User's Manual

CubeSuite+ V1.03.00

統合開発環境

ユーザーズマニュアル　ビルド編（CXコンパイラ）

対象デバイス

V850 ファミリ

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　2012.09

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、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 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に関し、当社は、何らの責任を負うものではありません。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. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製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）、交通用信号機器、
   防災・防犯装置、各種安全装置等

   当社製品は、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（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障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、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、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。お客様が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. 本資料に記載されている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。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
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。

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その総株主の議決権の過半数を直接または間接に保有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。

(2012.4)

このマニュアルの使い方

このマニュアルは，V850 ファミリ用アプリケーション・システムを開発する際の統合開発環境である CubeSuite+ について説明します。

CubeSuite+ は，V850 ファミリの統合開発環境（ソフトウエア開発における，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プラットフォームである IDE に統合）です。統合することで，さまざまなツールを使い分ける必要がなく，本製品のみを使用して開発のすべてを行うことができます。

対 象 者　このマニュアルは，CubeSuite+ を使用してアプリケーション・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，CubeSuite+ の持つソフトウエア機能をユーザに理解していただき，これらのデバイスを使用するシステムのハードウエア，ソフトウエア開発の参照用資料として役立つことを目的としています。

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。

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，マイクロコンピュータ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	：	$\overline{\text{XXX}}$（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
		16 進数　...　0xXXXX

関連資料　関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。あらかじめご了承ください。

資料名		資料番号	
		和文	英文
CubeSuite+ 統合開発環境 ユーザーズ・マニュアル	起動編	R20UT2133J	R20UT2133E
	V850 設計編	R20UT2134J	R20UT2134E
	RL78 設計編	R20UT2136J	R20UT2136E
	78K0R 設計編	R20UT2137J	R20UT2137E
	78K0 設計編	R20UT2138J	R20UT2138E
	RX コーディング編	R20UT0767J	R20UT0767E
	V850 コーディング編	R20UT0553J	R20UT0553E
	コーディング編（CX コンパイラ）	R20UT2139J	R20UT2139E
	RL78, 78K0R コーディング編	R20UT2140J	R20UT2140E
	78K0 コーディング編	R20UT2141J	R20UT2141E
	RX ビルド編	R20UT0768J	R20UT0768E
	V850 ビルド編	R20UT0557J	R20UT0557E
	ビルド編（CX コンパイラ）	このマニュアル	R20UT2142E
	RL78, 78K0R ビルド編	R20UT2143J	R20UT2143E
	78K0 ビルド編	R20UT0783J	R20UT0783E
	RX デバッグ編	R20UT2175J	R20UT2175E
	V850 デバッグ編	R20UT2144J	R20UT2144E
	RL78 デバッグ編	R20UT2145J	R20UT2145E
	78K0R デバッグ編	R20UT0732J	R20UT0732E
	78K0 デバッグ編	R20UT0731J	R20UT0731E
	解析編	R20UT2146J	R20UT2146E
	メッセージ編	R20UT2147J	R20UT2147E

注意　上記関連資料は，予告なしに内容を変更することがあります。設計などには，必ず最新の資料を使用してください。

目　　次

第1章　概　　説

この章では，ビルド・ツール（CX）の概要について説明します。

1.1　概　　要

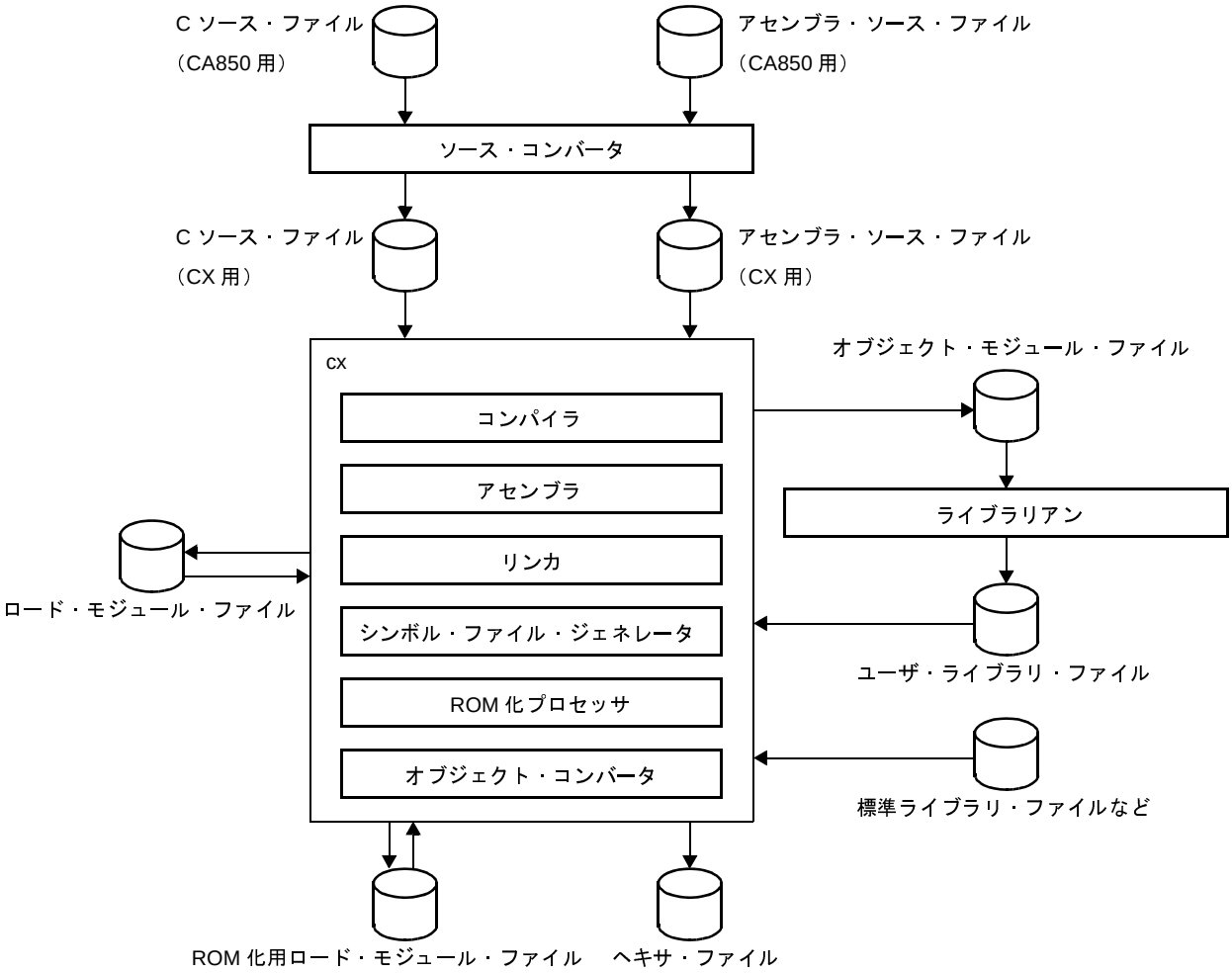
ビルド・ツール（CX）は，本製品が提供しているコンポーネントの一種であり，GUI ベースで各種情報を設定することにより，ソース・ファイルからロード・モジュール・ファイル，ROM 化用ロード・モジュール・ファイル，ヘキサ・ファイル，ユーザ・ライブラリ・ファイルを，目的に応じて生成することができます。

本マニュアルで説明する CX は，以下のコマンドで構成されています。

- cx
- ライブラリアン
- ソース・コンバータ

以下に，CX の処理の流れを示します。

図 1—1 CX の処理の流れ



1.2 特 長

以下に，ビルド・ツール（CX）の特長を示します。

- 最適化機能

  コンパイル時に，コード・サイズ優先，実行速度優先などの最適化を行うことにより，効率の良いロード・モジュール・ファイルを生成することができます。

- 8 バイト型（double 型，long long 型）のサポート

  CX では，double 型を 8 バイトとして扱います。

  また，C99 で規定されている long long 型（signed ／ unsigned）をサポートします。

- スマート・コレクション機能

  特定の関数を修正したい場合に，ほかの関数の内容（コード，およびアドレス）を変更することなく，その関数の処理内容のみを置き換えることができます。

  備考 スマート・コレクション機能についての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

- エラー後のリンク処理の継続

  メモリ・オーバフローのエラーが発生した場合，CX は，リカバリ可能であれば，リンク処理を継続し，メモリの不足情報を表示します。

第2章 機 能

この章では，CubeSuite+ を使用したビルドの手順，およびビルドに関する主な機能について説明します。

2.1 概 要

ここでは，ロード・モジュール，およびユーザ・ライブラリの作成手順について説明します。

備考 CA850 用ファイルから CX 用ファイルへの変換についての詳細は，「B. 3 ソース・コンバータ」を参照してください。

2.1.1 ロード・モジュールを作成する

ロード・モジュールの作成手順を以下に示します。

(1) プロジェクトの作成／読み込み

プロジェクトの新規作成，または既存のプロジェクトの読み込みを行います。

備考 プロジェクトの新規作成，および既存のプロジェクトの読み込みについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル 起動編」を参照してください。

(2) ビルド対象プロジェクトの設定

ビルド対象とするプロジェクトを設定します（「2. 15 ビルドの設定をする」参照）。

備考 1. プロジェクトにサブプロジェクトがない場合，そのプロジェクトは常にアクティブになります。

2. ビルド・モードを設定する場合は，ビルド・モードを追加してください（「2. 15. 6 ビルド・モードを追加する」参照）。

(3) ビルド対象ファイルの設定

ビルド対象ファイルの追加／削除，依存関係の更新などを行います（「2. 3 ビルド対象ファイルを設定する」参照）。

備考 1. ユーザ・ライブラリのプロジェクトへの追加方法については，「2. 7. 1 ユーザ・ライブラリを追加する」を参照してください。

2. オブジェクト・モジュール・ファイル，およびライブラリ・ファイルのリンク順は，ユーザが設定することもできます（「2. 15. 2 ファイルのリンク順を設定する」参照）。

(4) ロード・モジュールの出力指定

生成するロード・モジュールの種類を設定します（「2. 4 出力ファイルの種類を設定する」参照）。

(5) ビルド・オプションの設定

コンパイラ，アセンブラ，リンカなどに対するオプションを設定します（「2. 5 コンパイル・オプションを設定する」，「2. 6 アセンブル・オプションを設定する」，「2. 7 リンク・オプションを設定する」参照）。

(6) ビルドの実行

ビルドを実行します（「2. 16 ビルドを実行する」参照）。

ビルドには，次の種類があります。

- ビルド（「2. 16. 1 更新ファイルのビルドを実行する」参照）
- リビルド（「2. 16. 2 すべてのファイルのビルドを実行する」参照）
- ラピッド・ビルド（「2. 16. 3 他の処理と平行してビルドを実行する」参照）
- バッチ・ビルド（「2. 16. 4 ビルド・モードを一括してビルドを実行する」参照）

備考 ビルド処理前，およびビルド処理後に実行したいコマンドがある場合は，プロパティ パネルの［共通オプション］タブの［その他］カテゴリにおいて，［ビルド前に実行するコマンド］プロパティ，および［ビルド後に実行するコマンド］プロパティを設定してください。
ファイル単位でビルド処理前，およびビルド処理後に実行したいコマンドがある場合は，［個別コンパイル・オプション］タブ（C ソース・ファイルの場合），および［個別アセンブル・オプション］タブ（アセンブラ・ソース・ファイルの場合）において設定することができます。

(7) プロジェクトの保存

プロジェクトの設定内容をプロジェクト・ファイルに保存します。

備考 プロジェクトの保存についての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル 起動編」を参照してください。

2. 1. 2 ユーザ・ライブラリを作成する

ユーザ・ライブラリの作成手順を以下に示します。

(1) プロジェクトの作成／読み込み

プロジェクトの新規作成，または既存のプロジェクトの読み込みを行います。

プロジェクトを新規作成する際は，ライブラリ用のプロジェクトを設定します。

備考 プロジェクトの新規作成，および既存のプロジェクトの読み込みについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル 起動編」を参照してください。

(2) ビルド対象プロジェクトの設定

ビルド対象とするプロジェクトを設定します（「2. 15 ビルドの設定をする」参照）。

備考 1. プロジェクトにサブプロジェクトがない場合，そのプロジェクトは常にアクティブになります。

2. ビルド・モードを設定する場合は，ビルド・モードを追加してください（「2. 15. 6 ビルド・モードを追加する」参照）。

(3) ビルド対象ファイルの設定

ビルド対象ファイルの追加／削除，依存関係の更新などを行います（「2.3 ビルド対象ファイルを設定する」参照）。

(4) ビルド・オプションの設定

コンパイラ，アセンブラ，ライブラリアンに対するオプションを設定します（「2.5 コンパイル・オプションを設定する」，「2.6 アセンブル・オプションを設定する」，「2.10 ライブラリ生成オプションを設定する」参照）。

備考 デバイス共通のライブラリを作成する場合は，プロパティ パネルの［共通オプション］タブの［出力ファイルの種類と場所］カテゴリにおいて，［デバイス共通オブジェクト・モジュールを出力する］プロパティを設定してください。

(5) ビルドの実行

ビルドを実行します（「2.16 ビルドを実行する」参照）。

ビルドには，次の種類があります。

- ビルド（「2.16.1 更新ファイルのビルドを実行する」参照）
- リビルド（「2.16.2 すべてのファイルのビルドを実行する」参照）
- ラピッド・ビルド（「2.16.3 他の処理と平行してビルドを実行する」参照）
- バッチ・ビルド（「2.16.4 ビルド・モードを一括してビルドを実行する」参照）

備考 ビルド処理前，およびビルド処理後に実行したいコマンドがある場合は，プロパティ パネルの［共通オプション］タブの［その他］カテゴリにおいて，［ビルド前に実行するコマンド］プロパティ，および［ビルド後に実行するコマンド］プロパティを設定してください。

ファイル単位でビルド処理前，およびビルド処理後に実行したいコマンドがある場合は，［個別コンパイル・オプション］タブ（C ソース・ファイルの場合），および［個別アセンブル・オプション］タブ（アセンブラ・ソース・ファイルの場合）において設定することができます。

(6) プロジェクトの保存

プロジェクトの設定内容をプロジェクト・ファイルに保存します。

備考 プロジェクトの保存についての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル 起動編」を参照してください。

2.2 ビルド・ツールのバージョンを変更する

プロジェクト（メイン・プロジェクト，またはサブプロジェクト）で使用するビルド・ツール（コンパイラ・パッケージ）のバージョンを変更することができます。

プロジェクト・ツリー でビルド・ツール・ノードを選択したのち，プロパティ パネルの［共通オプション］タブを選択します。［バージョン選択］カテゴリの［使用するコンパイラ・パッケージのバージョン］プロパティで［常にインストール済みの最新版］，または該当バージョンを選択してください。

図 2—1 ［使用するコンパイラ・パッケージのバージョン］プロパティ

備考 1. メイン・プロジェクト，およびサブプロジェクトで使用するビルド・ツールが同じ場合，それらのビルド・ツール・ノードをすべて選択し，プロパティを設定することで，ビルド・ツールのバージョンをまとめて変更することができます。

2. ほかの実行環境で作成したプロジェクトを開いた場合など，インストールしていないコンパイラ・パッケージを選択している場合は，そのバージョンも表示します。

3. コンパイラ・パッケージによってオプションに変更がある場合は，選択したバージョンにあわせて，ビルド・ツールの各プロパティの表示を切り替えます。

バージョンの変更により非表示になるプロパティについては，プロジェクト・ファイル中に設定値を残しておき，再表示の際に値を再現します。

なお，オプションの変更は，以下の規則に基づいて行い，変更情報は出力 パネルに表示します。

- 旧バージョンから新バージョンへ変更した場合は，オプションの設定の引き継ぎ，および変換（必要な場合のみ）を行います。
- 新バージョンから旧バージョンへ変更した場合は，同一オプションの設定の引き継ぎのみを行います。旧バージョンのみに存在するオプションについては，デフォルト値を設定します。

2. 3 ビルド対象ファイルを設定する

ビルドを実行する前に，ビルド対象となるファイル（C ソース・ファイル，アセンブラ・ソース・ファイルなど）をプロジェクトに追加しておく必要があります。

ここでは，プロジェクトにおけるファイルの設定に関する操作を説明します。

2. 3. 1 スタートアップ・ルーチンを設定する

（1）標準のスタートアップ・ルーチンを使用する場合

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

標準のスタートアップ・ルーチン（CX 付属のオブジェクト・モジュール・ファイル）を使用するには，［入力ファイル］カテゴリの［標準のスタートアップを使用する］プロパティで［はい］を選択してください。

図 2—2 ［標準のスタートアップを使用する］プロパティ

備考 ［ROM 化オプション］タブの［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合は cstart.obj，［いいえ (-Xno_romize)］を選択した場合は cstartN.obj をリンクします。

（2）標準以外のスタートアップ・ルーチンを使用する場合

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

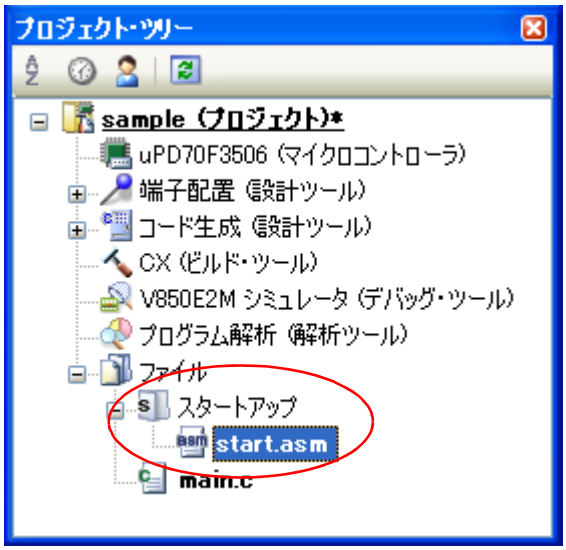
標準以外のスタートアップ・ルーチンを使用するには，［入力ファイル］カテゴリの［標準のスタートアップを使用する］プロパティで［いいえ (-Xno_startup)］を選択してください（デフォルト：［はい］）。

図 2—3 ［標準のスタートアップを使用する］プロパティ

次に，スタートアップ・ルーチンを記述したファイル（スタートアップ・ファイル）をプロジェクト・ツリーのスタートアップ・ノードに追加してください。

備考 プロジェクト・ツリーへのファイルの追加方法については，「2. 3. 3 プロジェクトにファイルを追加する」を参照してください。

図 2—4 プロジェクト・ツリー パネル（スタートアップ・ファイル追加後）



注意 プロジェクト・ツリーのスタートアップ・ノード直下に追加しているファイルのうち，ビルド対象ファイルをスタートアップ・ファイルとみなします。
スタートアップ・ノード以下のカテゴリに追加した場合は，スタートアップ・ファイルとはみなしません。
スタートアップ・ファイルをスタートアップ・ノードに追加する際，すでにスタートアップ・ファイルを追加している場合は，追加する最新のスタートアップ・ファイルのみがビルド対象となり，追加済みのスタートアップ・ファイルはビルド対象外となります。
ビルド対象外となっているスタートアップ・ファイルをビルド対象に設定する際，ほかにもスタートアップ・ファイルを追加している場合は，ビルド対象に設定したスタートアップ・ファイルのみがビルド対象となり，それ以外のスタートアップ・ファイルはビルド対象外となります。

備考 スタートアップ・ルーチンの作成方法については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

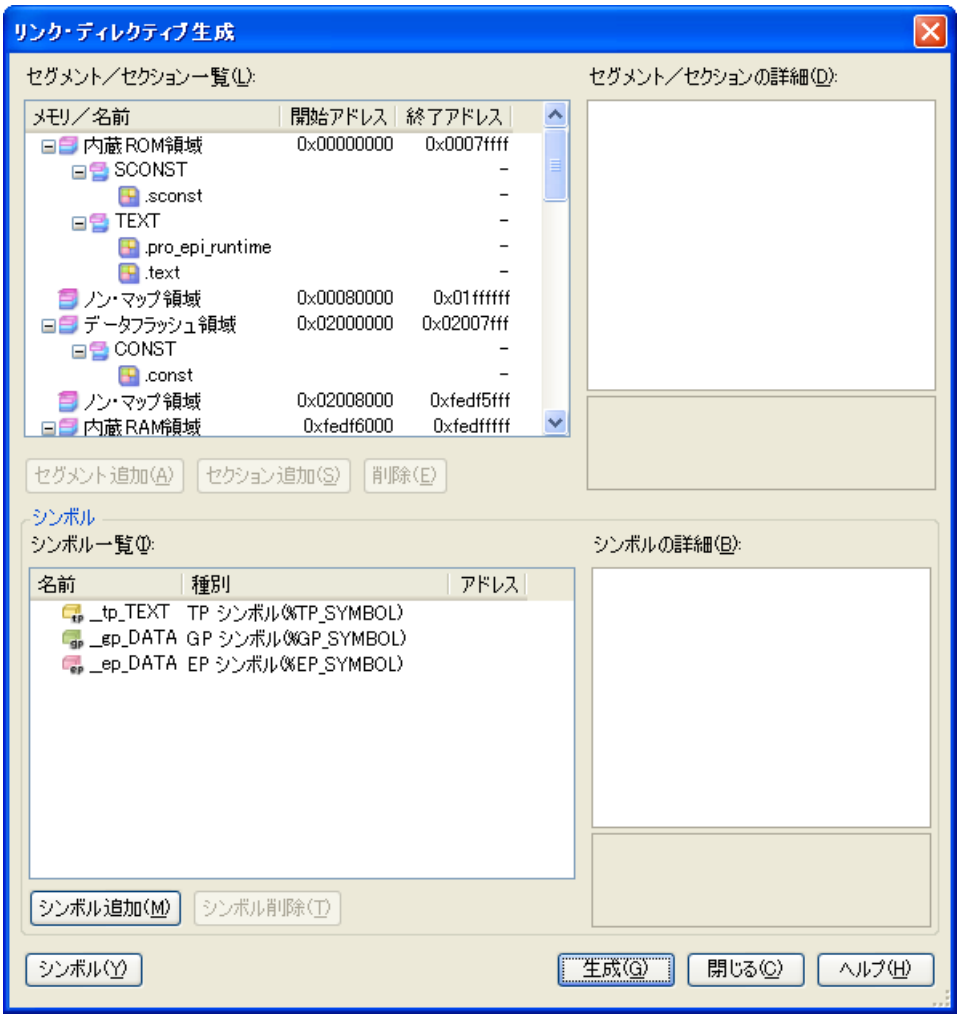
2. 3. 2 リンク・ディレクティブを自動生成する

リンク・ディレクティブ・ファイルはユーザが作成してプロジェクトに追加しますが，CubeSuite+ 上で自動生成することもできます。

備考 リンク・ディレクティブの詳細，およびリンク・ディレクティブ・ファイルの作成方法については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，コンテキスト・メニューの［リンク・ディレクティブ・ファイルを生成する ...］を選択すると，リンク・ディレクティブ生成 ダイアログがオープンします。

図 2—5 リンク・ディレクティブ生成 ダイアログ



ダイアログ上で，セグメント／セクション，シンボルの編集を行います。

(1) セグメント／セクションの編集

［セグメント／セクション一覧］エリアには，デバイスのメモリ配置情報と，現在設定しているセグメントとセクションの一覧を表示しています。

一覧において，セグメント／セクションを選択すると，［セグメント／セクションの詳細］エリアに該当セグメント／セクションの詳細情報を表示します。

［セグメント／セクションの詳細］エリアにおいて，各項目を編集してください。

備考 予約セクションについては，編集不可の項目（自動で値を設定する項目）があります。
各項目の詳細，および予約セクションの扱いについては，「付録Ａ ウインドウ・リファレンス」の「リンク・ディレクティブ生成 ダイアログ」を参照してください。

