

図 5-24 イベント・ブレークの編集

4. イベントポイントの種類に “Data Access” を選択します。
5. [アドレス設定] タブに進み、アイコン ... をクリックしてシンボル “g_adc_result” を検索します。（このグローバル変数のアドレスは “&g_adc_result” です。）

6. 次に、[データ・アクセス設定] タブに切り替え、[データ・アクセス設定] に “Write” を選択して、[OK] をクリックしてください。

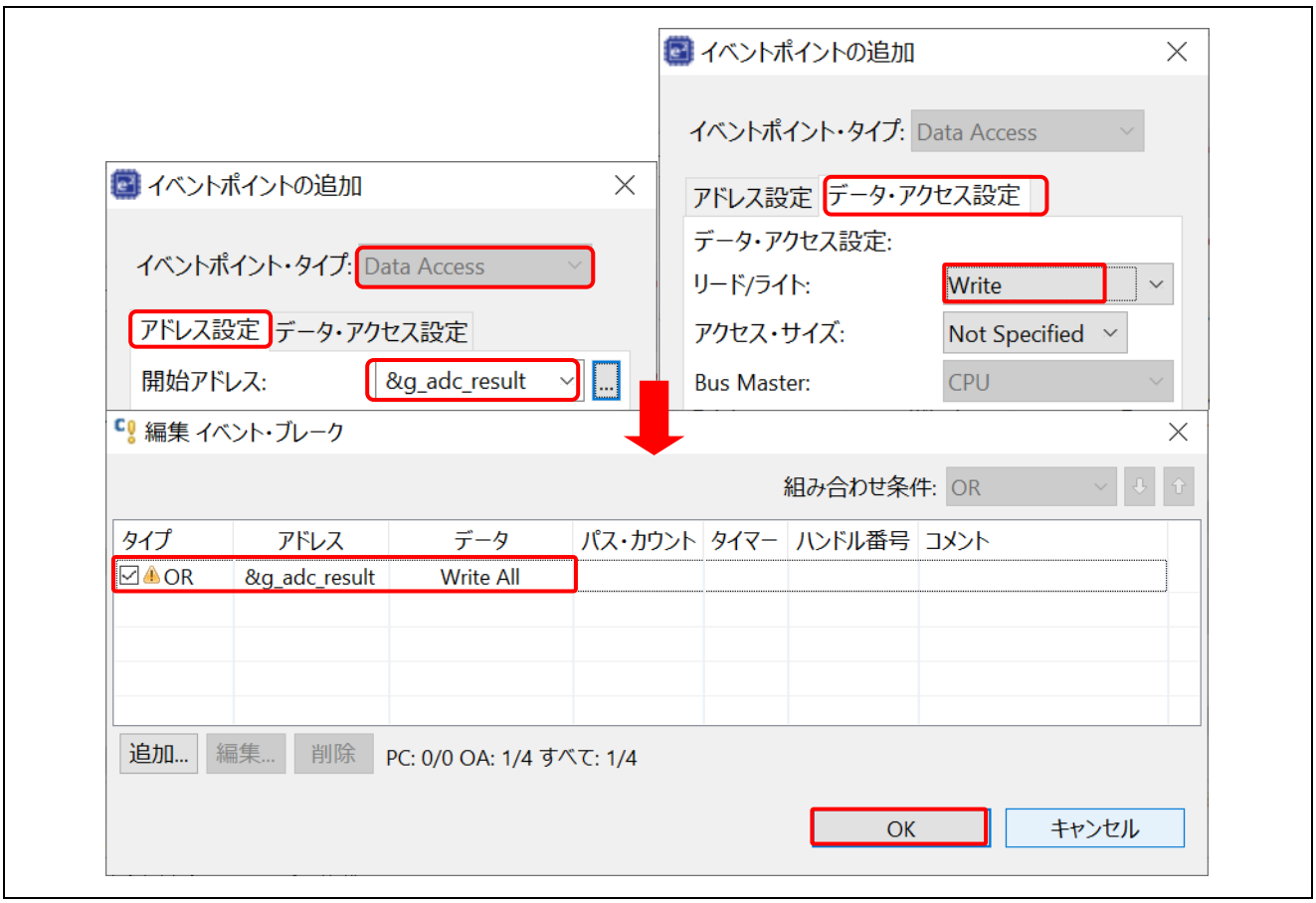

図 5-25 イベントポイントの追加

7. [イベントポイント] ビューで、イベント・ブレークが “g_adc_result” に設定されて有効になっていることを確認してください。プログラムを最初から実行するためにリセットします。