図 2—6 セグメントの詳細（SCONST を選択した場合）

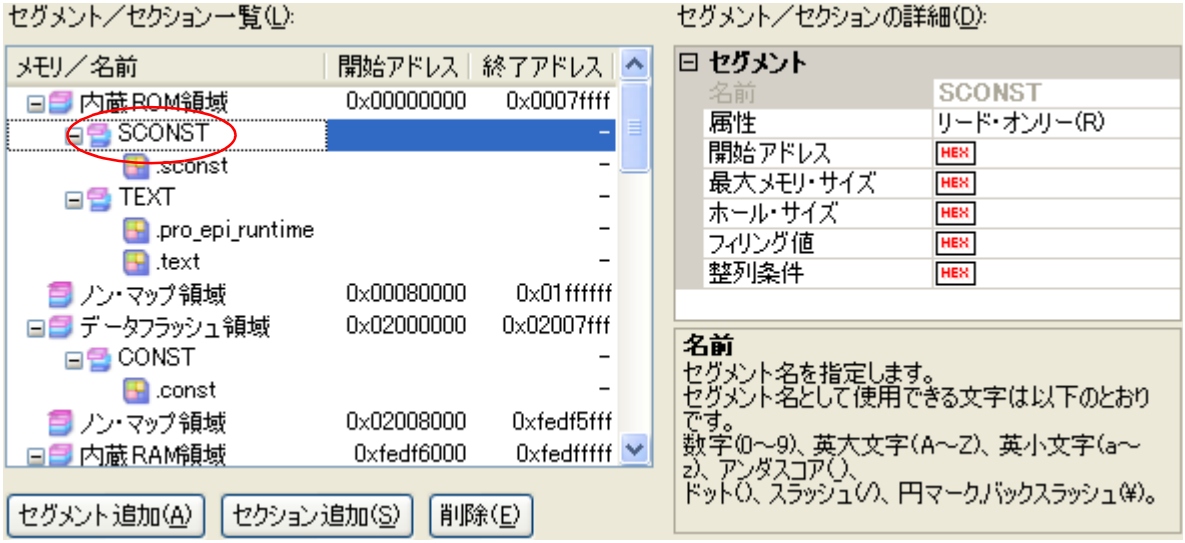
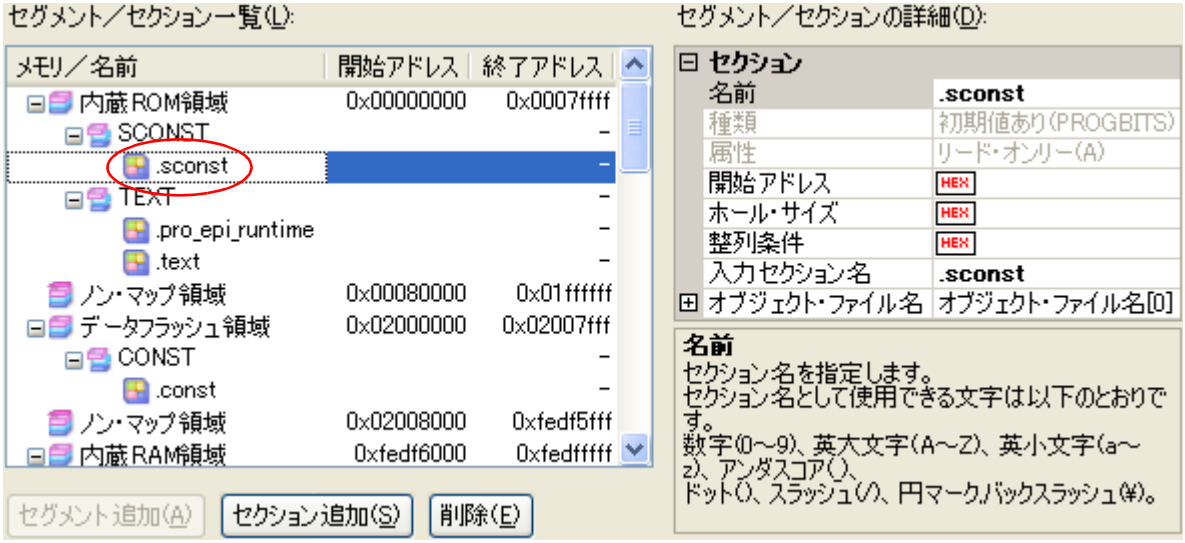


図 2—7 セクションの詳細（.sconst を選択した場合）



セグメント／セクションは，追加することもできます。

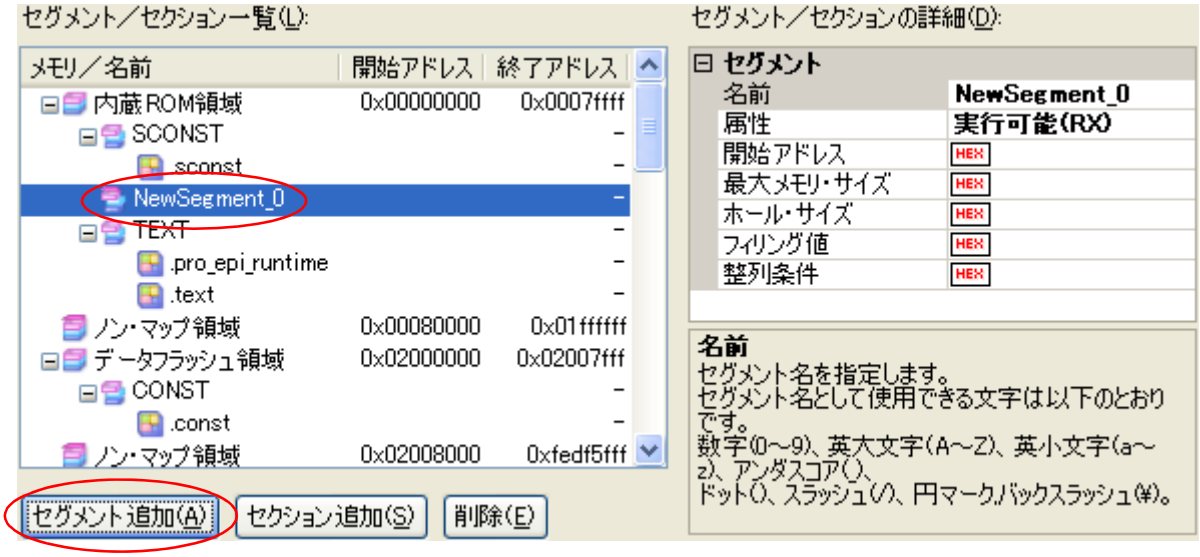
［セグメント追加］ボタンをクリックすると，一覧で選択している行の直下に新しいセグメント“NewSegment_*XXX*”を追加します（*XXX*：0 ～ 255 の 10 進数）。

［セグメント／セクションの詳細］エリアにおいて，各項目を編集してください。

デフォルトでは，［属性］に［実行可能 (RX)］（内蔵 ROM 領域，またはノン・マップ領域に追加した場合），または［リード／ライト可能 (RW)］（内蔵 RAM 領域に追加した場合），または［リード・オンリー (R)］（データフラッシュ領域に追加した場合）を選択しています。

注意　一覧でセクションの行を選択している場合，［セグメント追加］ボタンは無効となります。

図 2—8　セグメントの追加

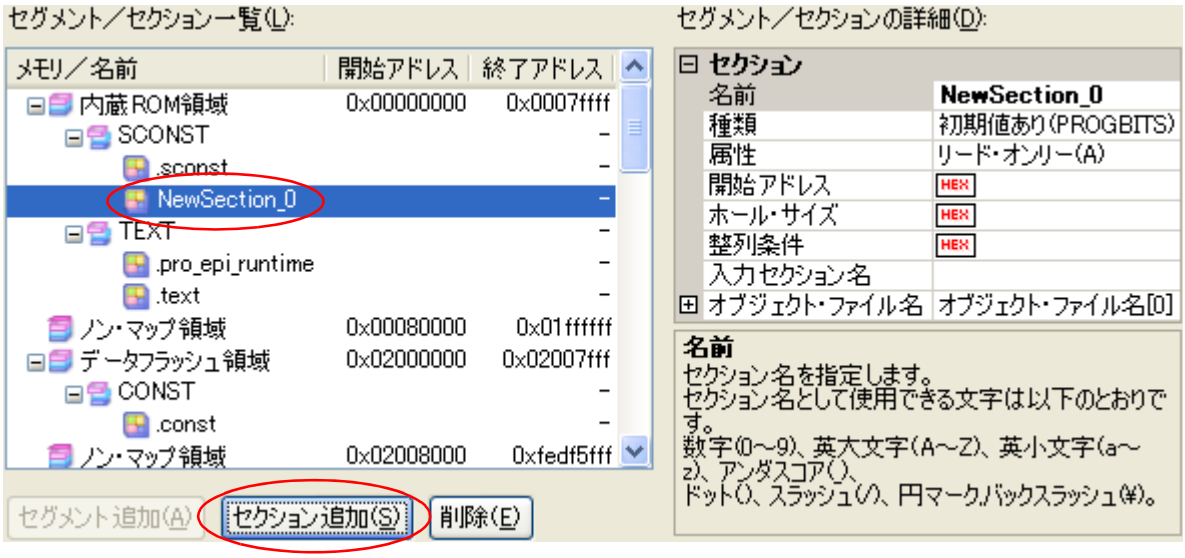


［セクション追加］ボタンをクリックすると，一覧で選択している行の直下に新しいセクション“NewSection_*XXX*”を追加します（*XXX*：0 ～ 255 の 10 進数）。

［セグメント／セクションの詳細］エリアにおいて，各項目を編集してください。

デフォルトでは，［種類］は［初期値あり (PROGBITS)］を選択し，［属性］は親セグメントの属性を引き継ぎます。

図 2—9 セクションの追加



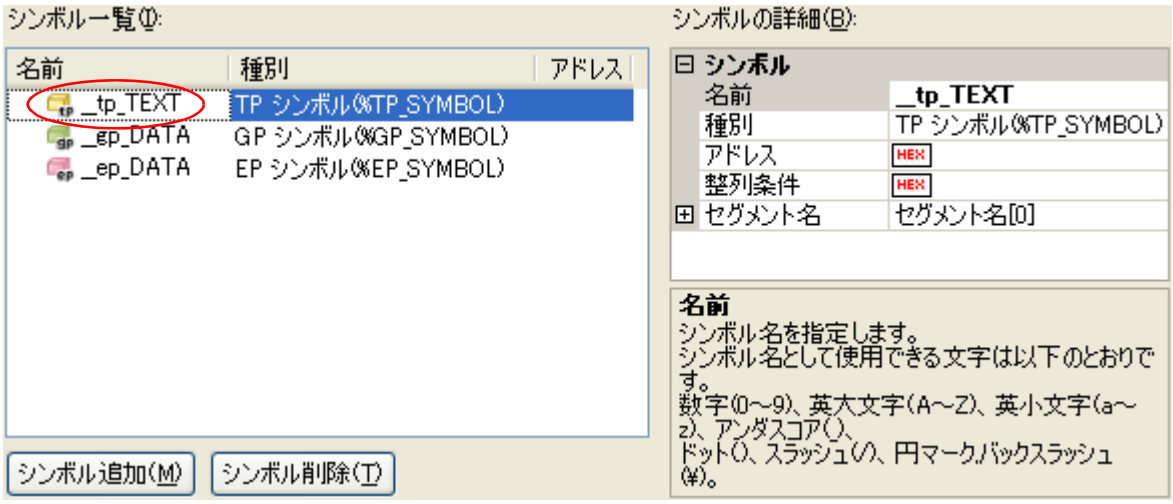
(2) シンボルの編集

［シンボル一覧］エリアには，現在設定しているシンボルの一覧を表示しています。

一覧において，シンボルを選択すると，［シンボルの詳細］エリアに該当シンボルの詳細情報を表示します。

［シンボルの詳細］エリアにおいて，各項目を編集してください。

図 2—10 シンボルの詳細（_tp_TEXT を選択した場合）



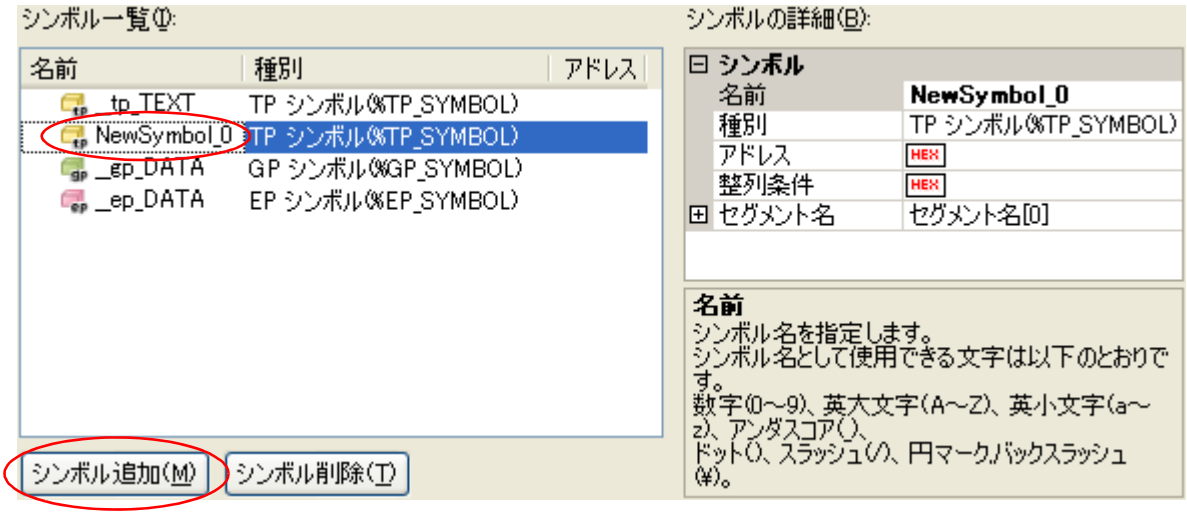
シンボルは，追加することもできます。

［シンボル追加］ボタンをクリックすると，一覧で選択している行の直下に新しいシンボル“NewSymbol_*XXX*”を追加します（*XXX*：0 ～ 255 の 10 進数）。

［シンボルの詳細］エリアにおいて，各項目を編集してください。

デフォルトでは，［種別］に［TP シンボル (%TP_SYMBOL)］を選択しています。

図 2—11　シンボルの追加



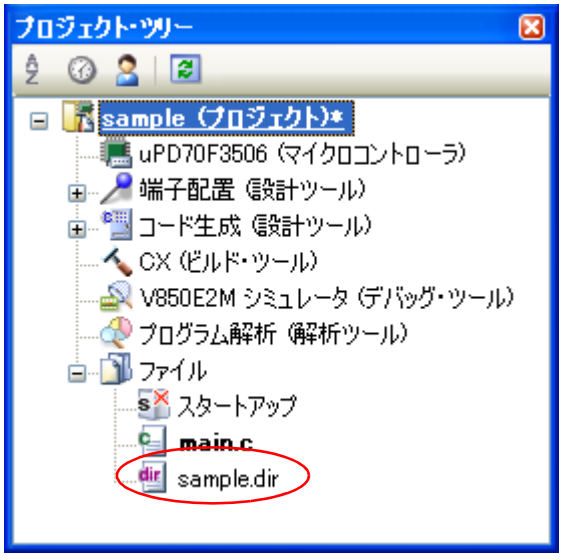
セグメント／セクション，シンボルの編集後，［生成］ボタンをクリックしてください。

指定したメモリ，セグメント／セクション，シンボルの配置情報を元に，リンク・ディレクティブ・ファイル（ファイル名：*プロジェクト名*.dir）を生成し，プロジェクトに登録します。

リンク・ディレクティブ・ファイルの生成先は，プロジェクト・フォルダとなります。

生成したリンク・ディレクティブ・ファイルは，プロジェクト・ツリーのファイル・ノードにも表示します。

図 2—12　プロジェクト・ツリー パネル（リンク・ディレクティブ・ファイル生成後）



注意　**生成したリンク・ディレクティブ・ファイルはビルド対象となります。**
すでにリンク・ディレクティブ・ファイルをプロジェクトに登録していた場合，登録済みのリンク・ディレクティブ・ファイルはビルド対象外となります。

2. 3. 3 プロジェクトにファイルを追加する

プロジェクトにファイルを追加するには，次の方法があります。

- 既存のファイルを追加する場合
- 空のファイルを作成して追加する場合

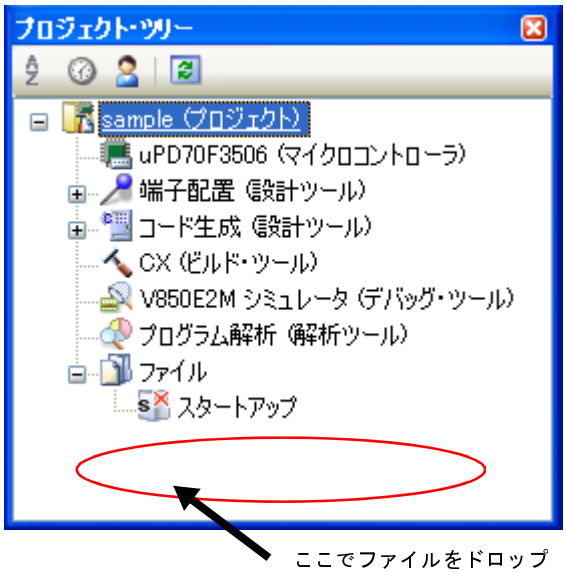
(1) 既存のファイルを追加する場合

(a) ファイル単位で追加する

エクスプローラなどからファイルをドラッグし，プロジェクト・ツリー下部の空白部分にドロップしてください。

ファイルの追加先はファイル・ノード以下となります。

図 2—13 プロジェクト・ツリー パネル（ファイルのドロップ位置）



注意 標準以外のスタートアップ・ルーチンを追加する場合は，スタートアップ・ノード上でファイルをドロップしてください。

標準以外のスタートアップ・ルーチンの使用についての詳細は，「2. 3. 1 スタートアップ・ルーチンを設定する」を参照してください。

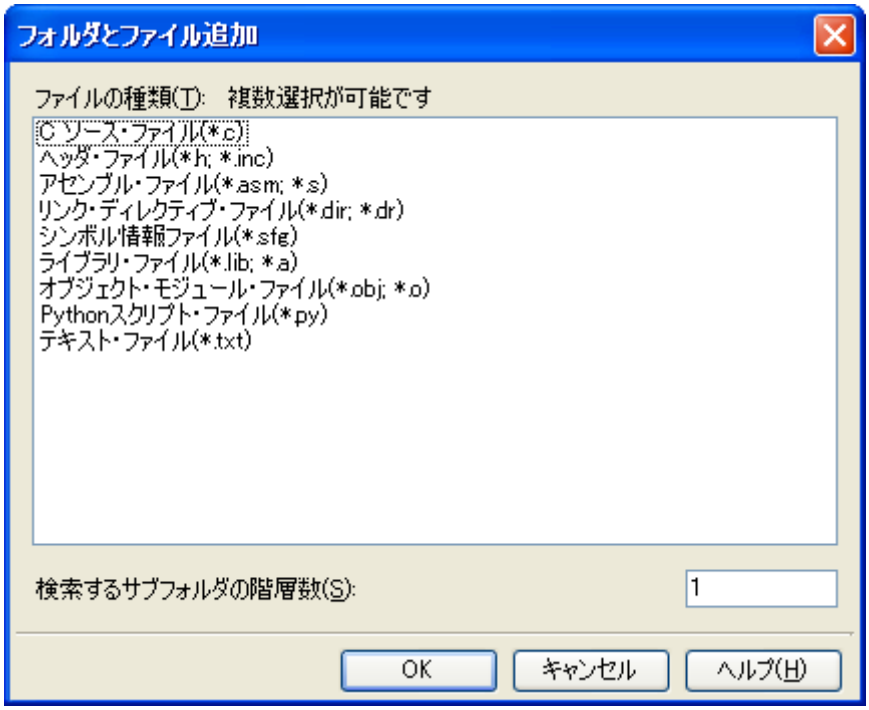
(b) フォルダ単位で追加する

エクスプローラなどからフォルダをドラッグし，プロジェクト・ツリー下部の空白部分にドロップすると，フォルダとファイル追加 ダイアログがオープンします。

備考 複数のフォルダを同時にドラッグし，プロジェクト・ツリーにドロップすることにより，複数のフォルダを同時にプロジェクトに追加することもできます。

注意 フォルダ名が 200 文字を越えるフォルダをドロップした場合，201 文字目以降は切り捨てたカテゴリ名で，プロジェクト・ツリーに追加します。

図 2—14 フォルダとファイル追加 ダイアログ



ダイアログ上で，プロジェクトに追加するファイルの種類を選択し，プロジェクトに追加するサブフォルダの階層数を指定したのち，［OK］ボタンをクリックしてください。

備考 ファイルの種類は，［Ctrl］キー＋左クリック，または［Shift］キー＋左クリックにより，複数選択することができます。
何も選択しない場合は，すべての種類を選択したものとみなします。

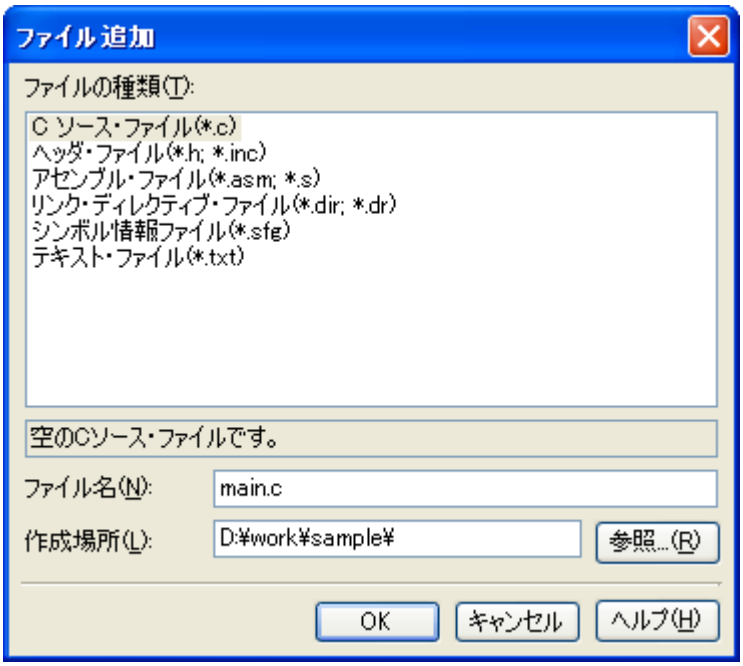
フォルダの追加先はファイル・ノード以下となります。
なお，フォルダはプロジェクト・ツリーではカテゴリとなります。

備考 ユーザが作成したカテゴリ・ノードが存在する場合，カテゴリ・ノード上でファイルをドロップすると，カテゴリ・ノード以下に追加することができます（カテゴリ・ノードについては，「2. 3. 6 ファイルをカテゴリに分類する」を参照してください）。

(2) 空のファイルを作成して追加する場合

プロジェクト・ツリー でプロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノードのいずれかを選択し，コンテキスト・メニューの［追加］→［新しいファイルを追加 ...］を選択すると，ファイル追加 ダイアログがオープンします。

図 2—15 ファイル追加 ダイアログ



ダイアログ上で，新しく作成するファイルを指定し，［OK］ボタンをクリックしてください。
ファイルの追加先はファイル・ノード以下となります。

ファイル，およびフォルダ追加後のプロジェクト・ツリーは，以下のようになります。

図 2—16 プロジェクト・ツリー パネル（ファイル main.c 追加後）

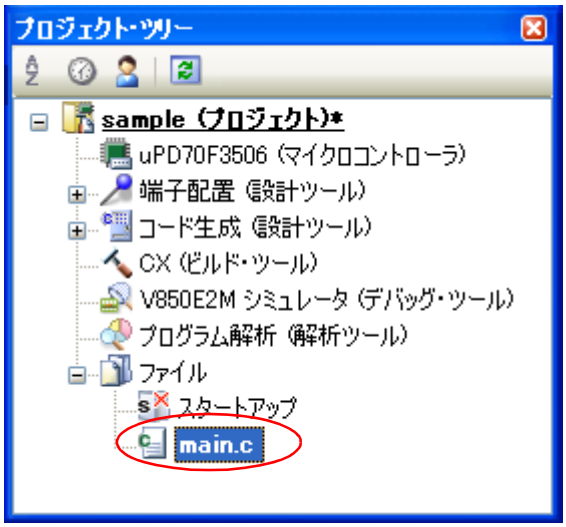
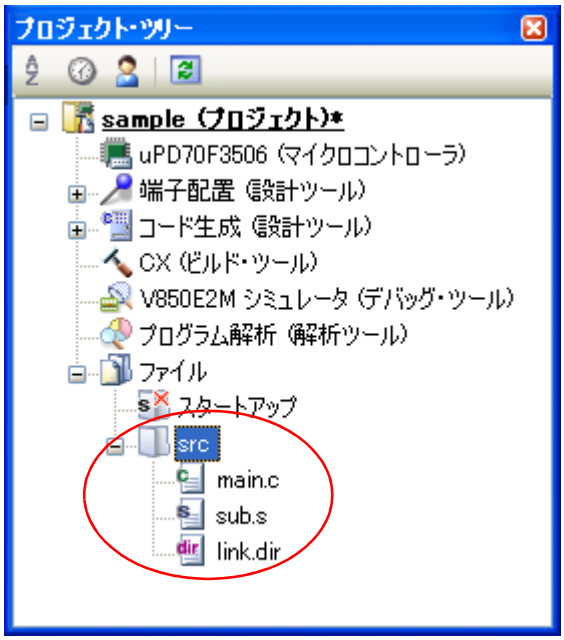