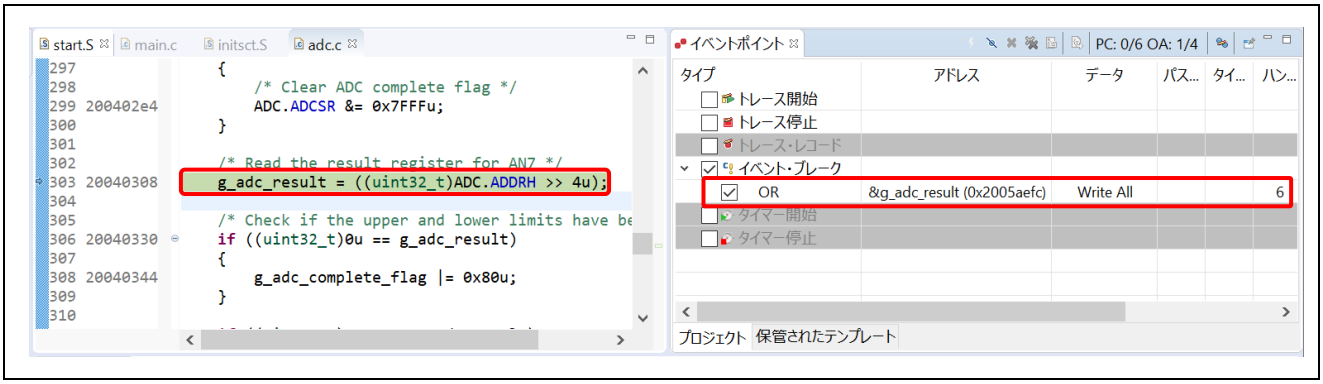

図 5-26 イベント・ブレークの実行

図 5-33 は、g_adc_result に新しい値が割り当てられたときにプログラムの 303 行目で実行が停止する様子を示します。

5.4.8 IO Registers ビュー

IO レジスタは Special Function Register（SFR）のことです。IO Registers ビュー(IO レジスタビュー)は、ターゲット専用の IO ファイルで定義された全レジスタセットの名前、アドレス、値（16 進数および 2 進数）を表示します。ユーザは、[選択されたレジスター] に必要な IO レジスタを選択して追加することによって、IO レジスタビューをカスタマイズすることができます。