図 2—17　プロジェクト・ツリー パネル（フォルダ src 追加後）



備考　ファイル・ノード以下におけるファイルの追加位置は，現在のファイルの表示順の設定に依存します。
ファイルの表示順の変更方法については，「2. 3. 7　ファイルの表示順を変更する」を参照してください。

注意 1.　パスが異なれば，同名のソース・ファイルを追加することができます。
ただし，それらの出力ファイル名の設定がデフォルトのままの場合，出力ファイル名が同名になるため，ビルドを正しく実行することができません（例えば，D:¥sample1¥func.c，D:¥sample2¥func.c を追加した場合，これらの出力ファイル名は，デフォルトではどちらも func.obj となります）。
この問題を回避するために，ソース・ファイルの個別ビルド・オプションで，出力ファイル名をそれぞれ異なる名前に設定してください。
C ソース・ファイルの出力ファイル名の変更は，［個別コンパイル・オプション］タブの［出力ファイル］カテゴリの［オブジェクト・モジュール・ファイル名］プロパティで行います。
アセンブラ・ソース・ファイルの出力ファイル名の変更は，［個別アセンブル・オプション］タブの［出力ファイル］カテゴリの［オブジェクト・モジュール・ファイル名］プロパティで行います。
個別ビルド・オプションの設定方法については，「2. 11. 2　ファイル単位でコンパイル／アセンブル・オプションを設定する」を参照してください。

2.　同名のソース・ファイルを追加した場合，デバッグ時に対象のソースをオープンすることができません。

3.　拡張子が“dr”，“dir”のファイルをプロジェクトに追加した場合，そのファイルはリンク・ディレクティブ・ファイルとみなします。
スタートアップ・ノード以下に追加した場合もリンク・ディレクティブ・ファイルとみなします。
リンク・ディレクティブ・ファイルをプロジェクトに追加する際，すでにリンク・ディレクティブ・ファイルを追加している場合は，追加する最新のリンク・ディレクティブ・ファイルのみがビルド対象となり，追加済みのリンク・ディレクティブ・ファイルはビルド対象外となります。
ビルド対象外となっているリンク・ディレクティブ・ファイルをビルド対象に設定する際，ほかにもリ

ンク・ディレクティブ・ファイルを追加している場合は，ビルド対象に設定したリンク・ディレクティブ・ファイルのみがビルド対象となり，それ以外のリンク・ディレクティブ・ファイルはビルド対象外となります。

4. **プロジェクトに追加可能なファイル数は，メイン・プロジェクト，およびサブプロジェクトごとに 5000 個までです。**

新しいファイルを追加した場合，ファイル追加 ダイアログで指定した場所に，空のファイルを作成します。

プロジェクト・ツリー でファイル名をダブルクリックすることにより，エディタ パネルをオープンし，ファイルを編集することができます。

以下に，エディタ パネルでオープン可能なファイルを示します。

- C ソース・ファイル（*.c）
- アセンブラ・ソース・ファイル（*.asm，*.s）
- ヘッダ・ファイル（*.h，*.inc）
- シンボル情報ファイル（*.sfg）
- リンク・ディレクティブ・ファイル（*.dir，*.dr）
- リンク・マップ・ファイル（*.map）
- ヘキサ・ファイル（*.hex）
- テキスト・ファイル（*.txt）

備考 1. 以下のいずれかの方法により，上記以外のファイルもエディタ パネルでオープンすることができます。

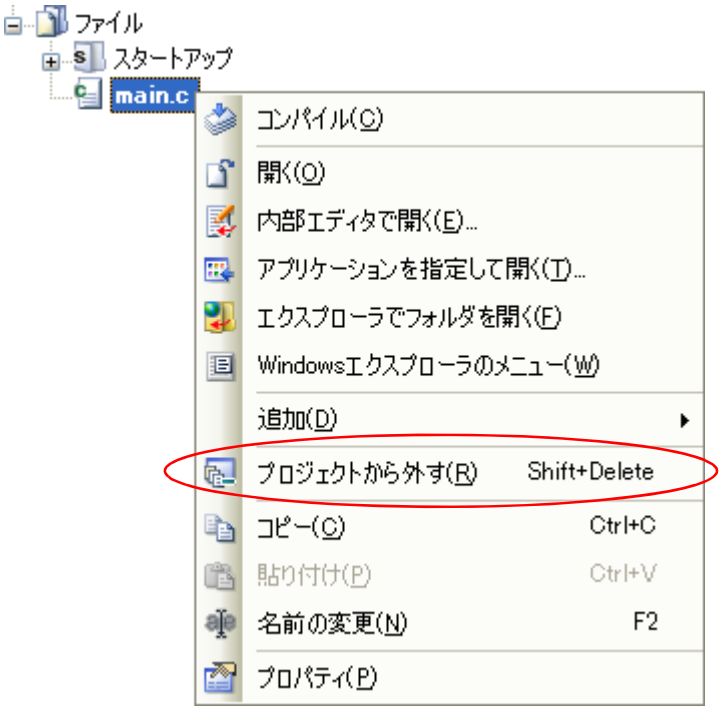
- ファイルをドラッグし，エディタ パネルにドロップする。
- ファイルを選択し，コンテキスト・メニューの［内部エディタで開く ...］を選択する。

2. オプション ダイアログで，外部テキスト・エディタを使用する設定になっている場合は，設定している外部テキスト・エディタでオープンします。
それ以外のファイルは，ホスト OS で関連付けられているアプリケーションで起動します。

2. 3. 4 プロジェクトからファイルを外す

プロジェクトに追加しているファイルをプロジェクトから外すには，プロジェクト・ツリー でプロジェクトから外すファイルを選択し，コンテキスト・メニューの［プロジェクトから外す］を選択してください。

図 2—18 ［プロジェクトから外す］項目



2. 3. 5 ファイルをビルド対象から外す

プロジェクトに追加しているファイルのうち，特定のファイルをビルド対象から外すことができます。

プロジェクト・ツリー でビルド対象から外すファイルを選択したのち，プロパティ パネルの［ビルド設定］タブを選択します。

［ビルド］カテゴリの［ビルドの対象とする］プロパティで［いいえ］を選択してください。

図 2—19 ［ビルドの対象とする］プロパティ

備考 この機能を適用できるファイルは，C ソース・ファイル，アセンブラ・ソース・ファイル，オブジェクト・モジュール・ファイル，リンク・ディレクティブ・ファイル，シンボル情報ファイル，ライブラリ・ファイルです。

2. 3. 6 ファイルをカテゴリに分類する

プロジェクトに追加しているファイルをプロジェクト・ツリー上で見やすくしたり，機能ごとに管理しやすくするために，ファイル・ノード以下にカテゴリ・ノードを作成して，ファイルを分類することができます。

カテゴリ・ノードを作成するには，プロジェクト・ツリー でプロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノードのいずれかを選択し，コンテキスト・メニューの［追加］→［新しいカテゴリを追加］を選択してください。

図 2—20 ［新しいカテゴリを追加］項目（ファイル・ノードの場合）

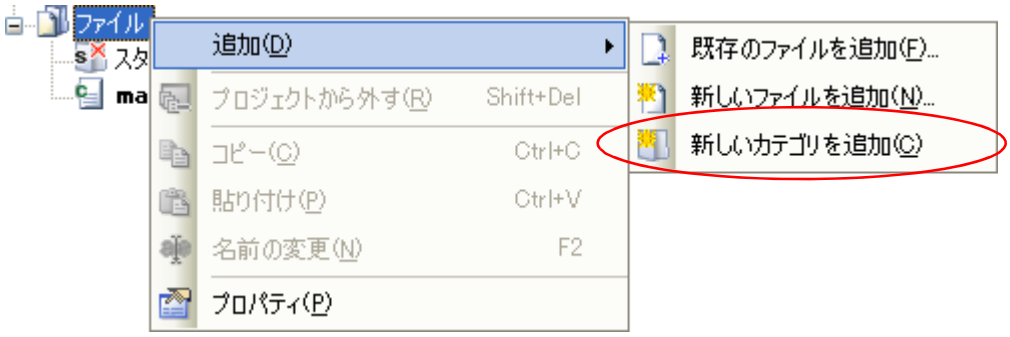
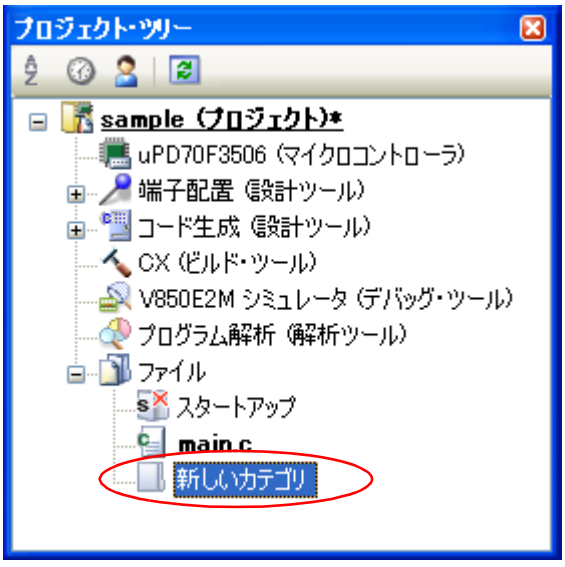


図 2—21 プロジェクト・ツリー パネル（カテゴリ・ノード追加後）



備考 1. カテゴリ名は，デフォルトで“新しいカテゴリ”となります。
カテゴリ名の変更は，カテゴリ・ノードのコンテキスト・メニューの［名前の変更］から行うことができます。

2. すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。

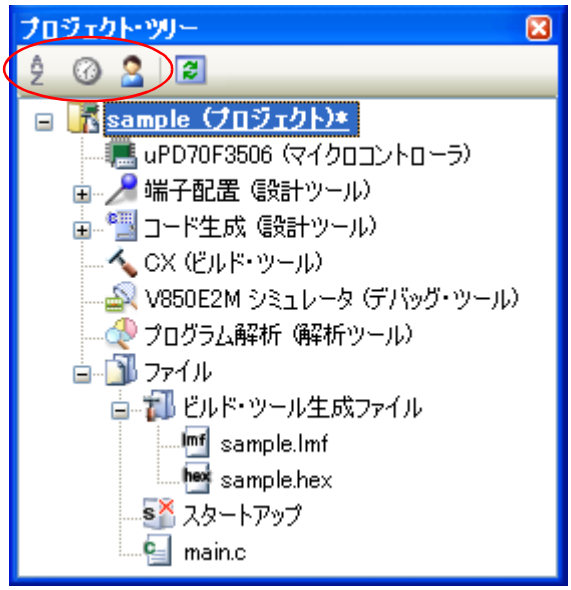
3. カテゴリのネスト数の上限は 20 です。

作成したカテゴリ・ノードにファイルを分類するには，ファイルのドラッグ・アンド・ドロップにより行うことができます。

2. 3. 7 ファイルの表示順を変更する

プロジェクト・ツリー上のボタンで，ファイル，およびカテゴリ・ノードの表示順を変更することができます。

図 2—22 ツールバー（プロジェクト・ツリー パネル）



プロジェクト・ツリー パネルのツールバーで，以下のいずれかのボタンを選択してください。

ボタン	説明
	カテゴリ・ノード，およびファイルを名前順でソートします。  : 昇順  : 降順  : 昇順
	カテゴリ・ノード，およびファイルをタイムスタンプ順でソートします。  : 降順  : 昇順  : 降順
	カテゴリ・ノードとファイルをユーザが指定した順で表示します（デフォルト）。 カテゴリ・ノード，およびファイルをドラッグ・アンド・ドロップすることにより，表示順を任意に変更することができます。

2. 3. 8 ファイルの依存関係を更新する

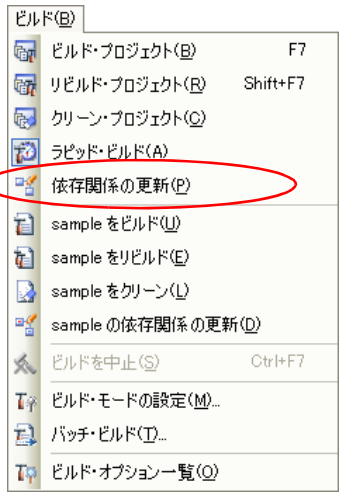
コンパイル・オプションの設定，アセンブル・オプションの設定で，ファイルの依存関係に影響する変更（インクルード・ファイルのパスの変更，ソース・ファイル中にヘッダ・ファイルのインクルード文を追加など）を行った場合は，該当ファイルの依存関係を更新する必要があります。

ファイルの依存関係の更新は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト），またはアクティブ・プロジェクトに対して行います。

(1) プロジェクト全体の場合

［ビルド］メニュー→［依存関係の更新］を選択してください。

図 2—23 ［依存関係の更新］項目



(2) アクティブ・プロジェクトの場合

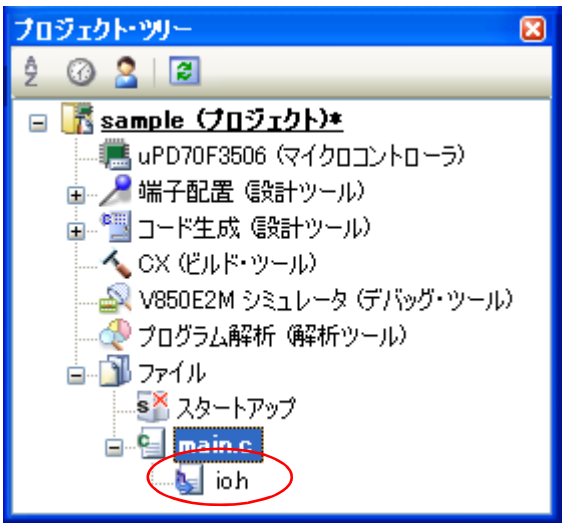
［ビルド］メニュー→［*アクティブ・プロジェクト*の依存関係の更新］を選択してください。

図 2—24 ［*アクティブ・プロジェクト*の依存関係の更新］項目

備考 ファイルの依存関係を更新する際，エディタ パネルで編集中のファイルがある場合は，該当ファイルを一括して保存します。

依存関係ファイル（インクルード・ファイル）は，プロジェクト・ツリー上のソース・ファイルにぶら下げて表示することができます。

図 2—25 プロジェクト・ツリー パネル（依存関係ファイル表示後）



なお，依存関係ファイルの表示は，以下のタイミングで更新します。

- プロジェクトを読み込んだのち，初めてビルドを実行したとき
- ツールバーの  をクリックしたとき
- ［ビルド］メニュー→［依存関係の更新］を選択したとき
- ［ビルド］メニュー→［*アクティブ・プロジェクト*の依存関係の更新］を選択したとき

備考 1. 依存関係ファイルの表示は，オプション ダイアログの［全般 - ビルド／デバッグ］カテゴリの［プロジェクト・ツリーに依存関係ファイルを表示する］がチェック状態の場合のみ有効となります。

2. プロジェクト・ツリーに表示している依存関係ファイルの情報は，プロジェクト・ファイルには保存しません。

注意 1. CubeSuite+ は，インクルード・ファイルの依存関係のチェックにおいて，#if などの条件文やコメントをサポートしません。
そのため，ビルドに不要なインクルード・ファイルを，必要なファイルであると認識するケースがあります（以下の例において，header1.h，header5.h は，ビルドに必要であると判断します）。

```
#if         0
#include    "header1.h"     /* 依存関係ありと判断する */
#else                       /* ! zero */
#include    "header2.h"     /* 依存関係あり */
#endif

#define     AAA
#ifdef      AAA
#include    "header3.h"     /* 依存関係あり */
#else
#include    "header4.h"     /* 依存関係あり */
#endif

/*
#include    "header5.h"     /* 依存関係ありと判断する */
*/
```

2. **CubeSuite+ は，インクルード・ファイルの依存関係のチェックにおいて，コメント文のあとに記述したインクルード文をサポートしません。**
   そのため，ビルドに必要なインクルード・ファイルを，不要なファイルであると認識するケースがあります（以下の例において，header6.h，header7.h は，ビルドに不要であると判断します）。

```
/* comment */   #include    "header6.h"     /* 依存関係なしと判断する */

/*
comment
*/  #include    "header7.h"                 /* 依存関係なしと判断する */
```

2.4 出力ファイルの種類を設定する

ビルドの生成物として出力するファイルの種類を設定します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［共通オプション］タブを選択します。

［出力ファイルの種類と場所］カテゴリの［出力ファイルの種類］プロパティにおいて，ファイルの種類を選択してください。

図 2—26 ［出力ファイルの種類］プロパティ

(1) ［実行形式（ロード・モジュール・ファイル）］を選択した場合（デフォルト）

ロード・モジュール・ファイル（ROM 化処理前），およびヘキサ・ファイルを生成します。

ロード・モジュール・ファイルがデバッグ対象となります。

なお，［ROM 化オプション］タブの［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合は，ROM 化用ロード・モジュール・ファイルがデバッグ対象となります。

(2) ［実行形式（ROM 化処理前ロード・モジュール・ファイル）］を選択した場合

ロード・モジュール・ファイル（ROM 化処理前），ROM 化用ロード・モジュール・ファイル，およびヘキサ・ファイルを生成します。

ロード・モジュール・ファイル（ROM 化処理前）がデバッグ対象となります。

備考 本項目は，［ROM 化オプション］タブの［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］，および［ROM 化処理前のロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合のみ表示します。

(3) ［実行形式（ヘキサ・ファイル）］を選択した場合

ロード・モジュール・ファイル（ROM 化処理前），およびヘキサ・ファイルを生成します。

ヘキサ・ファイルがデバッグ対象となります。

注意 ライブラリ用のプロジェクトの場合，本プロパティは常に［ライブラリ形式］となり，変更することはできません。

2. 4. 1　出力ファイル名を変更する

ビルド・ツールが出力するロード・モジュール・ファイル，ROM 化処理前ロード・モジュール・ファイル，ヘキサ・ファイル，ライブラリ・ファイルは，デフォルトで次の名前を設定しています。

ロード・モジュール・ファイル名	：%ProjectName%.lmf
ROM 化処理前ロード・モジュール・ファイル名	：%ProjectName%_NonROMize.lmf
ヘキサ・ファイル名	：%ProjectName%.hex
ライブラリ・ファイル名	：lib%ProjectName%.lib

備考　“%ProjectName%”はプレースホルダで，プロジェクト名に置換します。

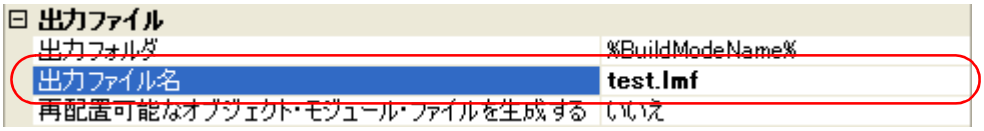
これらのファイル名の変更方法を，以下に示します。

(1) ロード・モジュール・ファイル名，および ROM 化処理前ロード・モジュール・ファイル名を変更する場合

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

［出力ファイル］カテゴリの［出力ファイル名］プロパティにおいて，変更するロード・モジュール・ファイル名を入力してください。

図 2―27　［出力ファイル名］プロパティ



本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName%	：アクティブ・プロジェクト名に置換します。
%MainProjectName%	：メイン・プロジェクト名に置換します。
%ProjectName%	：プロジェクト名に置換します。

なお，ROM 化処理前のロード・モジュール・ファイル名は，［出力ファイル名］プロパティで指定したファイル名に _NonROMize を付加した名前となります。

備考 1.　［共通オプション］タブの［よく使うオプション（リンク）］カテゴリの［出力ファイル名］プロパティでも，同様に変更することができます。

2.　ターゲットがマルチコア CPU の場合は，共通部用ロード・モジュールとコア *n* 用ロード・モジュールを生成し，それらを元に最終ロード・モジュールを生成します（*n*：ターゲット CPU が持つコアの数）。

共通部用ロード・モジュール：*拡張子を除いた入力文字列* _cmn.lmf

コア *n* 用ロード・モジュール：*拡張子を除いた入力文字列* _pe*n*.lmf

(2) ヘキサ・ファイル名を変更する場合

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ヘキサ出力オプション］タブを選択します。

［出力ファイル］カテゴリの［ヘキサ・ファイル名］プロパティにおいて，変更するヘキサ・ファイル名を入力してください。

図 2—28 ［ヘキサ・ファイル名］プロパティ

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName% ： アクティブ・プロジェクト名に置換します。
%MainProjectName% ： メイン・プロジェクト名に置換します。
%ProjectName% ： プロジェクト名に置換します。

(3) ライブラリ・ファイル名を変更する場合

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ライブラリ生成オプション］タブを選択します。

［出力ファイル］カテゴリの［生成ファイル名］プロパティにおいて，変更するライブラリ・ファイル名を入力してください。

図 2—29 ［生成ファイル名］プロパティ

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName% ： アクティブ・プロジェクト名に置換します。
%MainProjectName% ： メイン・プロジェクト名に置換します。
%ProjectName% ： プロジェクト名に置換します。

2. 4. 2 アセンブル・リストを出力する

アセンブル・リスト（アセンブル結果のコード）は，アセンブル・リスト・ファイルに出力します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。

アセンブル・リスト・ファイルを出力するには，［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択してください。

図 2—30 ［アセンブル・リスト・ファイルを出力する］プロパティ

アセンブル・リスト・ファイルを出力する場合，出力フォルダを設定することができます。

（1）出力フォルダの設定

［アセンブル・リスト・ファイル出力フォルダ］プロパティにおいて，テキスト・ボックスへの直接入力，または［...］ボタンにより行います。

本プロパティは，次のプレースホルダに対応しています。

%BuildModeName% ：ビルド・モード名に置換します。

デフォルトでは，“%BuildModeName%”を設定しています。

なお，ファイル名は，ソース・ファイルの拡張子を .prn で置き換えた名前となります。

備考 アセンブル・リスト・ファイルについての詳細は，「3. 1 アセンブル・リスト・ファイル」を参照してください。

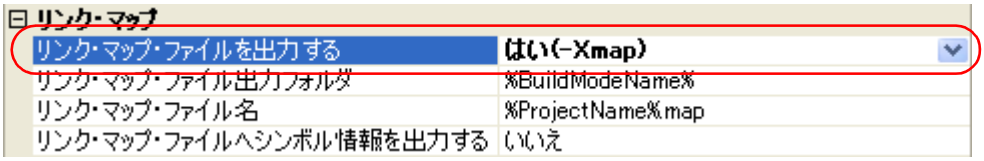
2. 4. 3 マップ情報を出力する

マップ情報（リンク結果の情報）は，リンク・マップ・ファイルに出力します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

リンク・マップ・ファイルを出力するには，［リンク・マップ］カテゴリの［リンク・マップ・ファイルを出力する］プロパティで［はい (-Xmap)］を選択してください。

図 2—31 ［リンク・マップ・ファイルを出力する］プロパティ



リンク・マップ・ファイルを出力する場合，出力フォルダ，および出力ファイル名を設定することができます。

(1) 出力フォルダの設定

［リンク・マップ・ファイル出力フォルダ］プロパティにおいて，テキスト・ボックスへの直接入力，または［...］ボタンにより行います。

本プロパティは，次のプレースホルダに対応しています。

%BuildModeName% ： ビルド・モード名に置換します。

デフォルトでは，“%BuildModeName%”を設定しています。

(2) 出力ファイル名の設定

［リンク・マップ・ファイル名］プロパティにおいて，テキスト・ボックスへの直接入力により行います。

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName% ： アクティブ・プロジェクト名に置換します。

%MainProjectName% ： メイン・プロジェクト名に置換します。

%ProjectName% ： プロジェクト名に置換します。

デフォルトでは，“%ProjectName%.map”を設定しています。

備考 リンク・マップ・ファイルについての詳細は，「3. 2 リンク・マップ・ファイル」を参照してください。

2. 4. 4　シンボル情報を出力する

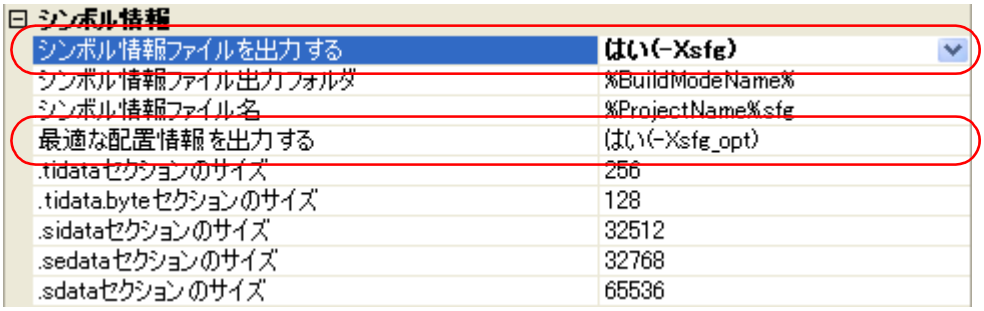
シンボル情報（変数の配置セクション情報）は，シンボル情報ファイルに出力します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

シンボル情報ファイルを出力するには，［シンボル情報］カテゴリの［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択してください。

また，セクション単位で変数の最適な配置情報を出力するために，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択します。

図 2―32　［シンボル情報ファイルを出力する］，および［最適な配置情報を出力する］プロパティ



シンボル情報ファイルを出力する場合，出力フォルダ，および出力ファイル名を設定することができます。

(1) 出力フォルダの設定

［シンボル情報ファイル出力フォルダ］プロパティにおいて，テキスト・ボックスへの直接入力，または［...］ボタンにより行います。

本プロパティは，次のプレースホルダに対応しています。

%BuildModeName%　：ビルド・モード名に置換します。

デフォルトでは，“%BuildModeName%”を設定しています。

(2) 出力ファイル名の設定

［シンボル情報ファイル名］プロパティにおいて，テキスト・ボックスへの直接入力により行います。

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName%　：アクティブ・プロジェクト名に置換します。

%MainProjectName%　：メイン・プロジェクト名に置換します。

%ProjectName%　：プロジェクト名に置換します。

デフォルトでは，“%ProjectName%.sfg”を設定しています。

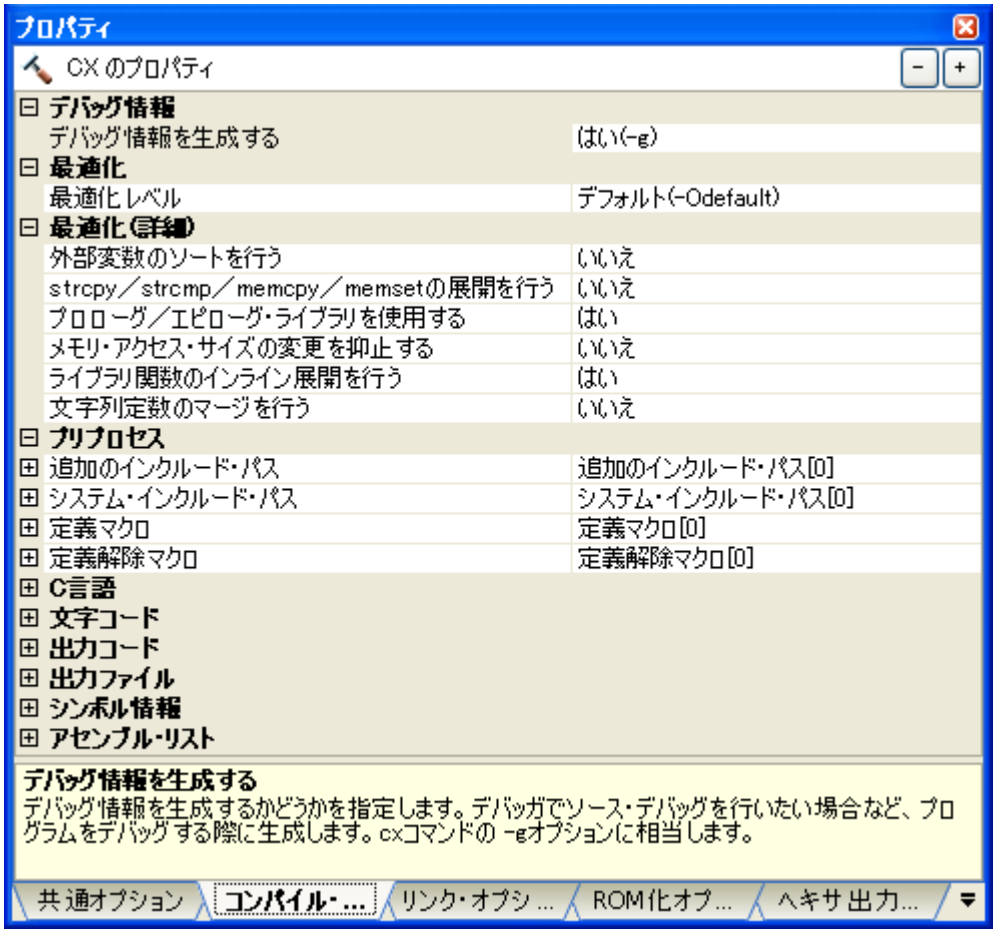
備考　シンボル情報ファイルについての詳細は，「3. 3　シンボル情報ファイル」を参照してください。

2.5 コンパイル・オプションを設定する

コンパイル・フェーズに対するオプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択してください。

タブ上で各プロパティを設定することにより，対応するコンパイル・オプションを設定することができます。

図 2—33 プロパティ パネル：［コンパイル・オプション］タブ



備考 よく使うオプションについては，［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリにまとめられています。

2.5.1 コード・サイズを優先した最適化を行う

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。

コード・サイズを優先した最適化を行うには，［最適化］カテゴリの［最適化レベル］プロパティで［サイズ優先(-Osize)］を選択してください（デフォルト：［デフォルト (-Odefault)］）。

図 2—34 ［最適化レベル］プロパティ（コード・サイズ優先の場合）

備考 1. ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［最適化レベル］プロパティでも，同様に設定することができます。

2. 最適化機能についての詳細は，「B. 1. 5 最適化機能」を参照してください。

2. 5. 2 実行速度を優先した最適化を行う

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。

実行速度を優先した最適化を行うには，［最適化］カテゴリの［最適化レベル］プロパティで［実行速度優先 (-Ospeed)］を選択してください（デフォルト：［デフォルト (-Odefault)］）。

図 2—35 ［最適化レベル］プロパティ（実行速度優先の場合）

備考 1. ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［最適化レベル］プロパティでも，同様に設定することができます。

2. 最適化機能についての詳細は，「B. 1. 5 最適化機能」を参照してください。

2. 5. 3 インクルード・パスを追加する

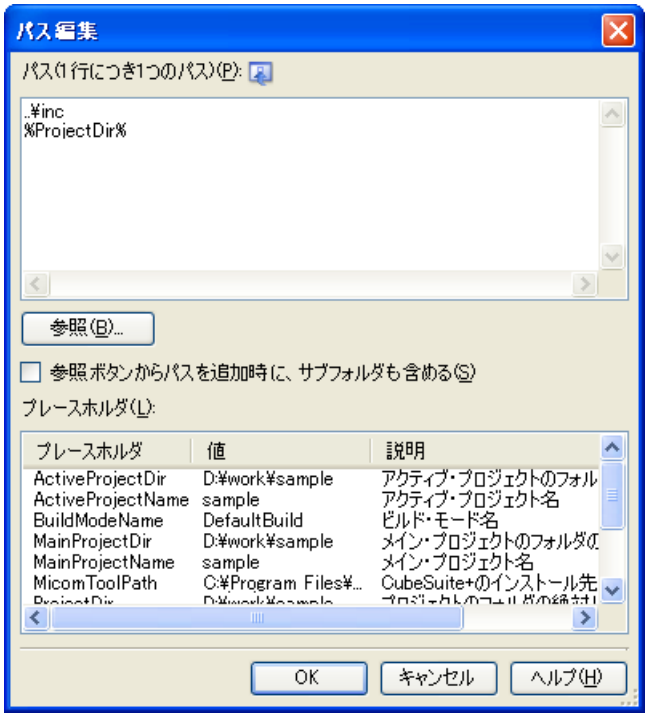
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。

インクルード・パスの設定は，［プリプロセス］カテゴリの［追加のインクルード・パス］プロパティで行います。

図 2—36 ［追加のインクルード・パス］プロパティ

［...］ボタンをクリックすると，パス編集 ダイアログがオープンします。

図 2—37 パス編集 ダイアログ



［パス (1 行につき 1 つのパス)］にインクルード・パスを 1 行に 1 つずつ入力します。
1 行に 259 文字まで，256 行まで指定可能です。

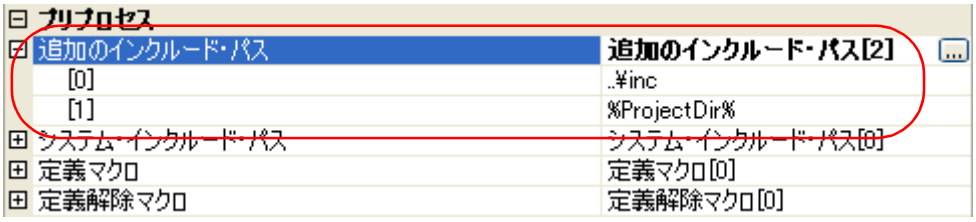
備考 1. インクルード・パスは，以下のいずれかの方法で指定することも可能です。

- エクスプローラなどからフォルダをドラッグ・アンド・ドロップ
- ［参照 ...］ボタンをクリックし，フォルダの参照 ダイアログによるフォルダの選択
- ［プレースホルダ］において行をダブルクリック

2. ［参照ボタンからパスを追加時に、サブフォルダも含める］をチェックしたのち，［参照 ...］ボタンからパスの指定を行うと，指定したパスとそのサブフォルダ 5 階層分までのパスを［パス (1 行につき 1 つのパス)］に追加します。

［OK］ボタンをクリックすると，入力したインクルード・パスをサブプロパティとして表示します。

図 2—38 ［追加のインクルード・パス］プロパティ（インクルード・パス追加後）



インクルード・パスの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。

また，プロジェクト・ツリーにインクルード・ファイルを追加すると，そのインクルード・パスをサブプロパティの一番最初に自動で追加します。

備考 ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［追加のインクルード・パス］プロパティでも，同様に設定することができます。

2. 5. 4 定義マクロを設定する

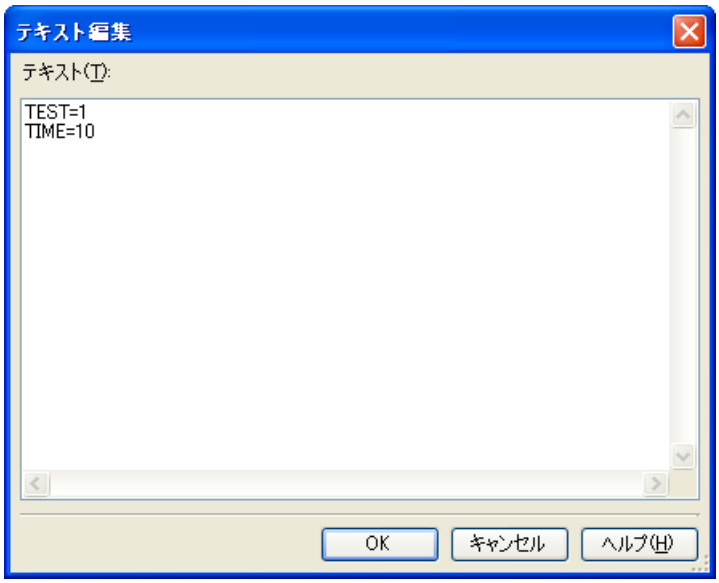
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。

定義マクロの設定は，［プリプロセス］カテゴリの［定義マクロ］プロパティで行います。

図 2—39 ［定義マクロ］プロパティ

［...］ボタンをクリックすると，テキスト編集 ダイアログがオープンします。

図 2—40 テキスト編集 ダイアログ



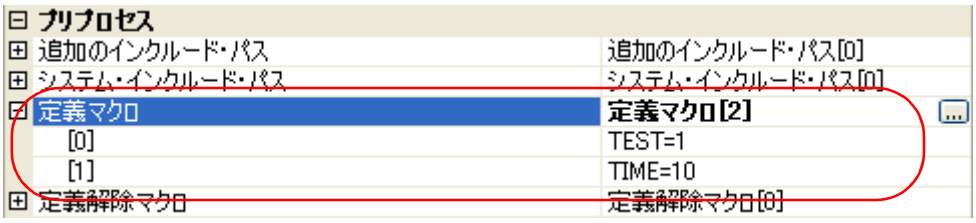
［テキスト］に定義マクロを「*マクロ名 = 定義値*」の形式で 1 行に 1 つずつ入力します。

1 行に 256 文字まで，256 行まで指定可能です。

「= *定義値*」の部分は省略可能で，省略した場合，定義値を 1 とします。

［OK］ボタンをクリックすると，入力した定義マクロをサブプロパティとして表示します。

図 2—41 ［定義マクロ］プロパティ（定義マクロ設定後）



定義マクロの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。

備考 ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［定義マクロ］プロパティでも，同様に設定することができます。

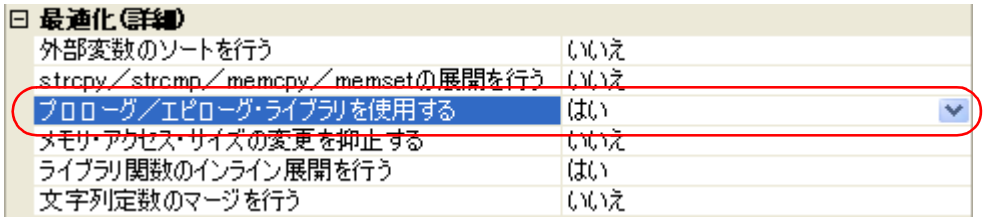
2. 5. 5 コード・サイズを削減する（プロローグ／エピローグ・ランタイム呼び出しを行う）

関数のプロローグ／エピローグ処理の一部をランタイム・ライブラリ呼び出しによる処理に変更することで，コード・サイズを削減することができます。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。

関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にするには，［最適化 (詳細)］カテゴリの［プロローグ／エピローグ・ライブラリを使用する］プロパティで，［はい］を選択してください。

図 2—42 ［プロローグ／エピローグ・ライブラリを使用する］プロパティ



2. 5. 6　レジスタ・モードを変更する

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［共通オプション］タブを選択します。

［レジスタ・モード］カテゴリの［レジスタ・モード］プロパティで，レジスタ・モードを変更してください。

図 2―43　［レジスタ・モード］プロパティ

以下のレジスタ・モードを選択することができます。

レジスタ・モード	作業用レジスタ	レジスタ変数用レジスタ
32 レジスタ・モード (なし) （デフォルト）	r10 ～ r19	r20 ～ r29
26 レジスタ・モード (-Xreg_mode=26)	r10 ～ r16	r23 ～ r29
22 レジスタ・モード (-Xreg_mode=22)	r10 ～ r14	r25 ～ r29
汎用レジスタ・モード (-Xreg_mode=common)	r10 ～ r14	r25 ～ r29

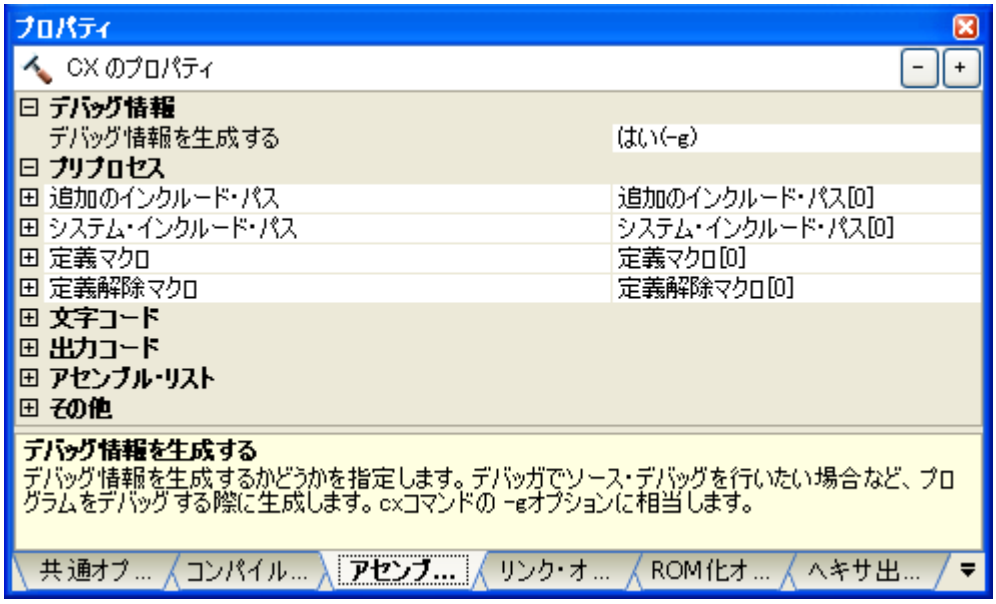
備考　レジスタ・モードについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

2. 6 アセンブル・オプションを設定する

アセンブル・フェーズに対するオプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［アセンブル・オプション］タブを選択してください。

タブ上で各プロパティを設定することにより，対応するアセンブル・オプションを設定することができます。

図 2—44 プロパティ パネル：［アセンブル・オプション］タブ



備考 よく使うオプションについては，［共通オプション］タブの［よく使うオプション（アセンブル）］カテゴリにまとめられています。

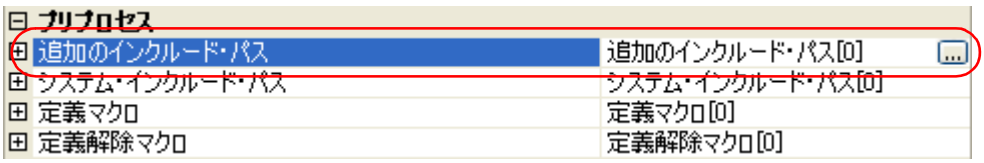
注意 上記のタブは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。

2. 6. 1 インクルード・パスを追加する

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［アセンブル・オプション］タブを選択します。

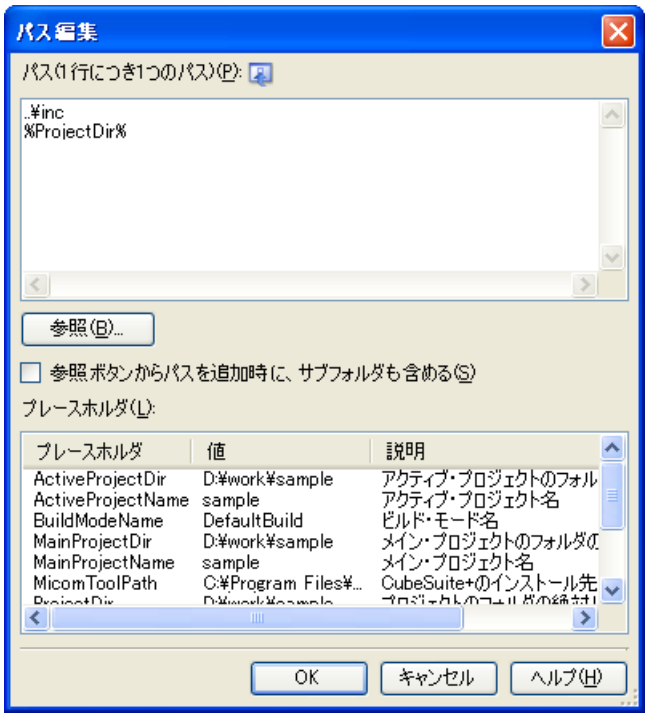
インクルード・パスの設定は，［プリプロセス］カテゴリの［追加のインクルード・パス］プロパティで行います。

図 2—45 ［追加のインクルード・パス］プロパティ



［...］ボタンをクリックすると，パス編集 ダイアログがオープンします。

図 2—46 パス編集 ダイアログ



［パス (1 行につき 1 つのパス)］にインクルード・パスを 1 行に 1 つずつ入力します。
1 行に 259 文字まで，256 行まで指定可能です。

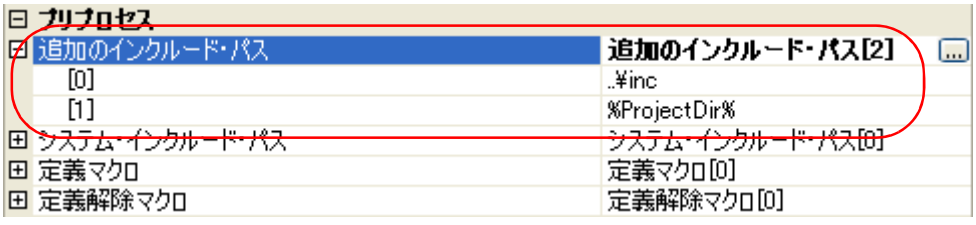
備考 1. インクルード・パスは，以下のいずれかの方法で指定することも可能です。

- エクスプローラなどからフォルダをドラッグ・アンド・ドロップ
- ［参照 ...］ボタンをクリックし，フォルダの参照 ダイアログによるフォルダの選択
- ［プレースホルダ］において行をダブルクリック

2. ［参照ボタンからパスを追加時に、サブフォルダも含める］をチェックしたのち，［参照 ...］ボタンからパスの指定を行うと，指定したパスとそのサブフォルダ 5 階層分までのパスを［パス (1 行につき 1 つのパス)］に追加します。

［OK］ボタンをクリックすると，入力したインクルード・パスをサブプロパティとして表示します。

図 2—47 ［追加のインクルード・パス］プロパティ（インクルード・パス追加後）



インクルード・パスの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。

また，プロジェクト・ツリーにインクルード・ファイルを追加すると，そのインクルード・パスをサブプロパティの一番最初に自動で追加します。

備考 ［共通オプション］タブの［よく使うオプション（アセンブル）］カテゴリの［追加のインクルード・パス］プロパティでも，同様に設定することができます。

2. 6. 2 定義マクロを設定する

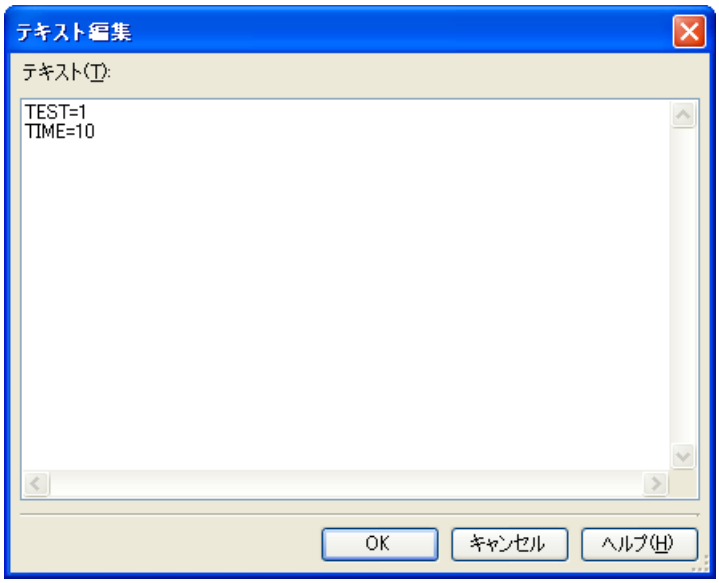
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［アセンブル・オプション］タブを選択します。

定義マクロの設定は，［プリプロセス］カテゴリの［定義マクロ］プロパティで行います。

図 2—48 ［定義マクロ］プロパティ

［...］ボタンをクリックすると，テキスト編集 ダイアログがオープンします。

図 2—49 テキスト編集 ダイアログ



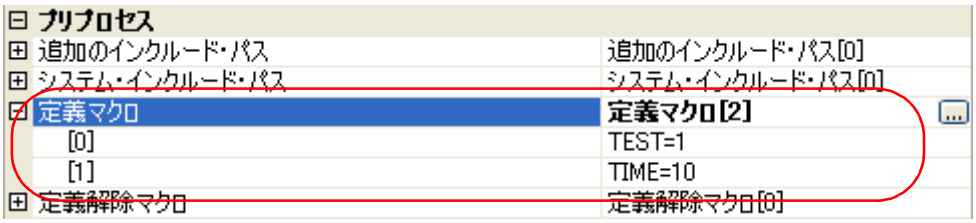
［テキスト］に定義マクロを「*マクロ名* = *定義値*」の形式で 1 行に 1 つずつ入力します。