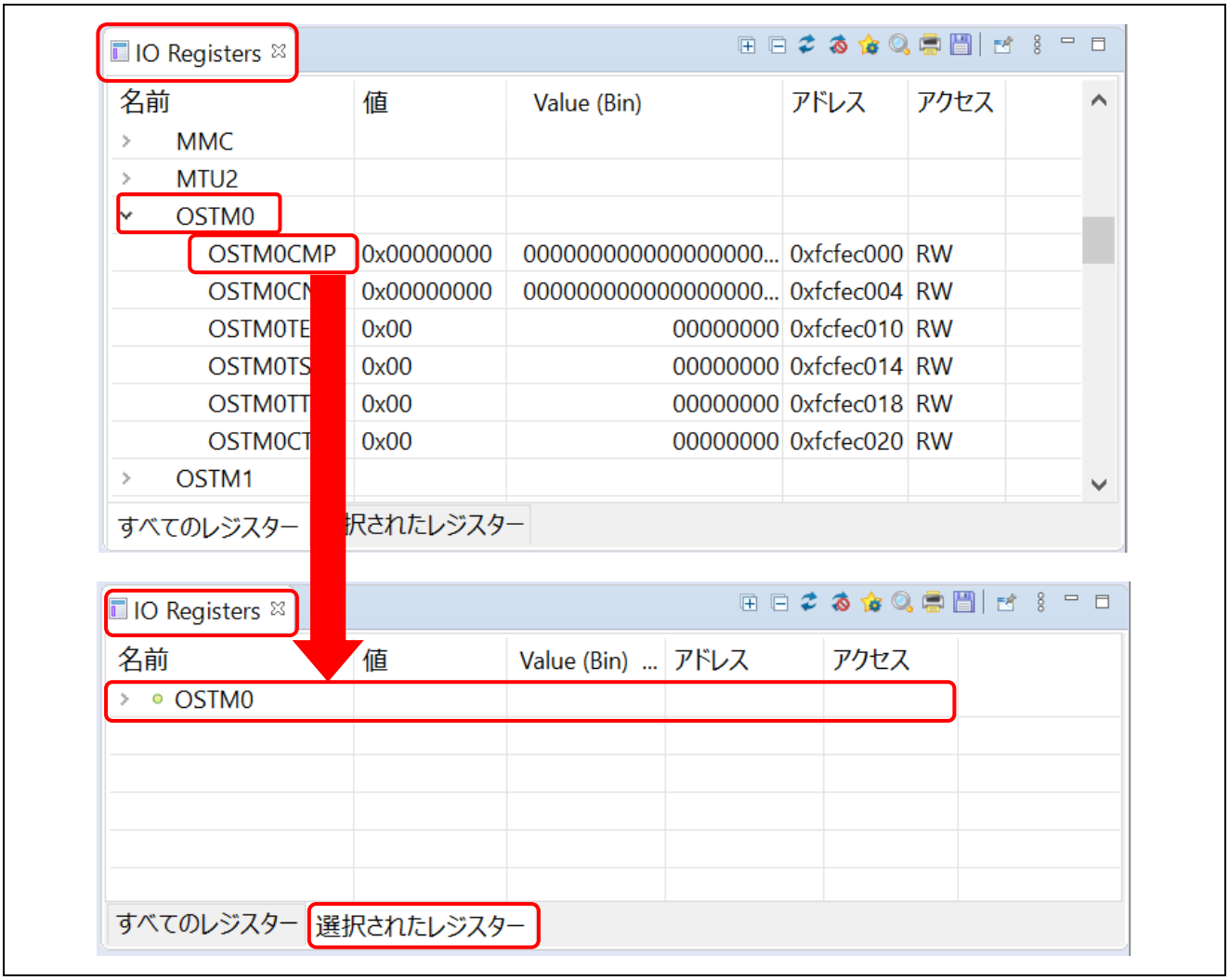

図 5-27 [IO Registers] ビュー

選択した IO レジスタ（例えば、OSTM0 の OSTM0CMP）を見るには、

1. IO Registersビューが表示されていない場合は、「ウィンドウ」メニューの「ビューの表示」から、「その他」のカテゴリ「デバッグ」の中から「IO Registers」を選択してください。
2. [すべてのレジスター] タブの [IO Registers] ビューに、[OSTM0] を表示します。OSTM0 IOレジスタの一覧を展開してください。
3. “OSTM0CMP” を [選択されたレジスター] へドラッグ&ドロップしてください。IOレジスタの横にある緑色のドット ● は、選択されているレジスタの状態を示します。
4. [選択されたレジスター] タブに切り替えて、“OSTM0” IOレジスタの “OSTM0CMP” を表示します。

「選択されたレジスター」タブには、指定したレジスタだけを表示させることができます。必要なレジスタを複数まとめて表示するのに便利です。「選択されたレジスター」を使用すると必要な IO レジスタの値だけをロードするのでデバッガ動作が重くなるのを防ぐこともできます。

5.4.9 トレースビュー

トレースとは、ユーザプログラム実行中、1 サイクルごとのバス情報をトレースメモリから取得することを意味します。取得されたトレース情報はトレースビューに表示されます。それによりユーザはプログラムの実行を追跡し、問題が発生した箇所を探すことができます。

トレースバッファは有限（1~32M バイトのサイズ）なため、バッファが一杯になると、最も古いトレースデータを新しいデータで上書きします。

RZ/A1H では、トレースを完全にサポートしていません。

5.5 デバッグ手順（RZ/A2M の場合）

本節では、RZ/A2M のデバッグ手順について、Segger J-Link エミュレータと RZ/A2M Evaluation Board Kit をターゲットボードとする”Loader”プロジェクト、”Application”プロジェクト(3.2.1 節)を例として説明します。

1. [プロジェクト・エクスプローラー] の “Loader” プロジェクトをクリックします。
2. (4.2節)を参考にビルドし、”Loader.elf”ファイルを作成しておきます。
3. Launch Barの[Launch Configuration]にて “Loader HardwareDebug”を選択します。

図 5-28 [Launch Configuration]の選択

4. Launch Barの ボタンをクリックし、ターゲットボートとの接続、およびターゲットボードへの”Loader.elf”ダウンロードを実行します。

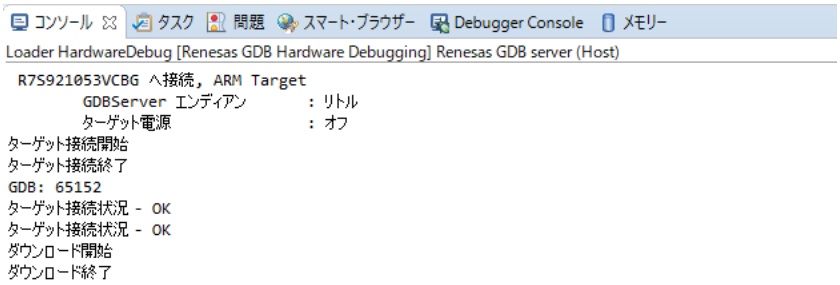

図 5-29 ダウンロード実行後の[コンソール]ビュー

5. Launch Barの ボタンをクリックして終了します。
6. 次に[プロジェクト・エクスプローラー] の “Application” プロジェクトをクリックします。
7. (4.2節)を参考にビルドし、”Application.elf”ファイルを作成しておきます。
8. Launch Barの[Launch Configuration]にて “Application HardwareDebug”を選択します。