1 行に 256 文字まで，256 行まで指定可能です。

「= *定義値*」の部分は省略可能で，省略した場合，定義値を 1 とします。

［OK］ボタンをクリックすると，入力した定義マクロをサブプロパティとして表示します。

図 2—50 ［定義マクロ］プロパティ（定義マクロ設定後）



定義マクロの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。

備考 ［共通オプション］タブの［よく使うオプション（アセンブル）］カテゴリの［定義マクロ］プロパティでも，同様に設定することができます。

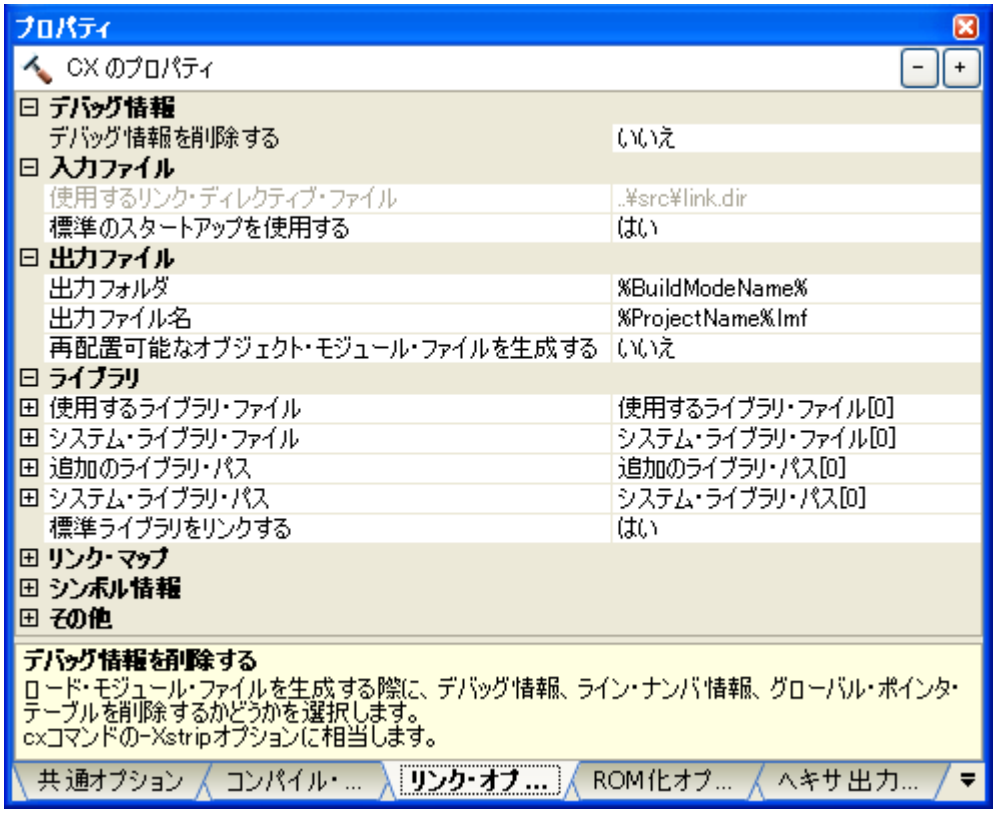
2.7 リンク・オプションを設定する

リンク・フェーズに対するオプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択してください。

タブ上で各プロパティを設定することにより，対応するリンク・オプションを設定することができます。

注意　本タブは，ライブラリ用のプロジェクトの場合は表示しません。

図 2—51　プロパティ パネル：［リンク・オプション］タブ



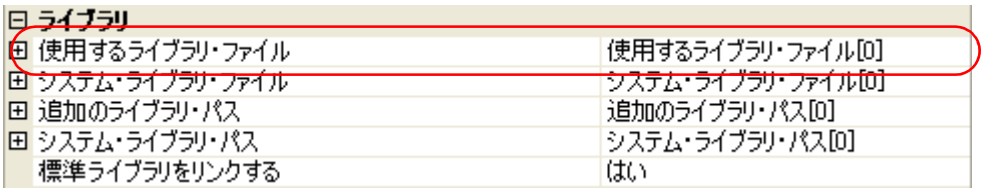
備考　よく使うオプションについては，［共通オプション］タブの［よく使うオプション（リンク）］カテゴリにまとめられています。

2. 7. 1 ユーザ・ライブラリを追加する

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

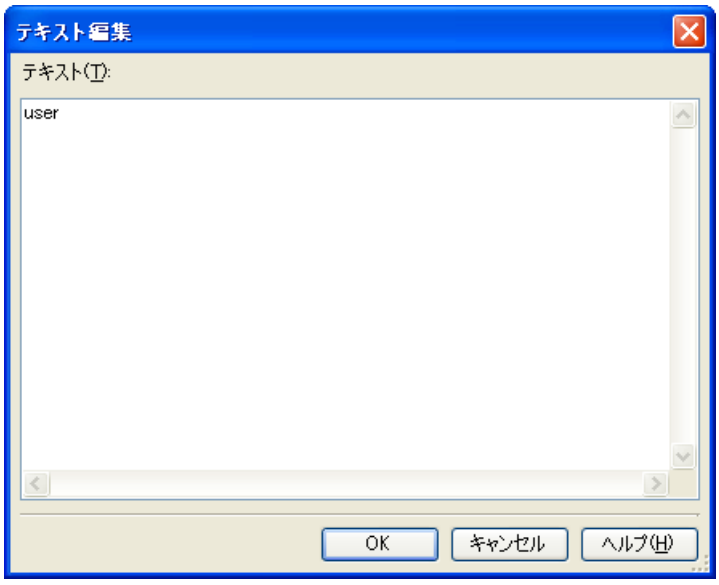
ユーザ・ライブラリの追加は，［ライブラリ］カテゴリの［使用するライブラリ・ファイル］プロパティで行います。

図 2—52 ［使用するライブラリ・ファイル］プロパティ



［...］ボタンをクリックすると，テキスト編集 ダイアログがオープンします。

図 2—53 テキスト編集 ダイアログ



［テキスト］にライブラリ・ファイル名“lib*xxx*.lib”，または“libxxx.a”のうち，*xxx* のみを指定します（例：“user”と指定すると，libuser.lib，または libuser.a を指定したものとみなします）。

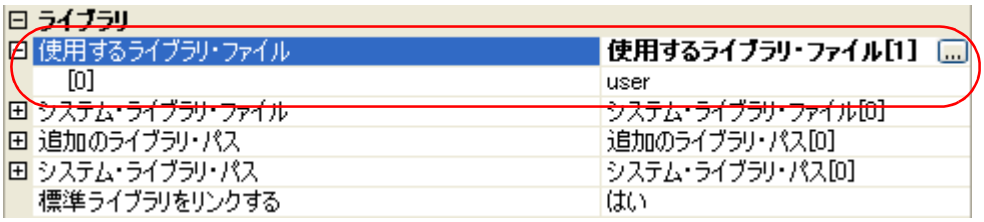
1 行に 1 つずつ入力します。

1 行に 249 文字まで，256 行まで指定可能です。

備考 lib*xxx*.lib を優先して検索して，見つからなかった場合に lib*xxx*.a を検索します。

［OK］ボタンをクリックすると，入力したライブラリ・ファイルをサブプロパティとして表示します。

図 2—54 ［使用するライブラリ・ファイル］プロパティ（ライブラリ・ファイル設定後）



ライブラリ・ファイルの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。

備考 ［共通オプション］タブの［よく使うオプション（リンク）］カテゴリの［使用するライブラリ・ファイル］プロパティでも，同様に設定することができます。

なお，ライブラリ・ファイルはライブラリ・パスから検索します。
ライブラリ・パスを追加する場合は，［追加のライブラリ・パス］プロパティを設定してください。

**注意 ライブラリ・ファイルは，プロジェクトに直接追加することでもリンクします。
その場合は，追加したライブラリ・ファイルの絶対パスで直接リンクするため，ライブラリ・パスからは検索しません。**

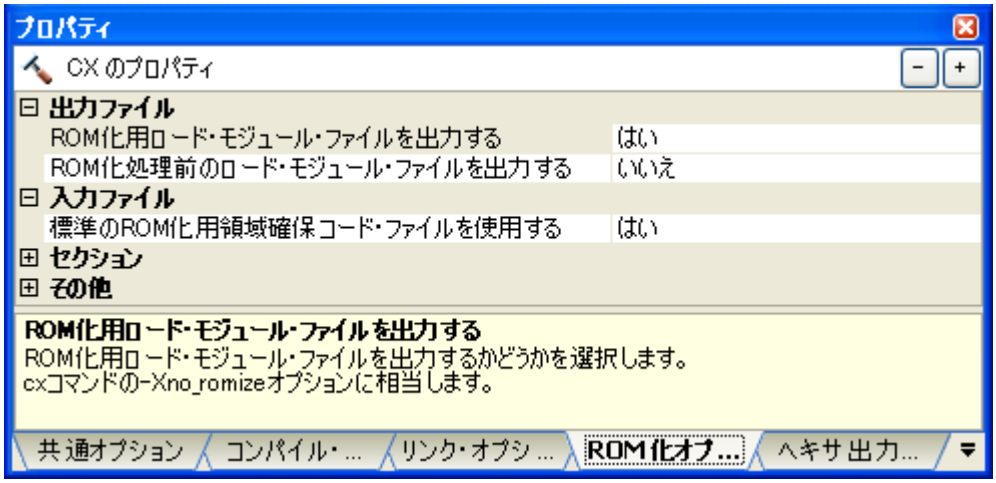
2. 8 ROM 化オプションを設定する

ROM 化フェーズに対するオプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ROM 化オプション］タブを選択してください。

タブ上で各プロパティを設定することにより，対応する ROM 化オプションを設定することができます。

注意 本タブは，ライブラリ用のプロジェクトの場合は表示しません。

図 2—55 プロパティ パネル：［ROM 化オプション］タブ



2. 8. 1 ROM 化用ロード・モジュールを作成する

デフォルトで用意されている ROM 化用領域確保コード・ファイル（rompcrt.obj）を使用して，ROM 化用ロード・モジュールを作成する方法を以下に示します。

ROM 化プロセッサは，data 属性セクションの変数の初期値情報や RAM 上に配置するプログラムを，1 つのセクションにパッキングします。

デフォルトでは，このセクションは“rompsec セクション”となります。

rompsec セクションを ROM 上に配置し，コピー関数を呼び出すことによって，初期値情報やプログラムを RAM 上へ展開することができます。

備考 ROM 化用ロード・モジュールの作成方法についての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

（1）コピー関数の呼び出し

スタートアップ・ルーチン内でコピー関数 _rcopy を呼び出します。

標準スタートアップ・ルーチン（cstart.obj）には，すでに _rcopy の呼び出しを記述しているので，それをそのまま使用することができます。

_rcopy の代わりに _rcopy2，_rcopy4 を使用したい場合，またはコピーするセクションを指定したい場合は，標準スタートアップ・ルーチンをカスタマイズする必要があります。

備考 1. コピー関数についての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

2. 標準以外のスタートアップ・ルーチンを使用する場合は，使用するファイルをプロジェクト・ツリーのスタートアップ・ノードに追加してください。
スタートアップ・ルーチンの設定についての詳細は，「2. 3. 1 スタートアップ・ルーチンを設定する」を参照してください。

(2) ROM 化オプションの設定

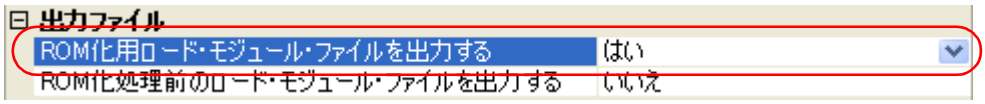
ROM 化オプションにより，ROM 化用ロード・モジュールの生成を指定します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ROM 化オプション］タブを選択します。

(a) ROM 化用ロード・モジュール・ファイルの出力設定

［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択してください。

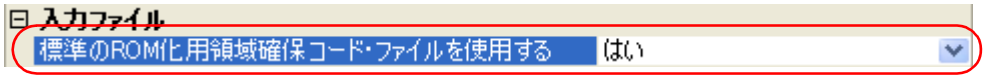
図 2—56 ［ROM 化用ロード・モジュール・ファイルを出力する］プロパティ



(b) 標準の ROM 化用領域確保コード・ファイルの使用設定

［入力ファイル］カテゴリの［標準の ROM 化用領域確保コード・ファイルを使用する］プロパティで［はい］（デフォルト）を選択してください。

図 2—57 ［標準の ROM 化用領域確保コード・ファイルを使用する］プロパティ



(3) ビルドの実行

ビルドを実行することにより，ROM 化用ロード・モジュール・ファイルを生成します。

その際，以下の順番でファイルをリンクします。

- スタートアップ・ルーチンのオブジェクト・モジュール・ファイル（cstart.obj）
- コピー関数が格納されているライブラリ・ファイル（libc.lib）
- ROM 化用領域確保コード・ファイル（rompcrt.obj）

備考 ROM 化用領域確保コード・ファイルは，最後にリンクする必要があります。
しかし，［リンク・オプション］タブの［その他］カテゴリの［ライブラリ・ファイルを再スキャンする］プロパティで［はい (-Xrescan)］を選択した場合は，ROM 化用領域確保コード・ファイルの後にライブラリ・ファイルをリンクし，ROM 化処理中にエラーとなることがあります。

その場合は，rompsec セクションの領域を明示的に確保してください。
詳細については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

なお，本製品は，ROM 化処理を行ったのち，デフォルトでヘキサ出力を行うため，ヘキサ・ファイルも生成します。

ROM ライタを使用して，生成したヘキサ・ファイルをターゲットにロードしてください。

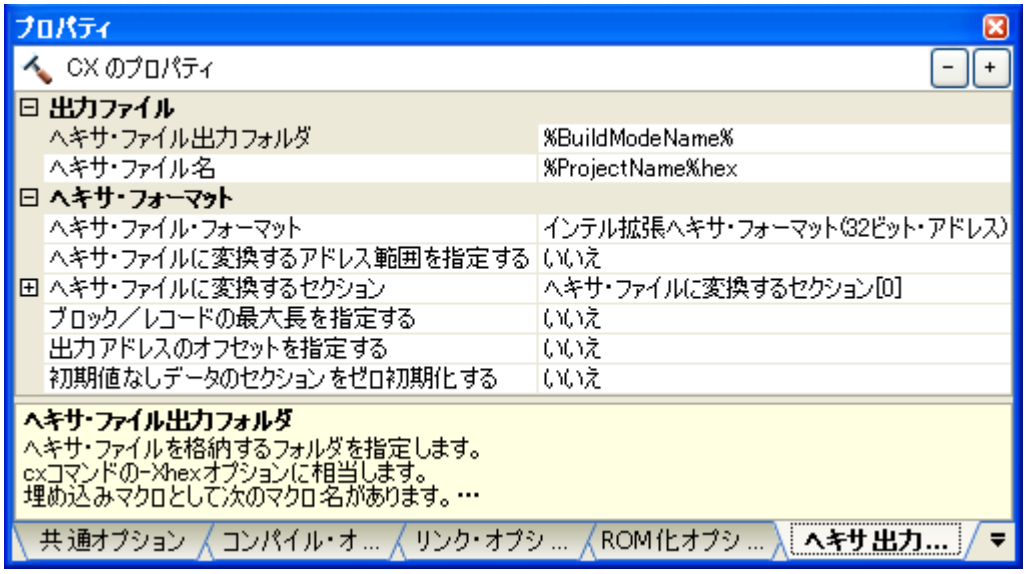
2. 9 ヘキサ出力オプションを設定する

ヘキサ出力フェーズに対するオプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ヘキサ出力オプション］タブを選択してください。

タブ上で各プロパティを設定することにより，対応するヘキサ出力オプションを設定することができます。

注意　本タブは，ライブラリ用のプロジェクトの場合は表示しません。

図 2—58　プロパティ パネル：［ヘキサ出力オプション］タブ



備考　よく使うオプションについては，［共通オプション］タブの［よく使うオプション（ヘキサ出力）］カテゴリにまとめられています。

2. 9. 1 ヘキサ・ファイルの出力を設定する

ヘキサ・ファイルは，ロード・モジュール・ファイルの生成後，デフォルトで生成します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ヘキサ出力オプション］タブを選択します。

ヘキサ・ファイルの出力先の設定は，［出力ファイル］カテゴリで行います。

図 2—59　［出力ファイル］カテゴリ

(1) 出力フォルダの設定

［ヘキサ・ファイル出力フォルダ］プロパティにおいて，テキスト・ボックスへの直接入力，または［...］ボタンにより行います。

テキスト・ボックスには 247 文字まで指定可能です。

本プロパティは，次のプレースホルダに対応しています。

%BuildModeName% ： ビルド・モード名に置換します。

デフォルトでは，“%BuildModeName%”を設定しています。

(2) 出力ファイル名の設定

［ヘキサ・ファイル名］プロパティにおいて，テキスト・ボックスへの直接入力により行います。

テキスト・ボックスには 259 文字まで指定可能です。

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName% ： アクティブ・プロジェクト名に置換します。

%MainProjectName% ： メイン・プロジェクト名に置換します。

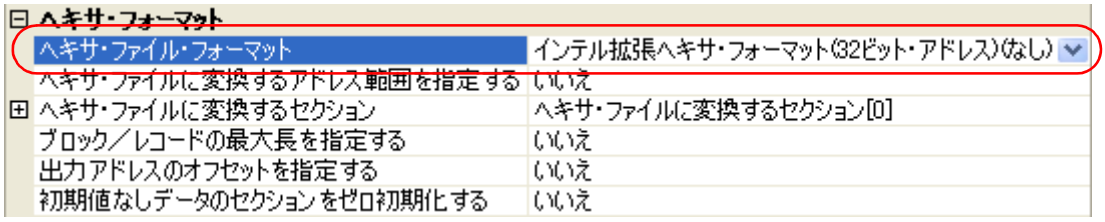
%ProjectName% ： プロジェクト名に置換します。

デフォルトでは，“%ProjectName%.hex”を設定しています。

また，ヘキサ・ファイルのフォーマットを設定することができます。

［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイル・フォーマット］プロパティで，フォーマットを選択してください。

図 2—60 ［ヘキサ・ファイル・フォーマット］プロパティ



以下のフォーマットを選択することができます。

フォーマット	構成
インテル拡張ヘキサ・フォーマット (-Xhex_format=I)	スタート・アドレス・レコード，拡張アドレス・レコード，データ・レコード，およびエンド・レコードの 4 種類のレコード
インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(なし)	スタート・リニア・アドレス・レコード，拡張リニア・スタート・アドレス・レコード，スタート・アドレス・レコード，拡張アドレス・レコード，データ・レコード，およびエンド・レコードの 6 種類のレコード
モトローラ S タイプ・フォーマット (スタンダード・アドレス)(-Xhex_format=S)	ヘッダ・レコードである S0 レコード，データ・レコードである S2 レコード，エンド・レコードである S8 レコード
モトローラ S タイプ・フォーマット (32 ビット・アドレス)(-Xhex_format=s)	ヘッダ・レコードである S0 レコード，データ・レコードである S3 レコード，エンド・レコードである S7 レコード

フォーマット	構成
拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)	データ・ブロック，シンボル・ブロック，およびターミネーション・ブロックの 3 種類のブロック

備考 ヘキサ・ファイルについての詳細は，「3. 4 ヘキサ・ファイル」を参照してください。

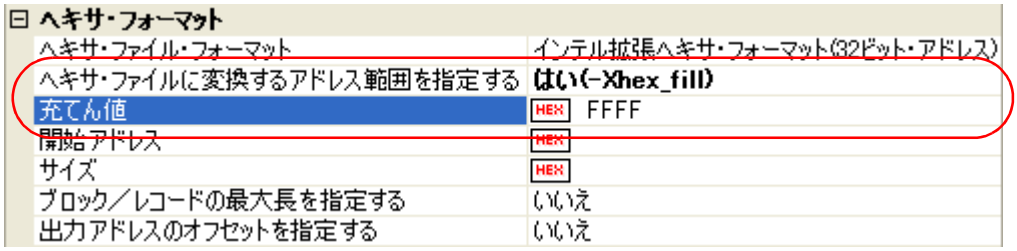
2. 9. 2 空き領域を充てんする

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ヘキサ出力オプション］タブを選択します。

空き領域の充てんについての設定は，［ヘキサ・フォーマット］カテゴリで行います。

［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［はい (-Xhex_fill)］を選択すると，［充てん値］プロパティを表示します。

図 2—61 ［ヘキサ・ファイルに変換するアドレス範囲を指定する］，および［充てん値］プロパティ



テキスト・ボックスに空き領域の充てん値を直接入力してください。

指定可能な値の範囲は 00 ～ FFFF（2 桁，または 4 桁の 16 進数）です。

デフォルトでは，“FFFF”を設定しています。

なお，ヘキサ・ファイルに変換する領域のアドレス範囲は，［開始アドレス］プロパティ，および［サイズ］プロパティで設定します。

指定可能な値の範囲は，［開始アドレス］プロパティは 0 ～ FFFFFFFF（16 進数），［サイズ］プロパティは 1 ～ 100000000（16 進数）です。

デフォルトでは，どちらも空欄となっているので，両方のプロパティを設定してください。

どちらか一方が空欄の場合は，デバイス・ファイルで定義された内蔵 ROM 領域のすべてのコードをヘキサ・ファイルに変換します。

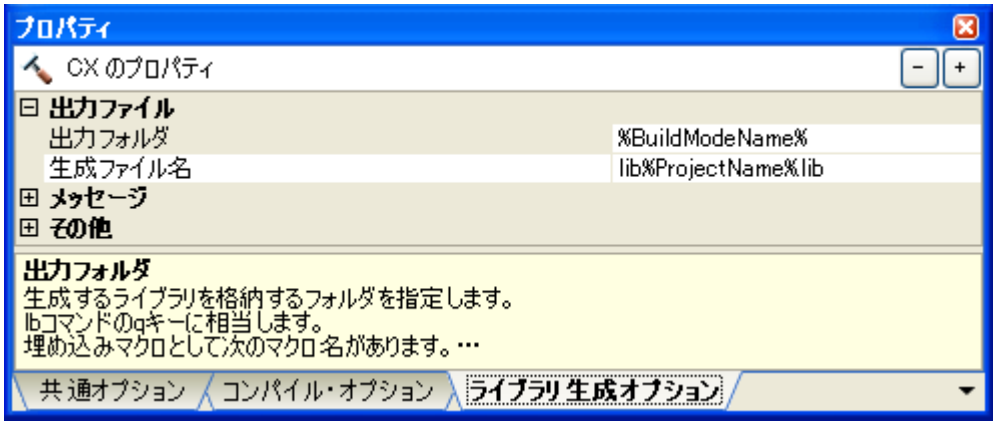
2. 10 ライブラリ生成オプションを設定する

ライブラリアンに対するオプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ライブラリ生成オプション］タブを選択してください。

タブ上で各プロパティを設定することにより，対応するライブラリ生成オプションを設定することができます。

注意 本タブは，ライブラリ用のプロジェクトの場合のみ表示します。

図 2—62 プロパティ パネル：［ライブラリ生成オプション］タブ



2. 10. 1 ライブラリ・ファイルの出力を設定する

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［ライブラリ生成オプション］タブを選択します。

ライブラリ・ファイルの出力の設定は，［出力ファイル］カテゴリで行います。

図 2—63 ［出力ファイル］カテゴリ

（1）出力フォルダの設定

［出力フォルダ］プロパティにおいて，テキスト・ボックスへの直接入力，または［...］ボタンにより行います。

テキスト・ボックスには 247 文字まで指定可能です。

本プロパティは，次のプレースホルダに対応しています。

%BuildModeName% ：ビルド・モード名に置換します。

デフォルトでは，“%BuildModeName%” を設定しています。

(2) 出力ファイル名の設定

［出力ファイル名］プロパティにおいて，テキスト・ボックスへの直接入力により行います。

テキスト・ボックスには 259 文字まで指定可能です。

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。

%MainProjectName% ：メイン・プロジェクト名に置換します。

%ProjectName% ：プロジェクト名に置換します。

デフォルトでは，“lib%ProjectName%.lib”を設定しています。

2. 11　個別にビルド・オプションを設定する

ビルド・オプションの設定は，プロジェクト単位，またはファイル単位で行います。

プロジェクト単位　→「2. 11. 1　プロジェクト単位でビルド・オプションを設定する」参照

ファイル単位　　→「2. 11. 2　ファイル単位でコンパイル／アセンブル・オプションを設定する」参照

2. 11. 1　プロジェクト単位でビルド・オプションを設定する

プロジェクト（メイン・プロジェクト，またはサブプロジェクト）に対するビルド・オプションを設定するには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルを表示します。

フェーズに対する各タブを選択し，必要なプロパティを設定することにより，ビルド・オプションを設定することができます。

コンパイル・フェーズ　→［コンパイル・オプション］タブ

アセンブル・フェーズ　→［アセンブル・オプション］タブ

リンク・フェーズ　　→［リンク・オプション］タブ

ROM 化フェーズ　　→［ROM 化オプション］タブ

ヘキサ出力フェーズ　　→［ヘキサ出力オプション］タブ

ライブラリ生成フェーズ　→［ライブラリ生成オプション］タブ

2. 11. 2　ファイル単位でコンパイル／アセンブル・オプションを設定する

プロジェクトに追加している各ソース・ファイルに対して，コンパイル・オプション，またはアセンブル・オプションを個別に設定することができます。

(1) C ソース・ファイルにコンパイル・オプションを設定する場合

プロジェクト・ツリー で C ソース・ファイルを選択し，プロパティ パネルの［ビルド設定］タブを選択します。

［ビルド］カテゴリの［個別コンパイル・オプションを設定する］プロパティで［はい］を選択すると，「図 2—65　メッセージ ダイアログ」のメッセージ ダイアログがオープンします。

図 2—64　［個別コンパイル・オプションを設定する］プロパティ

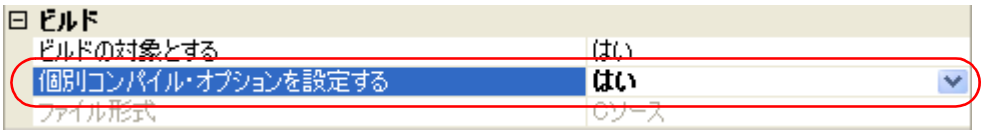
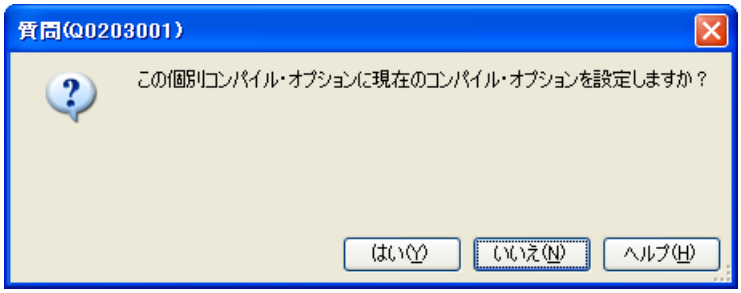
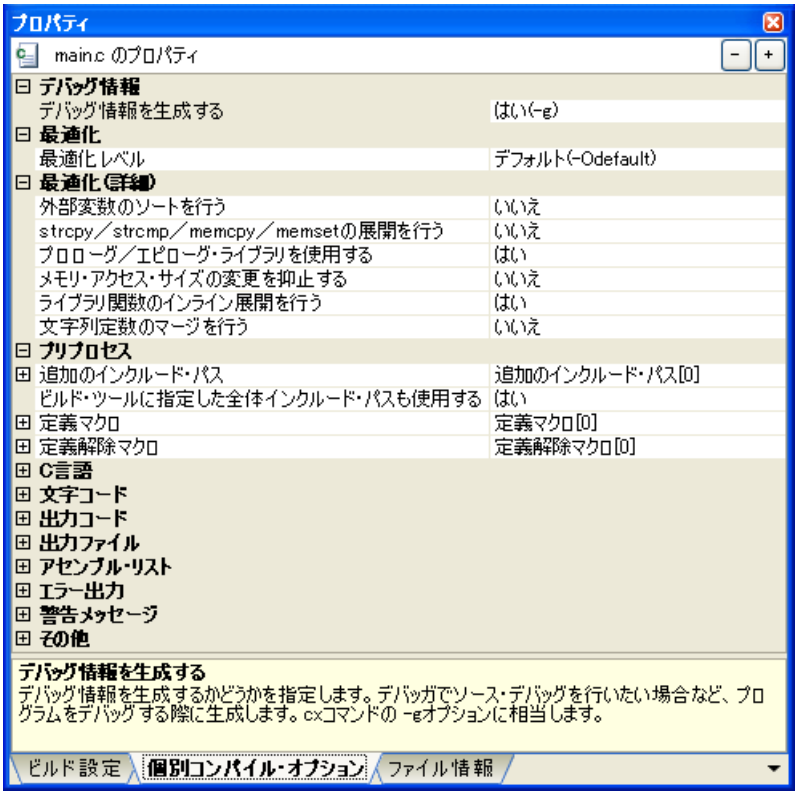


図 2—65 メッセージ ダイアログ



ダイアログ上で［はい］をクリックすると，［個別コンパイル・オプション］タブを表示します。

図 2—66 プロパティ パネル：［個別コンパイル・オプション］タブ



タブ上で必要なプロパティを設定することにより，C ソース・ファイルに対するコンパイル・オプションを設定することができます。

なお，本タブは，以下のプロパティを除いて，デフォルトでは［共通オプション］タブ，および［コンパイル・オプション］タブの設定内容を継承します。

- ［プリプロセス］カテゴリの［追加のインクルード・パス］，［ビルド・ツールに指定した全体インクルード・パスも使用する］
- ［出力ファイル］カテゴリの［オブジェクト・モジュール・ファイル名］

(2) アセンブラ・ソース・ファイルにアセンブル・オプションを設定する場合

プロジェクト・ツリー でアセンブラ・ソース・ファイルを選択し，プロパティ パネルの［ビルド設定］タブを選択します。

［ビルド］カテゴリの［個別アセンブル・オプションを設定する］プロパティで［はい］を選択すると，「図 2—68 メッセージ ダイアログ」のメッセージ ダイアログがオープンします。

図 2—67 ［個別アセンブル・オプションを設定する］プロパティ

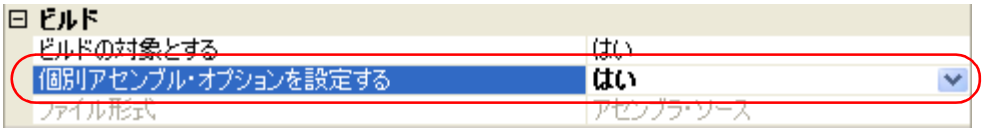
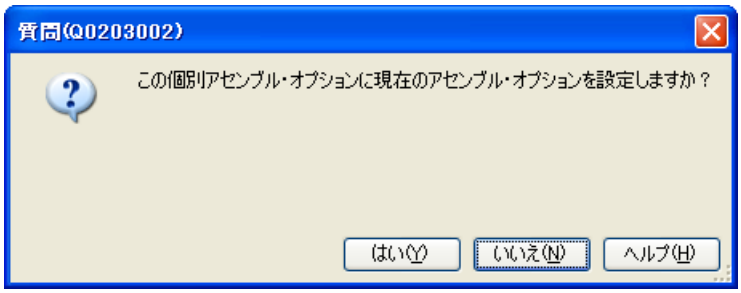
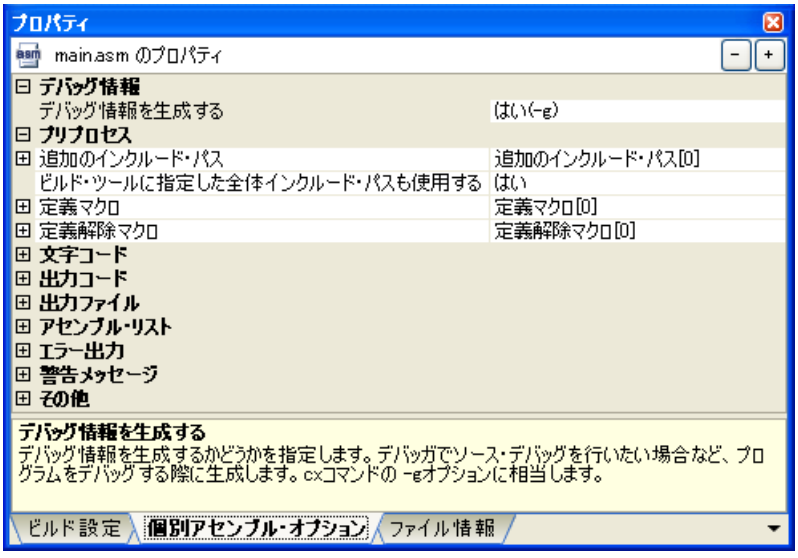


図 2—68 メッセージ ダイアログ



ダイアログ上で［はい］をクリックすると，［個別アセンブル・オプション］タブを表示します。

図 2—69 プロパティ パネル：［個別アセンブル・オプション］タブ



タブ上で必要なプロパティを設定することにより，アセンブラ・ソース・ファイルに対するアセンブル・オプションを設定することができます。

なお，本タブは，以下のプロパティを除いて，デフォルトでは［共通オプション］タブ，および［コンパイル・オプション］タブ／［アセンブル・オプション］タブの設定内容を継承します。

- ［プリプロセス］カテゴリの［追加のインクルード・パス］，［ビルド・ツールに指定した全体インクルード・パスも使用する］
- ［出力ファイル］カテゴリの［オブジェクト・モジュール・ファイル名］

2. 12 ブートーフラッシュの再リンク機能を実現するための準備をする

システムによっては，書き換え／取り換えが不可能な領域（ブート領域）に加え，フラッシュや外付け ROM といった，書き換え／取り換えが可能な領域（フラッシュ領域）を使用することがあります。

このようなシステムにおいて，フラッシュ領域上のプログラムのみを変更したい場合，ブート領域上のプログラムの再構築を行わず，ブート領域とフラッシュ領域間の関数呼び出しを正常に行う機能を“再リンク機能”と呼んでいます。

ブート領域側，フラッシュ領域側のロード・モジュール・ファイルを作成することにより，再リンク機能を実現します。

再リンク機能の実現方法を以下に示します。

備考 再リンク機能とその実現方法についての詳細は，「B. 1. 6 ブートーフラッシュ再リンク機能」を参照してください。

2. 12. 1 ビルド対象ファイルを準備する

（1）$ext_func 制御命令の指定

ブート領域側のソース・ファイルに，$ext_func 制御命令を記述します。

$ext_func 制御命令で，対象となる関数（実体がフラッシュ領域に存在し，ブート領域から呼び出す関数）について，ID 値を指定します。

備考 記述漏れやソース間の矛盾が生じることを防ぐため，$ext_func 制御命令の記述は 1 つのファイルにまとめて，ブート領域側，フラッシュ領域側を問わず，すべてのソースに $include 制御命令（C 言語で記述する場合は #include 命令）でインクルードすることを推奨します。

（2）スタートアップ・ルーチンの用意

スタートアップ・ルーチンを，ブート領域側，フラッシュ領域側のプロジェクト用にそれぞれ用意します。

それぞれのスタートアップ・ルーチンで必要な処理を，以下に示します。

- ブート領域側で tp，gp，ep 値をセットする
- ブート領域側で使用する RAM 領域を初期化するため，_rcopy 関数を呼び出す
- ブート領域側からフラッシュ領域側のスタートアップ・ルーチンへ分岐する
- フラッシュ領域側で使用する RAM 領域を初期化するため，_rcopy 関数を呼び出す
- フラッシュ領域側の処理へ移行

備考 1. tp，gp，ep をブート領域内で使用しない場合は，値のセットをフラッシュ領域で行っても問題ありません。

また，ROM 化処理を行わない場合は，_rcopy 関数の呼び出しは不要です。

2. tp，gp，ep 値は，ブート領域側，およびフラッシュ領域側で同じアドレス値を使用してください。

異なる値にすることも可能ですが，その場合は，ブート領域側とフラッシュ領域側の命令コードを行き来するたびに，各値を設定し直す必要があります。

(3) リンク・ディレクティブ・ファイルの用意

リンク・ディレクティブ・ファイルを，ブート領域側，フラッシュ領域側のプロジェクト用にそれぞれ用意します。

備考 ブート領域側，フラッシュ領域側で同じリンク・ディレクティブ・ファイルを使用してもかまいませんが，記述が煩雑になるため，それぞれの領域用にリンク・ディレクティブ・ファイルを使用することを推奨します。

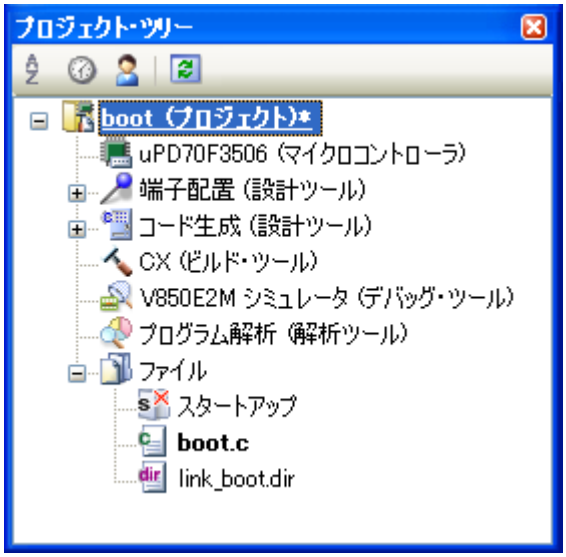
2. 12. 2 ブート領域側のプロジェクトを設定する

(1) ブート領域側のプロジェクトの作成

ブート領域側のプロジェクトを作成し，ビルド対象ファイルをプロジェクトに追加します。

スタートアップ・ルーチンは，スタートアップ・ノード直下に追加します。

図 2—70 ブート領域側のプロジェクト



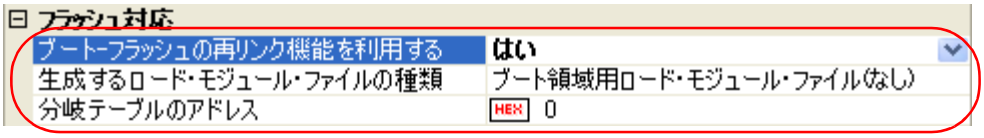
(2) ブート領域側のプロジェクトのビルド・オプションの設定

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［共通オプション］タブを選択します。

ビルド・オプションの設定は，［フラッシュ対応］カテゴリで行います。

［ブート－フラッシュの再リンク機能を利用する］プロパティで［はい］を選択すると，［生成するロード・モジュール・ファイルの種類］プロパティと［分岐テーブルのアドレス］プロパティを表示します。

図 2—71 ブート領域側の［フラッシュ対応］カテゴリ



［生成するロード・モジュール・ファイルの種類］プロパティで，［ブート領域用ロード・モジュール・ファイル (なし)］を選択します（デフォルト）。

また，［分岐テーブルのアドレス］プロパティで，分岐テーブルの先頭アドレス（フラッシュ領域内のアドレス）を指定します。

指定可能な値の範囲は，0 ～ FFFFFFFF（16 進数）です。

デフォルトでは，“0”を設定しています。

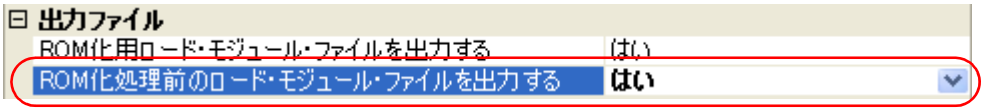
注意 フラッシュ領域側のプロジェクトでは，ROM 化処理前のロード・モジュール・ファイルの指定が必要です。

本製品はデフォルトで ROM 化処理を行うため，ROM 化処理を行う場合は，ROM 化処理前のロード・モジュールを出力しておく必要があります。

［ROM 化オプション］タブを選択して，［出力ファイル］カテゴリの［ROM 化処理前のロード・モジュール・ファイルを出力する］プロパティで［はい］を選択してください。

ファイル名は，ロード・モジュール・ファイル名に“_NonROMize”を付加した名前となります。

図 2—72 ［ROM 化処理前のロード・モジュール・ファイルを出力する］プロパティ

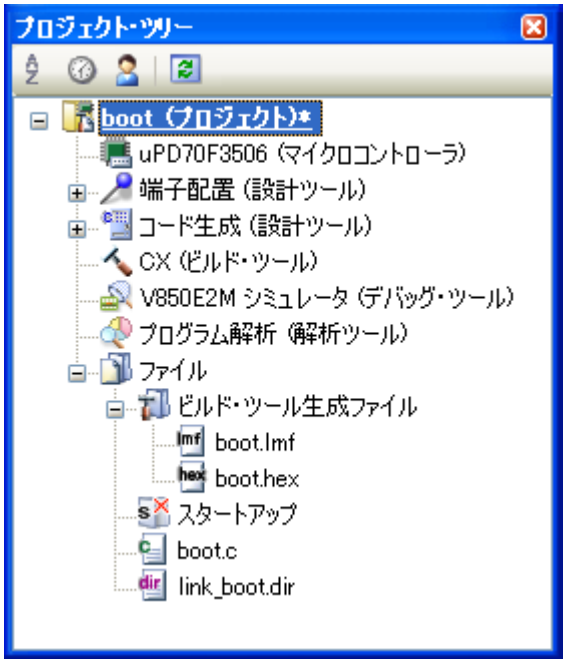


(3) ブート領域側のプロジェクトのビルドの実行

ブート領域側のプロジェクトのビルドを実行すると，ロード・モジュール・ファイルを生成します。

また，ヘキサ・ファイルを生成します。

図 2—73 ブート領域側の生成ファイル



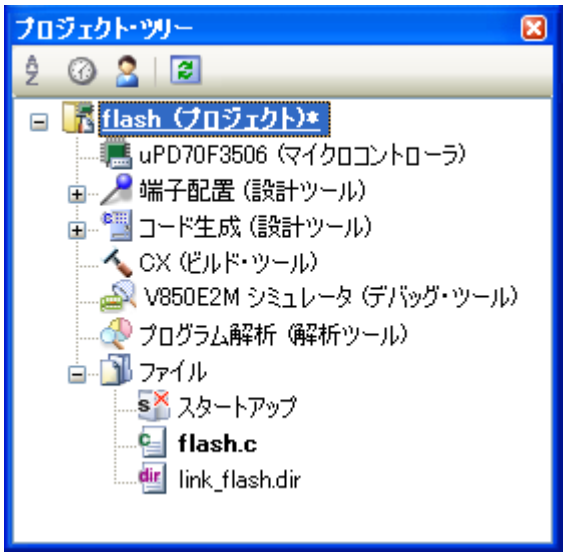
2. 12. 3　フラッシュ領域側のプロジェクトを設定する

(1) フラッシュ領域側のプロジェクトの作成

フラッシュ領域側のプロジェクトを作成し，ビルド対象ファイルをプロジェクトに追加します。

スタートアップ・ルーチンは，スタートアップ・ノード直下に追加します。

図 2―74　フラッシュ領域側のプロジェクト



(2) フラッシュ領域側のプロジェクトのビルド・オプションの設定

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［共通オプション］タブを選択します。

ビルド・オプションの設定は，［フラッシュ対応］カテゴリで行います。

［ブート－フラッシュの再リンク機能を利用する］プロパティで［はい］を選択すると，［生成するロード・モジュール・ファイルの種類］プロパティと［分岐テーブルのアドレス］プロパティを表示します。

図 2―75　フラッシュ領域側の［フラッシュ対応］カテゴリ

［生成するロード・モジュール・ファイルの種類］プロパティで，［フラッシュ領域用ロード・モジュール・ファイル (-Xflash)］を選択すると，［ブート領域用ロード・モジュール・ファイル名］プロパティを表示します。

ここで，ブート領域側のロード・モジュール・ファイルを指定します。

注意 ここで指定するものは，ROM 化処理前のロード・モジュール・ファイルです。
ROM 化処理後のロード・モジュール・ファイルを指定すると，エラーとなるので注意してください。

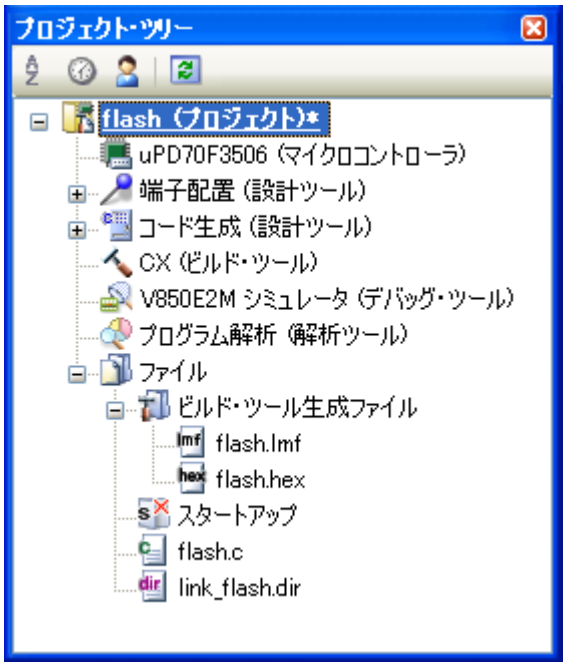
また，［分岐テーブルのアドレス］プロパティで，分岐テーブルの先頭アドレス（ブート領域側と同じアドレス）を指定します。

(3) フラッシュ領域側のプロジェクトのビルドの実行

フラッシュ領域側のプロジェクトのビルドを実行することにより，再リンク機能を実現したロード・モジュール・ファイルを生成します。

また，ブート領域用のヘキサ・ファイル（「2. 12. 2 ブート領域側のプロジェクトを設定する」で生成したファイルと同じ内容となります）とフラッシュ領域用のヘキサ・ファイルを生成します。

図 2—76 フラッシュ領域側の生成ファイル



2. 13　変数を最適なセクションに配置する

変数を最適なセクションに配置するには，シンボル情報ファイル（C ソース・ファイル内で定義した変数に対して，配置情報を記述したテキスト形式のファイル）を利用します。

シンボル情報ファイルを生成し，そのシンボル情報ファイルを参照してコンパイルを行うことにより，C ソース・ファイルを修正することなく，変数を最適なセクションに配置することができます。

以下に，操作手順を示します。

- シンボル情報ファイルを自動生成して変数の配置を行う場合
- 自動生成したシンボル情報ファイルを編集して使用する場合

なお，本機能を使用する前に，ビルドが正常に終了してロード・モジュール・ファイルが生成されていることを確認してください。

備考　シンボル情報ファイルについての詳細は，「B. 1. 4　シンボル情報ファイル」を参照してください。

(1) シンボル情報ファイルを自動生成して変数の配置を行う場合

1 回のビルドにより，シンボル情報ファイルを自動生成し，そのファイルを参照して変数の配置までを行う場合の手順を示します。

(a) シンボル情報ファイルの生成の設定

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

［シンボル情報］カテゴリの［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択すると，空のシンボル情報ファイルを生成し，プロジェクトに追加します（プロジェクト・ツリーのファイル・ノードにも表示します）。

ファイルの出力先は，［シンボル情報ファイル出力フォルダ］プロパティ，および［シンボル情報ファイル名］プロパティで設定しているものとなります。

また，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択します。

備考　すでに同名のシンボル情報ファイルが存在する場合は，ビルド対象に設定します。

図 2—77　［シンボル情報ファイルを出力する］，および［最適な配置情報を出力する］プロパティ

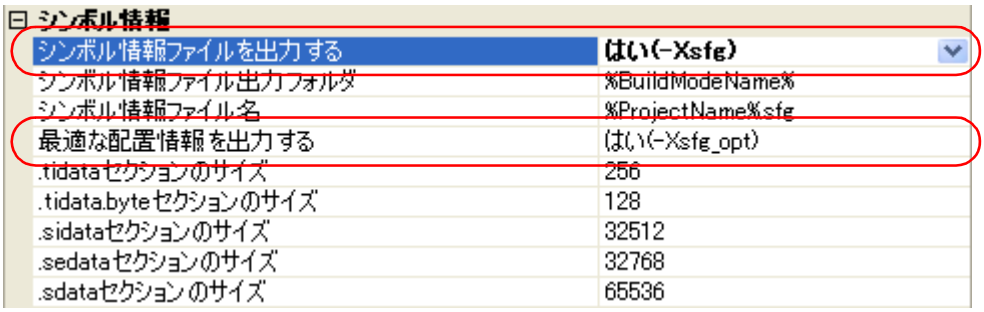
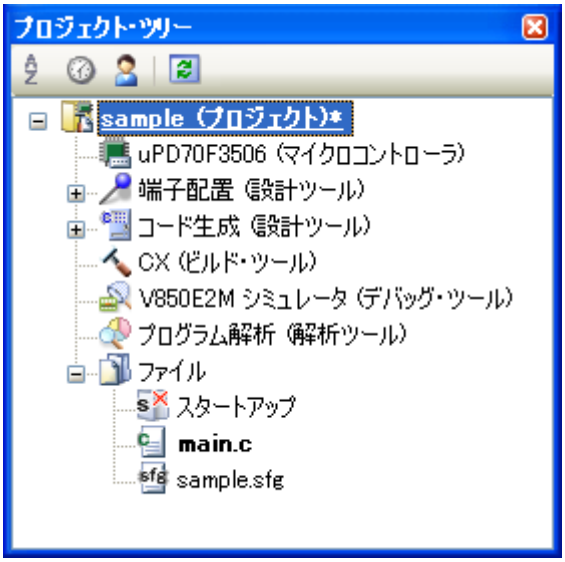