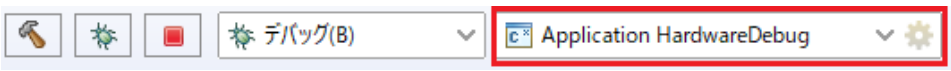

図 5-30 [Launch Configuration]の選択

9. Launch Barの ボタンをクリックし、ターゲットボートとの接続、およびターゲットボードへの”Application.elf”ダウンロードを実行します。
10. ダウンロード完了後、 ボタンをクリックし、”Loader.elf”のエントリポイントからプログラムを再実行させます。

6. ヘルプ

ヘルプシステムによって、ユーザはワークベンチ内の各ヘルプウィンドウやヘルプ画面から、ヘルプドキュメントのブラウズ、検索、ブックマーク、印刷が可能です。また、ヘルプメニューから e² studio 専用のオンラインフォーラムにアクセスできます。

[ヘルプ] をクリックしてヘルプメニューをプルダウンしてください。

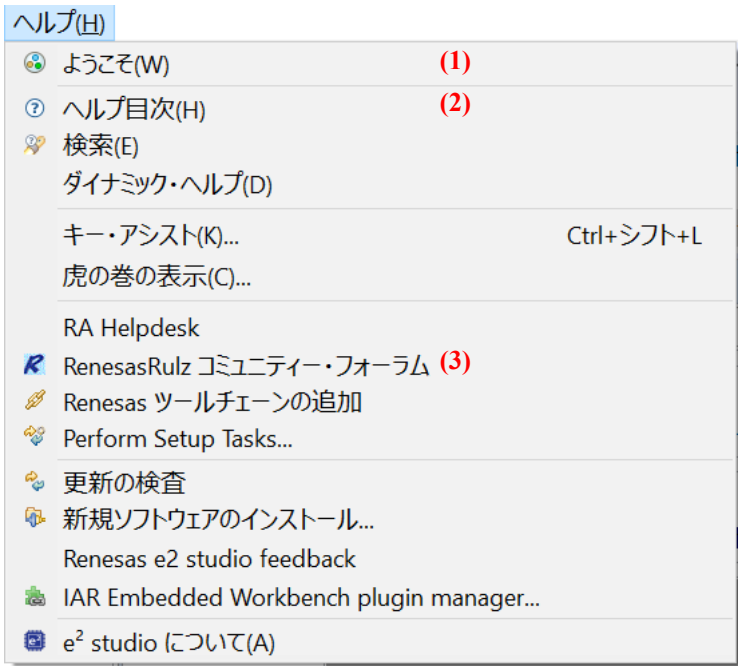

図 6-1 ヘルプメニュー

ヘルプメニューの使い方

(1) [ようこそ] をクリックすると、e² studioの概要、e² studioチュートリアルとサンプルプログラムへのリンク、リリースノートを表示します。

(2) [ヘルプ目次] をクリックすると、新たにヘルプウィンドウが開きヘルプを検索できます。

(3) [RenesasRulzコミュニティー・フォーラム] をクリックすると、e² studio関連のディスカッション参加型オンラインフォーラムにアクセスします。インターネット接続が必要です。

改訂記録	e^2 studio 統合開発環境ユーザーズマニュアル　入門ガイド

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Sep.13.19	ー	初版発行
1.01	Mar.3.20	22-33	「RZ/A2M のプロジェクトの作成」を追加
		67	「デバッグ手順（RZ/A2M の場合）」を追加
1.02	Sep.24.20	ー	e^2 studio 7.8.0, 2020-04 に対応

e^2 studio 統合開発環境ユーザーズマニュアル
入門ガイド

発行年月日　2020年9月24日　Rev.1.02

発行　ルネサス エレクトロニクス株式会社
〒135-0061　東京都江東区豊洲 3-2-24（豊洲フォレシア）

e2 studio

RENESAS

ルネサス エレクトロニクス株式会社

R20UT4535JJ0102