図 2―78 プロジェクト・ツリー パネル（シンボル情報ファイル生成後）



シンボル情報ファイルの出力フォルダ，およびファイル名の設定は，変更することもできます。

- 出力フォルダの設定

［シンボル情報ファイル出力フォルダ］プロパティにおいて，テキスト・ボックスへの直接入力，または［...］ボタンにより行います。

テキスト・ボックスには 247 文字まで指定可能です。

本プロパティは，次のプレースホルダに対応しています。

%BuildModeName% ：ビルド・モード名に置換します。

デフォルトでは，“%BuildModeName%”を設定しています。

- 出力ファイル名の設定

［シンボル情報ファイル名］プロパティにおいて，テキスト・ボックスへの直接入力により行います。

テキスト・ボックスには 259 文字まで指定可能です。

本プロパティは，次のプレースホルダに対応しています。

%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。

%MainProjectName% ：メイン・プロジェクト名に置換します。

%ProjectName% ：プロジェクト名に置換します。

デフォルトでは，“%ProjectName%.sfg”を設定しています。

本プロパティを変更すると，空のシンボル情報ファイルを生成し，プロジェクトに追加します（プロジェクト・ツリーのファイル・ノードにも表示します）。

（b）プロジェクトのビルドの実行

プロジェクトのビルドを実行してください。

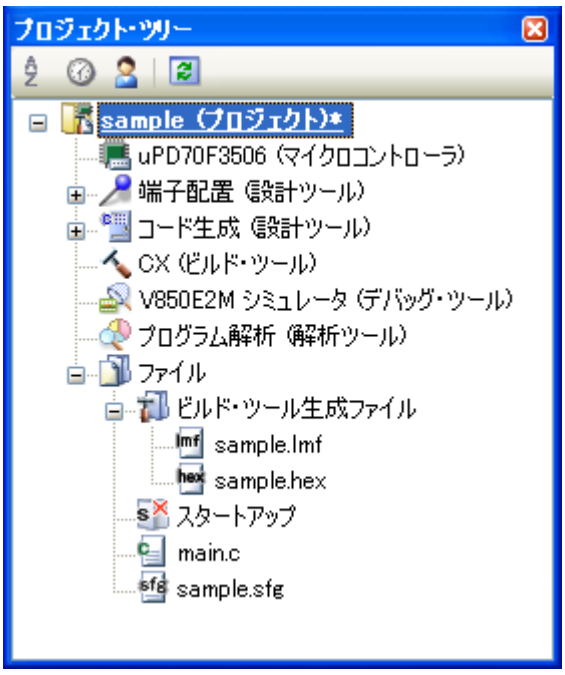
シンボル情報ファイルを生成し，自動的にそれを CX に入力して再度リビルドを実行します。

備考 1. ビルドの実行により，「(a) シンボル情報ファイルの生成の設定」で生成したシンボル情報ファイルを上書きします。

2. シンボル情報ファイルを使用してオブジェクトを再生成するため，2 回目のビルドはリビルドとなります。

ビルドが正常に終了すると，変数の配置を行ったロード・モジュール・ファイルを生成します。

図 2—79 プロジェクト・ツリー パネル（ロード・モジュール・ファイル生成後）



(2) 自動生成したシンボル情報ファイルを編集して使用する場合

シンボル情報ファイルは，ユーザが編集することも可能です。

「(1) シンボル情報ファイルを自動生成して変数の配置を行う場合」で生成したシンボル情報ファイルをユーザが編集し，そのファイルを参照して変数の配置を行う場合の手順を示します。

(a) シンボル情報ファイルの編集

「(1) シンボル情報ファイルを自動生成して変数の配置を行う場合」で自動生成したシンボル情報ファイルを編集します。

シンボル情報ファイルの記述内容は，以下のフォーマットに従ってください。

なお，変数は，利用頻度の高いものから順に記述してください。

```
[tidata.byte]:Size_セクション・サイズbytes
[tidata.word]:Size_セクション・サイズbytes
[sidata]:Size_セクション・サイズbytes
[sedata]:Size_セクション・サイズbytes
[sdata]:Size_セクション・サイズbytes
グローバル変数名，参照回数，バイト・サイズ
ファイル内static 変数名，参照回数，バイト・サイズ，パス付きファイル名
関数内static 変数名，参照回数，バイト・サイズ，パス付きファイル名，関数名
```

備考 シンボル情報ファイルのフォーマットについての詳細は，「3. 3 シンボル情報ファイル」を参照してください。

（b）シンボル情報ファイルの生成の設定

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。

［シンボル情報］カテゴリの［シンボル情報ファイルを出力する］プロパティで［いいえ］を選択します。

図 2—80 ［シンボル情報ファイルを出力する］プロパティ

（c）プロジェクトのビルドの実行

プロジェクトのビルドを実行してください。

シンボル情報ファイルで指定した内容で変数の配置を行ったロード・モジュール・ファイルを生成します。

注意 拡張子が“sfg”のファイルをプロジェクトに追加した場合，そのファイルはシンボル情報ファイルとみなします。
スタートアップ・ノード以下に追加した場合もシンボル情報ファイルとみなします。
シンボル情報ファイルをプロジェクトに追加する際，すでにシンボル情報ファイルを追加している場合は，追加する最新のシンボル情報ファイルのみがビルド対象となり，追加済みのシンボル情報ファイルはビルド対象外となります。
ビルド対象外となっているシンボル情報ファイルをビルド対象に設定する際，ほかにもシンボル情報ファイルを追加している場合は，ビルド対象に設定したシンボル情報ファイルのみがビルド対象となり，それ以外のシンボル情報ファイルはビルド対象外となります。

2. 14 マルチコア用ロード・モジュールを作成する

本製品では，複数のコア（マルチコア）で動作するプログラムから１つのロード・モジュールを作成することができます。

以下に，操作手順を示します。

備考 マルチコア・プログラミングについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

(1) リンク・ディレクティブ・ファイルの作成

リンク・ディレクティブ・ファイルは，それぞれのターゲット・システムにあわせて作成する必要があります。

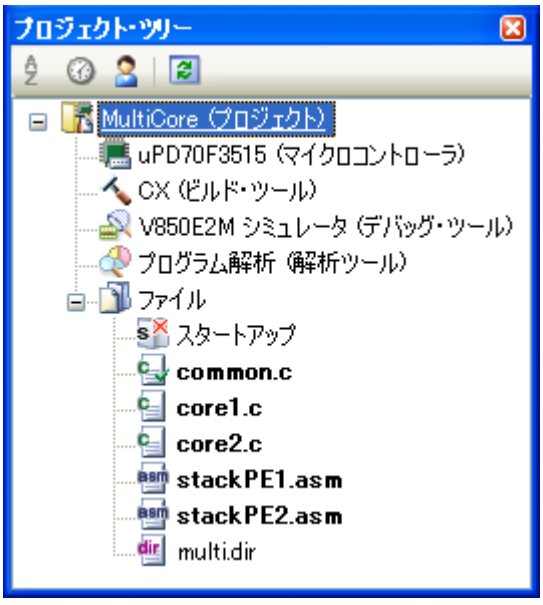
以下のサンプル・リンク・ディレクティブ・ファイルを参考に，作成してください。

本製品のインストール・フォルダ¥ CubeSuite+ ¥ CX ¥ Vx.xx ¥ smp ¥ 850 ¥ multi_smp.dir

(2) プロジェクトの作成

プロジェクトを作成し，ビルド対象ファイル（ソース・ファイル，リンク・ディレクティブ・ファイルなど）をプロジェクトに追加します。

図 2—81 マルチコア用プロジェクト



備考 標準以外のスタートアップ・ルーチンを使用する場合は，スタートアップ・ノード直下に追加します（「2. 3. 1 スタートアップ・ルーチンを設定する」参照）。

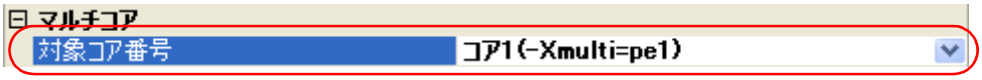
(3) ビルド・オプションの設定

(a) コア用ファイルの設定

プロジェクト・ツリーで各コア用ファイルを選択し，プロパティ パネルの［ビルド設定］タブを選択します。

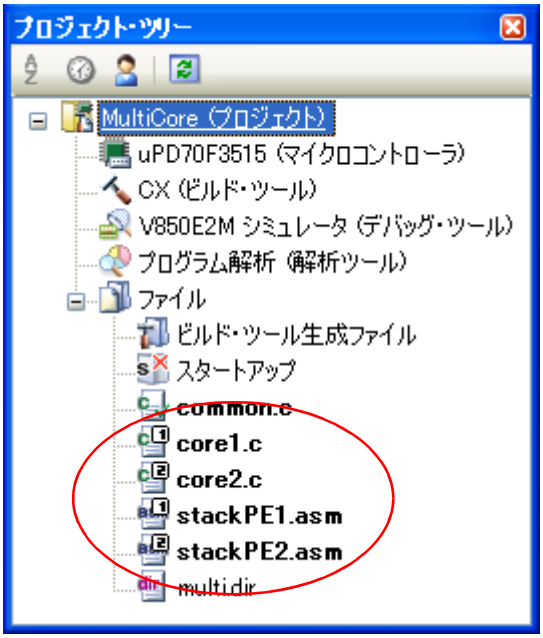
［マルチコア］カテゴリの［対象コア番号］プロパティで，該当するコア番号（［コア *n*(-Xmulti=pe*n*)］）を選択してください（*n*：ターゲット CPU が持つコアの数）。

図 2—82 ［対象コア番号］プロパティ



コア番号を指定したファイルのアイコンは，コア番号を付加したものに変更されます。

図 2—83 マルチコア用プロジェクト（ビルド・オプション設定後）

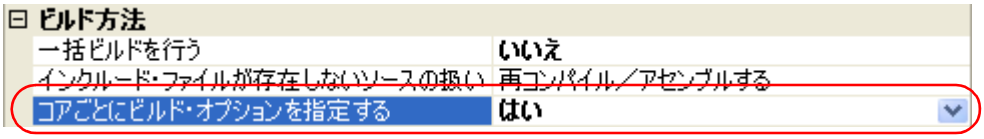


(b) コアごとのビルド・オプションの設定

プロパティ パネルの［共通オプション］タブを選択します。

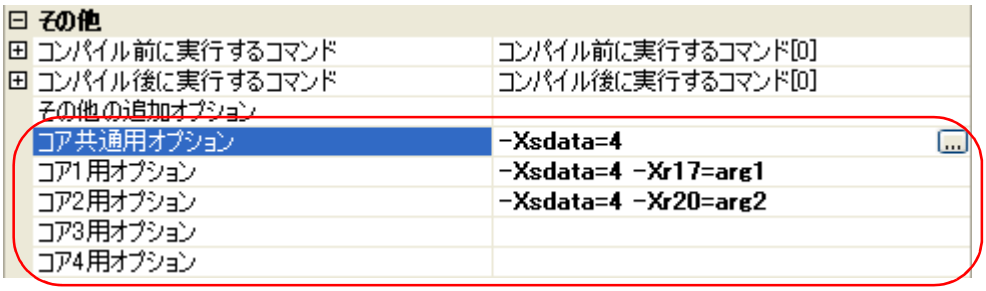
［ビルド方法］カテゴリの［コアごとにビルド・オプションを指定する］プロパティで，［はい］を選択します。

図 2—84 ［コアごとにビルド・オプションを指定する］プロパティ



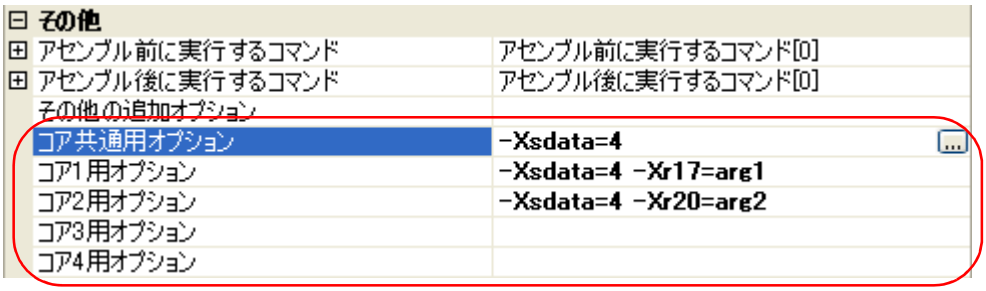
［コンパイル・オプション］タブを選択して，［その他］カテゴリの［コア *n* 用オプション］プロパティ（*n*：“共通”，またはコア番号）でコアごとのコンパイル・オプションを指定します。

図 2—85 ［コア *n* 用オプション］プロパティ（コンパイル・オプション）



また，［アセンブル・オプション］タブを選択して，［その他］カテゴリの［コア *n* 用オプション］プロパティ（*n*：“共通”，またはコア番号）でコアごとのアセンブル・オプションを指定します。

図 2—86 ［コア *n* 用オプション］プロパティ（アセンブル・オプション）

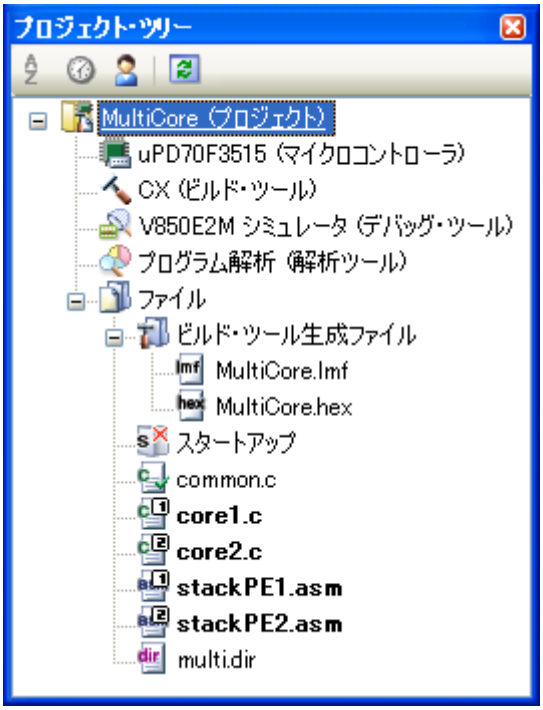


なお，コアごとに指定可能なオプションは，-Xr，および -Xsdata オプションです。

(4) プロジェクトのビルドの実行

プロジェクトのビルドを実行することにより，マルチコア用ロード・モジュール・ファイルを生成します。

図 2—87 マルチコア用プロジェクト（ビルド実行後）



2. 15 ビルドの設定をする

ここでは，ビルドに関する以下の操作を説明します。

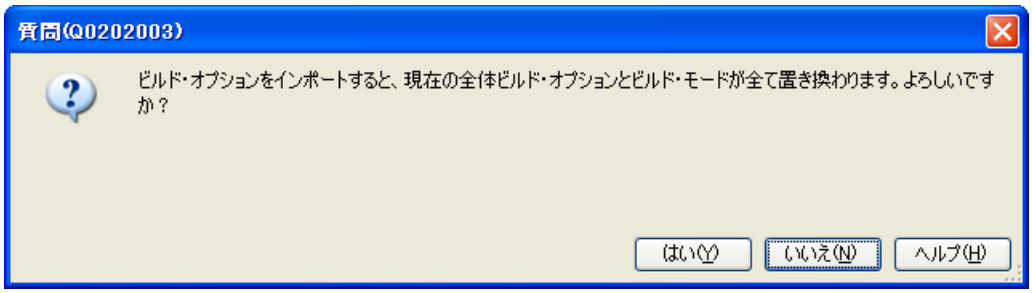
- 他のプロジェクトのビルド・オプションをインポートする
- ファイルのリンク順を設定する
- サブプロジェクトのビルド順を変更する
- ビルド・オプションを一覧表示する
- ビルド対象プロジェクトを変更する
- ビルド・モードを追加する
- ビルド・モードを変更する
- ビルド・モードを削除する
- 現在のビルド・オプションをプロジェクトの標準に設定する

2. 15. 1 他のプロジェクトのビルド・オプションをインポートする

他のプロジェクトのビルド・オプションを現在のプロジェクトにインポートすることができます。

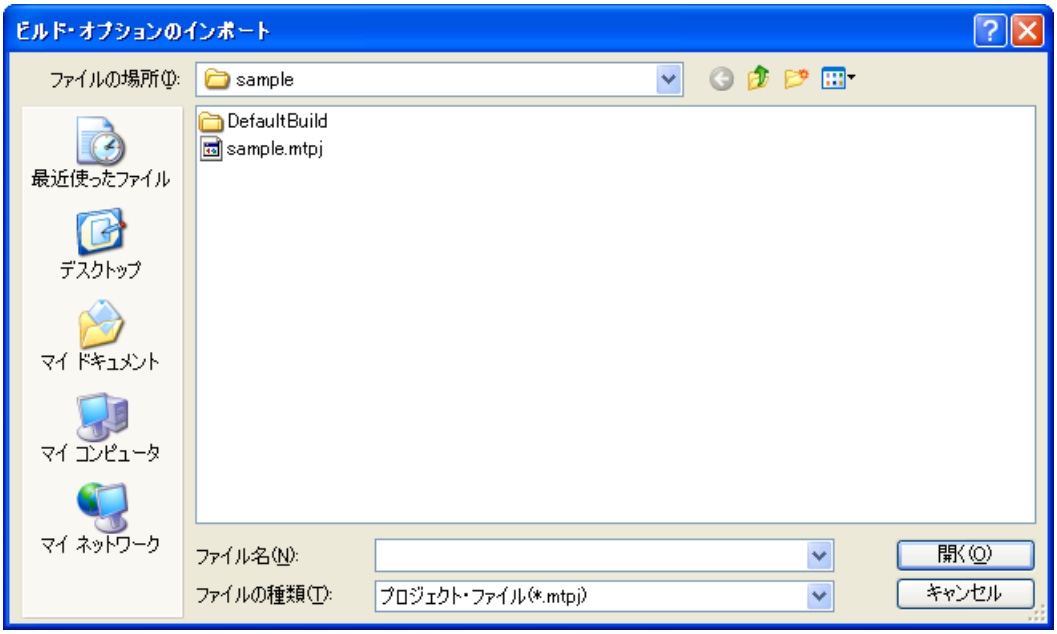
プロジェクト・ツリーでビルド・ツール・ノードを選択し，コンテキスト・メニューの［ビルド・オプションのインポート ...］を選択すると，以下のメッセージ ダイアログがオープンします。

図 2—88 メッセージ ダイアログ



ダイアログ上で［はい］をクリックすると，ビルド・オプションのインポート ダイアログがオープンします。

図 2—89 ビルド・オプションのインポート ダイアログ



ダイアログ上でビルド・オプションのインポート対象となるプロジェクト・ファイルを選択し，［開く］ボタンをクリックしてください。

選択したプロジェクト・ファイルのビルド・オプションを現在のプロジェクトにインポートします。

備考 1. インポート可能なプロジェクトの条件を以下に示します。

- ビルド・ツールが同じである。
- プロジェクトの種類（アプリケーション，ライブラリ等）が同じである。
- 同じメジャー・バージョンの CubeSuite+ で作成したプロジェクトである。

2. インポート対象となるビルド・オプションは，ビルド・ツールのプロパティで設定した全体オプションのみです。

標準ビルド・オプションの設定（「2. 15. 9 現在のビルド・オプションをプロジェクトの標準に設定する」参照）や個別オプションのインポートは行いません。

3. インポート対象のすべてのビルド・モードのインポートも行います。

なお，現在のプロジェクトのビルド・モードは“DefaultBuild”以外は削除します。

4. 使用するビルド・ツールのバージョンのインポートも行います。

2. 15. 2 ファイルのリンク順を設定する

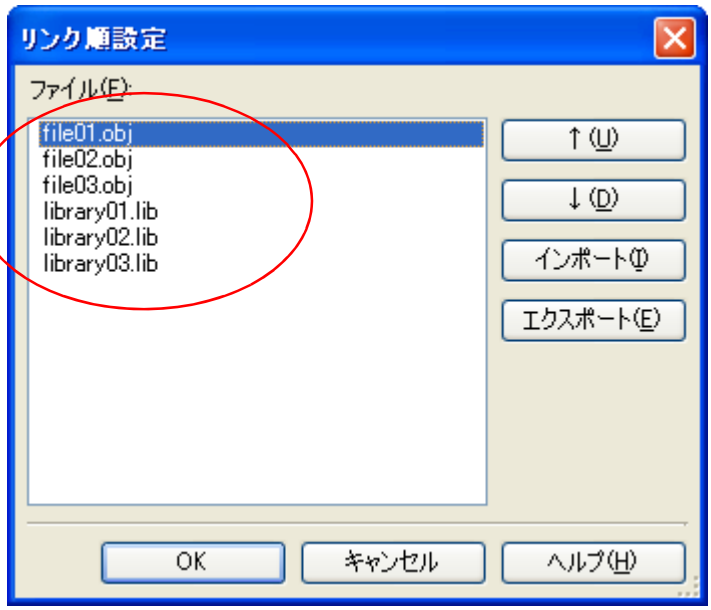
オブジェクト・モジュール・ファイル，およびライブラリ・ファイルのリンク順は，自動で決定しますが，ユーザが設定することもできます。

以下に，操作手順を示します。

(1) リンク順設定 ダイアログのオープン

プロジェクト・ツリーでビルド・ツール・ノードを選択し，コンテキスト・メニューの［リンク順を設定する ...］を選択すると，リンク順設定 ダイアログがオープンします。

図 2―90 リンク順設定 ダイアログ



［ファイル］には，以下のファイルのファイル名一覧を，リンカへの入力順で表示します。

- 選択しているメイン・プロジェクト，またはサブプロジェクトに追加しているソース・ファイルから生成するオブジェクト・モジュール・ファイル
- 選択しているメイン・プロジェクト，またはサブプロジェクトのプロジェクト・ツリーに直接追加したオブジェクト・モジュール・ファイル
- 選択しているメイン・プロジェクト，またはサブプロジェクトのプロジェクト・ツリーに直接追加したライブラリ・ファイル

備考 デフォルトでは，プロジェクトに追加している順番となります。
新規に追加したソース・ファイルから生成するオブジェクト・モジュール・ファイル，および新規に追加したオブジェクト・モジュール・ファイルは，一覧の最後のオブジェクト・モジュール・ファイルの次に追加します。
新規に追加したライブラリ・ファイルは，一覧の最後に追加します。

（2）ファイルの表示順の変更

ファイルの表示順を変更することにより，リンカへのファイルの入力順を設定することができます。

以下のいずれかの方法により，ファイルの表示順を変更します。

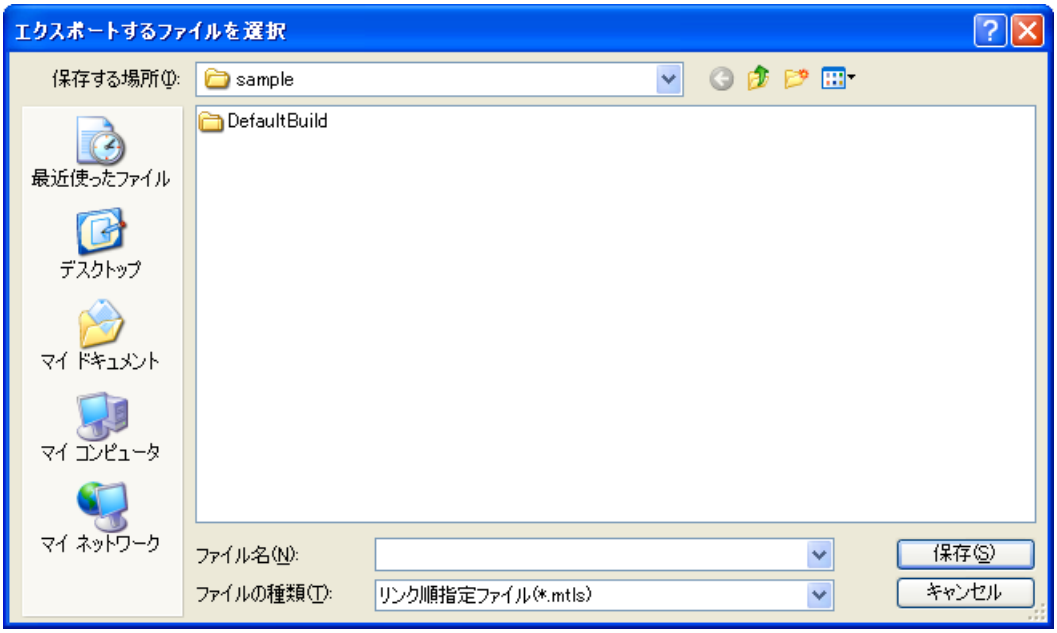
- ［↑］，および［↓］ボタンによるファイル名の移動
- ファイル名のドラッグ・アンド・ドロップ
- リンク順指定ファイルの利用

備考　リンク順指定ファイルを利用することにより，ファイル・ベースで表示順を変更することができます。以下に，操作手順を示します。

（a）リンク順指定ファイルの生成

リンク順設定 ダイアログの［エクスポート］ボタンをクリックすると，エクスポートするファイルを選択 ダイアログがオープンします。

図 2—91　エクスポートするファイルを選択 ダイアログ



ダイアログ上で，リンク順設定 ダイアログの［ファイル］に表示しているファイル名一覧を出力するファイル（リンク順指定ファイル）を指定します。

［保存］ボタンをクリックすると，リンク順指定ファイルが生成されます。

注意　リンク順指定ファイルには，ファイル名のみを出力します。
同名のファイルが存在する場合は，リンク順指定ファイルのインポート後にポップアップ表示でファイルの存在場所を確認してください。

(b) リンク順指定ファイルの編集

エディタでリンク順指定ファイルをオープンし，ファイル名の記述順を変更します。

リンク順指定ファイルの記述例を以下に示します。

```
# CubeSuite+ Vx.xx.xx Link order specification file
# SampleProject: xxxx 年 xx 月 xx 日

file01.obj
file03.obj
library02.lib
file02.obj
library01.lib
library03.lib
    :
```

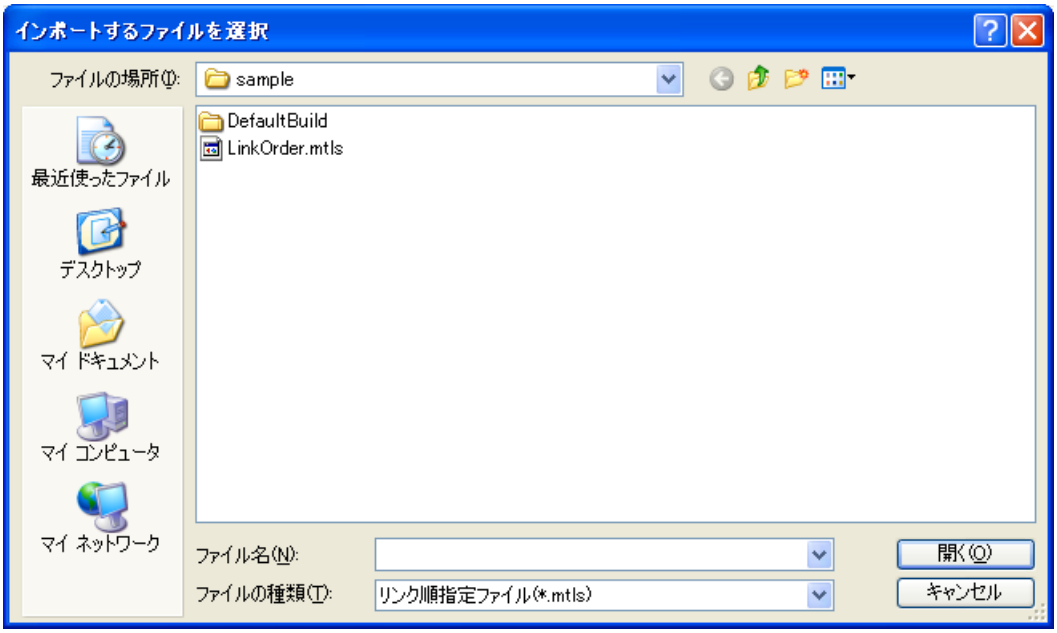
リンク順指定ファイルを編集する際の注意事項を以下に示します。

- ファイル名は 1 行に 1 つずつ指定してください。
- ファイル名の大文字／小文字は区別しません。
- “#”で始まる行は，コメント行とみなします。
- 空白文字（半角スペース，タブ）は無視します。

(c) リンク順指定ファイルのインポート

リンク順設定 ダイアログの［インポート］ボタンをクリックすると，インポートするファイルを選択 ダイアログがオープンします。

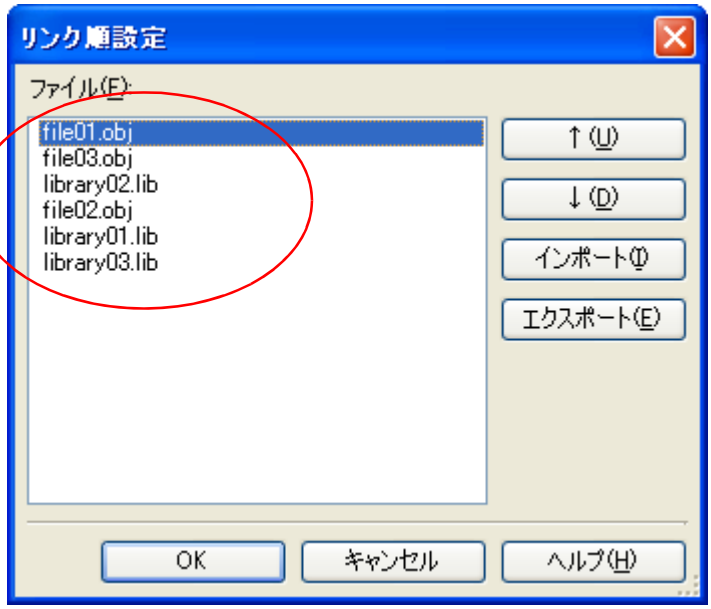
図 2—92 インポートするファイルを選択 ダイアログ



ダイアログ上でリンク順指定ファイルを選択し，［開く］ボタンをクリックしてください。

選択したリンク順指定ファイルからファイル名の記述順を取得し，リンク順設定 ダイアログの［ファイル］に反映します。

図 2—93 リンク順設定 ダイアログ（リンク順設定後）



注意 1. リンク順指定ファイルに記述しているが，プロジェクトには追加していないファイルは，表示されません。
該当ファイルが存在する場合は，出力 パネルにファイル名一覧が表示されます。

2. プロジェクトに追加しているが，リンク順指定ファイルには記述していないファイルは，［ファイル］の最後に表示されます。

3. 同名のファイルが存在する場合は，ポップアップ表示（ファイル名にマウス・カーソルをあわせると表示されます）でファイルの存在場所を確認してください。
リンク順の変更が必要な場合は，［↑］，および［↓］ボタン，またはファイル名のドラッグ・アンド・ドロップにより行ってください。

(3) ファイルのリンク順の設定

リンク順設定 ダイアログで［OK］ボタンをクリックすることにより，リンカへのファイルの入力順を設定することができます。

2. 15. 3 サブプロジェクトのビルド順を変更する

ビルドの実行は，サブプロジェクト，メイン・プロジェクトの順で行いますが，複数のサブプロジェクトを追加している場合，サブプロジェクトのビルド順はプロジェクト・ツリーでの表示順となります。

プロジェクト・ツリーでのサブプロジェクトの表示順を変更するには，移動するサブプロジェクトをドラッグし，移動先でドロップしてください。

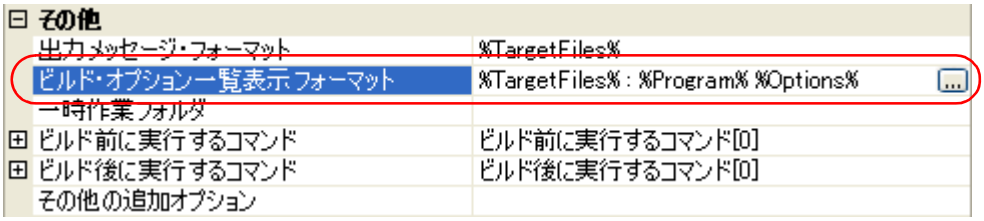
2. 15. 4 ビルド・オプションを一覧表示する

プロジェクト（メイン・プロジェクト，およびサブプロジェクト）に対して，プロパティ パネルで現在設定しているビルド・オプションを一覧表示することができます。

［ビルド］メニュー→［ビルド・オプション一覧］を選択すると，プロジェクトに対する現在のオプションの設定内容を，出力 パネルの［ビルド・ツール］タブにビルド順に表示します。

備考 ビルド・オプション一覧の表示フォーマットは，変更することができます。
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［共通オプション］タブを選択します。
［その他］カテゴリの［ビルド・オプション一覧表示フォーマット］プロパティを設定してください。

図 2—94 ［ビルド・オプション一覧表示フォーマット］プロパティ



次のプレースホルダに対応しています。

%Program% ： 実行中のプログラム名に置換します。
%Options% ： ビルド実行時のコマンド・ライン・オプションに置換します。
%TargetFiles% ： コンパイル／アセンブル中のファイル名，またはリンク後の出力ファイル名に置換します。

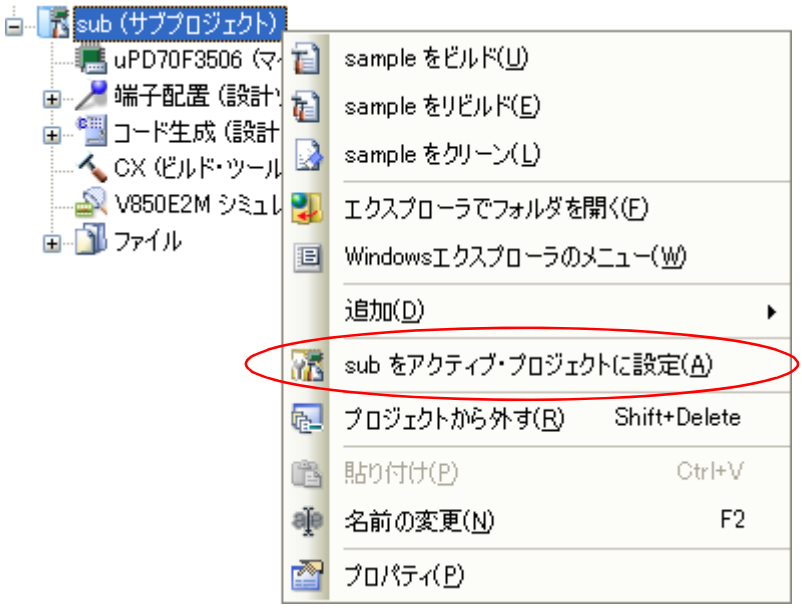
デフォルトでは，“%TargetFiles% : %Program% %Options%”を設定しています。

2. 15. 5 ビルド対象プロジェクトを変更する

特定のプロジェクト（メイン・プロジェクト，またはサブプロジェクト）を対象にビルドを行う場合，そのプロジェクトを“アクティブ・プロジェクト”として設定する必要があります。

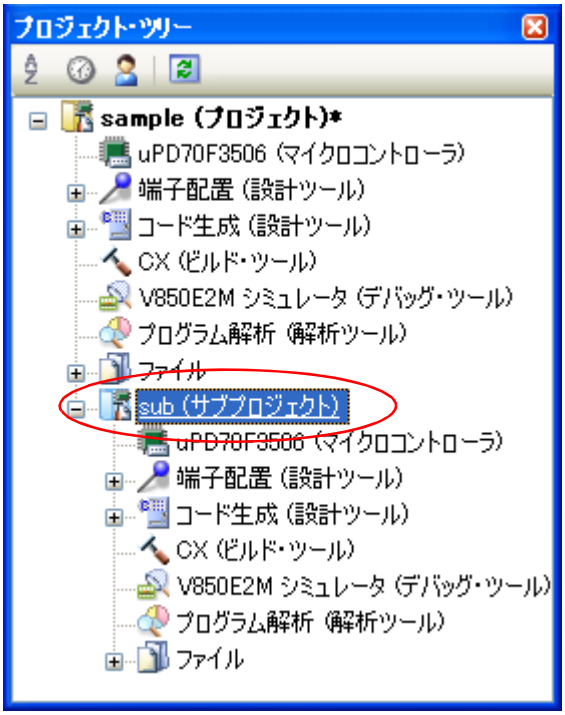
アクティブ・プロジェクトを設定するには，プロジェクト・ツリーでアクティブ・プロジェクトに設定するメイン・プロジェクト・ノード，またはサブプロジェクト・ノード選択し，コンテキスト・メニューの［*選択しているプロジェクト*をアクティブ・プロジェクトに設定］を選択してください。

図 2—95 ［*選択しているプロジェクトをアクティブ・プロジェクトに設定*］項目



アクティブ・プロジェクトを設定すると，そのプロジェクトには下線を付加します。

図 2—96 アクティブ・プロジェクト



備考 1. プロジェクトの作成直後は，メイン・プロジェクトがアクティブ・プロジェクトとなります。

2. アクティブ・プロジェクトに設定しているサブプロジェクトをプロジェクトから外した場合は，メイン・プロジェクトがアクティブ・プロジェクトとなります。

2. 15. 6 ビルド・モードを追加する

ビルドの目的に応じてビルド・オプションや定義マクロを変更したい場合，それらの設定を一括して変更することができます。

ビルド・オプションや定義マクロの設定をまとめたものをビルド・モードと呼び，ビルド・モードを変更することにより，ビルド・オプションや定義マクロの設定を毎回変更する必要がなくなります。

ビルド・モードは，デフォルトでは“DefaultBuild”のみ用意していますので，ビルドの目的に応じてユーザが追加してください。

以下に，ビルド・モードの追加方法を示します。

(1) ビルド・モードの新規作成

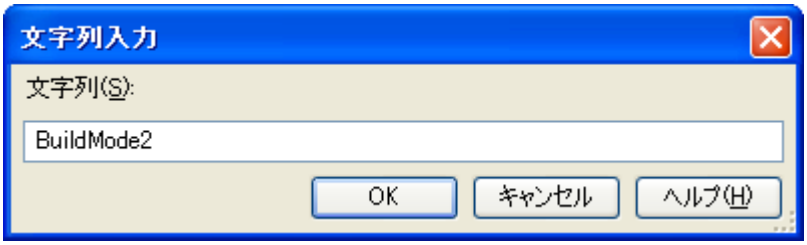
新規のビルド・モードの作成は，既存のビルド・モードの複製により行います。

［ビルド］メニュー→［ビルド・モードの設定 ...］を選択すると，ビルド・モード設定 ダイアログがオープンします。

図 2—97 ビルド・モード設定 ダイアログ

ビルド・モードの一覧から複製元のビルド・モードを選択したのち，［複製 ...］ボタンをクリックすると，文字列入力 ダイアログがオープンします。

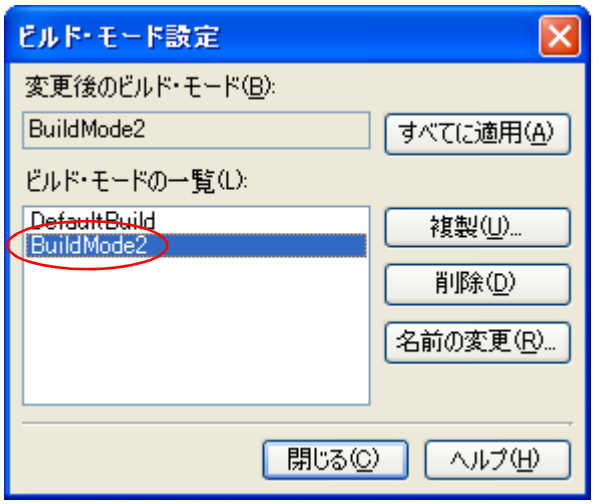
図 2—98 文字列入力 ダイアログ



ダイアログ上で新規作成するビルド・モードの名前を入力し，［OK］ボタンをクリックすると，その名前でビルド・モードを複製します。

プロジェクトに属するメイン・プロジェクト，およびすべてのサブプロジェクトのビルド・モードに，作成したビルド・モードを追加します。

図 2—99 ビルド・モード設定 ダイアログ（ビルド・モード追加後）



(2) ビルド・モードの変更

ビルド・モードを新規に作成したビルド・モードに変更します（「2. 15. 7 ビルド・モードを変更する」参照）。

(3) ビルド・モードの設定内容の変更

プロジェクト・ツリー でビルド・ツール・ノードを選択したのち，プロパティ パネルでビルド・オプションや定義マクロの設定を変更します。

備考 ビルド・モードの作成は，プロジェクトの変更とみなします。
プロジェクトを閉じる際に，ビルド・モードを保存するかどうかの確認を行います。

2. 15. 7 ビルド・モードを変更する

ビルドの目的に応じてビルド・オプションや定義マクロを変更したい場合，それらの設定を一括して変更することができます。

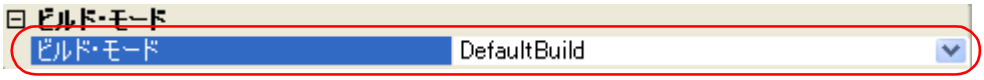
ビルド・オプションや定義マクロの設定をまとめたものをビルド・モードと呼び，ビルド・モードを変更することにより，ビルド・オプションや定義マクロの設定を毎回変更する必要がなくなります。

(1) メイン・プロジェクト，またはサブプロジェクトのビルド・モードを変更する場合

プロジェクト・ツリー で対象プロジェクトのビルド・ツール・ノードを選択したのち，プロパティ パネルの［共通オプション］タブを選択します。

［ビルド・モード］カテゴリの［ビルド・モード］プロパティで変更するビルド・モードを選択してください。

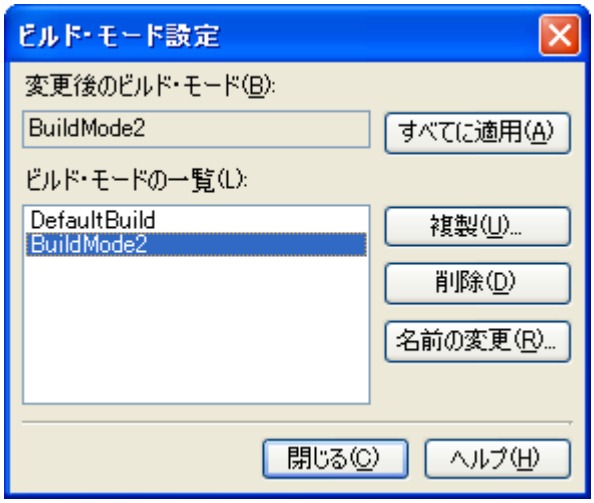
図 2—100 ［ビルド・モード］プロパティ



(2) プロジェクト全体のビルド・モードを変更する場合

［ビルド］メニュー→［ビルド・モードの設定 ...］を選択すると，ビルド・モード設定 ダイアログがオープンします。

図 2—101 ビルド・モード設定 ダイアログ



ビルド・モードの一覧から変更するビルド・モードを選択すると，［変更後のビルド・モード］に選択したビルド・モードを表示します。

［すべてに適用］ボタンをクリックすると，プロジェクトに属するメイン・プロジェクト，およびすべてのサブプロジェクトのビルド・モードを，ダイアログ上で選択したビルド・モードに変更します。

注意 選択したビルド・モードが存在しないプロジェクトについては，“DefaultBuild”を選択したビルド・モード名で複製し，複製したビルド・モードに変更します。

備考 1. ビルド・モードは，デフォルトでは“DefaultBuild”のみ用意しています。
ビルド・モードの追加方法については，「2. 15. 6 ビルド・モードを追加する」を参照してください。

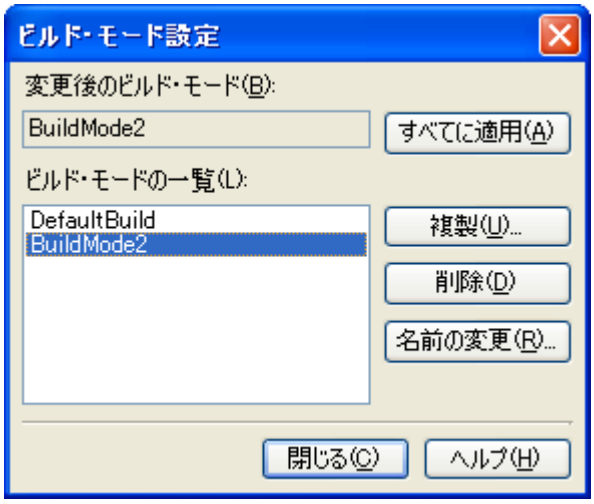
2. ビルド・モードの一覧でビルド・モードを選択したのち，［名前の変更］ボタンをクリックすることにより，ビルド・モードの名前を変更することができます。
ただし，“DefaultBuild”は名前を変更することができません。

2. 15. 8 ビルド・モードを削除する

ビルド・モードの削除は，ビルド・モード設定 ダイアログで行います。

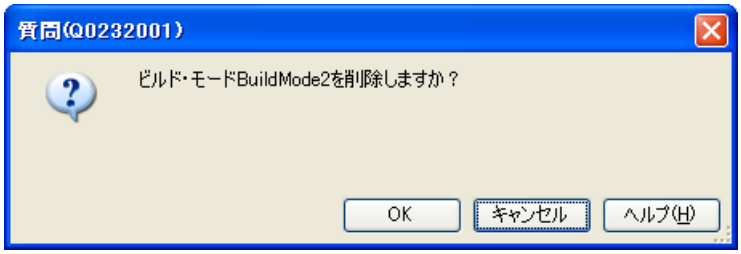
［ビルド］メニュー→［ビルド・モードの設定 ...］を選択すると，ダイアログがオープンします。

図 2—102 ビルド・モード設定 ダイアログ



ビルド・モードの一覧で削除するビルド・モードを選択したのち，［削除］ボタンをクリックすると，以下のメッセージ ダイアログがオープンします。

図 2—103 メッセージ ダイアログ



処理を継続するには，ダイアログ上で［OK］をクリックしてください。

選択したビルド・モードをプロジェクトから削除します。

注意 “DefaultBuild”を削除することはできません。

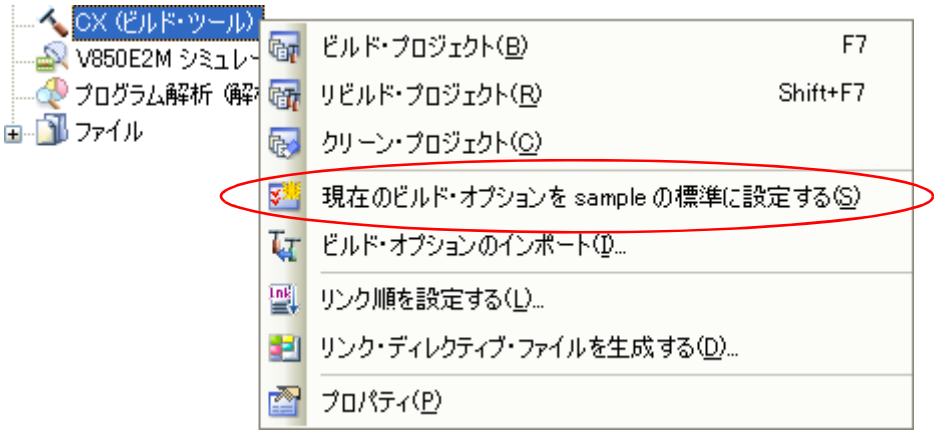
2. 15. 9　現在のビルド・オプションをプロジェクトの標準に設定する

プロパティ パネルにおいて，標準ビルド・オプションの設定に変更を加えると，プロパティの値を太字表示します。

図 2—104　プロパティ パネル（標準ビルド・オプション変更後）

現在選択しているプロジェクト（メイン・プロジェクト，またはサブプロジェクト）のビルド・オプションを標準ビルド・オプションとする（太字表示を解除する）には，プロジェクト・ツリーでビルド・ツール・ノードを選択し，コンテキスト・メニューの［現在のビルド・オプションを*選択しているプロジェクト*の標準に設定する］を選択してください。

図 2—105　［現在のビルド・オプションを*選択しているプロジェクト*の標準に設定する］項目



標準ビルド・オプションに設定後のプロパティの値は，以下のようになります。

図 2—106　プロパティ パネル（標準ビルド・オプション設定後）

注意　メイン・プロジェクトを選択している場合，メイン・プロジェクトの設定のみ行います。サブプロジェクトを追加していても，サブプロジェクトの設定は行いません。

2. 16 ビルドを実行する

ここでは，ビルドの実行に関する操作を説明します。

(1) ビルドの種類

ビルドには，次の種類があります。

表 2—1 ビルドの種類

種類	説明
ビルド	ビルド対象ファイルのうち，更新したファイルのみビルドを実行します。 →「2. 16. 1 更新ファイルのビルドを実行する」参照
リビルド	ビルド対象のすべてのファイルのビルドを実行します。 →「2. 16. 2 すべてのファイルのビルドを実行する」参照
ラピッド・ビルド	ビルド設定の変更と平行してビルドを実行します。 →「2. 16. 3 他の処理と平行してビルドを実行する」参照
バッチ・ビルド	プロジェクトが持つビルド・モードを一括してビルドを実行します。 →「2. 16. 4 ビルド・モードを一括してビルドを実行する」参照

備考 1. ビルドの実行は，サブプロジェクト，メイン・プロジェクトの順で行います。
サブプロジェクトは，プロジェクト・ツリーでの表示順にビルドを行います（「2. 15. 3 サブプロジェクトのビルド順を変更する」参照）。

2. ビルド，リビルド，バッチ・ビルドを実行する際，エディタ パネルで編集中のファイルがある場合は，該当ファイルを一括して保存します。

(2) 実行結果の表示

ビルドの実行結果（ビルド・ツールの出力メッセージ）は，出力 パネルの各タブに表示します。

- ビルド，リビルド，バッチ・ビルドの場合
  → [すべてのメッセージ] タブ，および [ビルド・ツール] タブ
- ラピッド・ビルドの場合
  → [ラピッド・ビルド] タブ

図 2—107 ビルドの実行結果（ビルド，リビルド，バッチ・ビルドの場合）

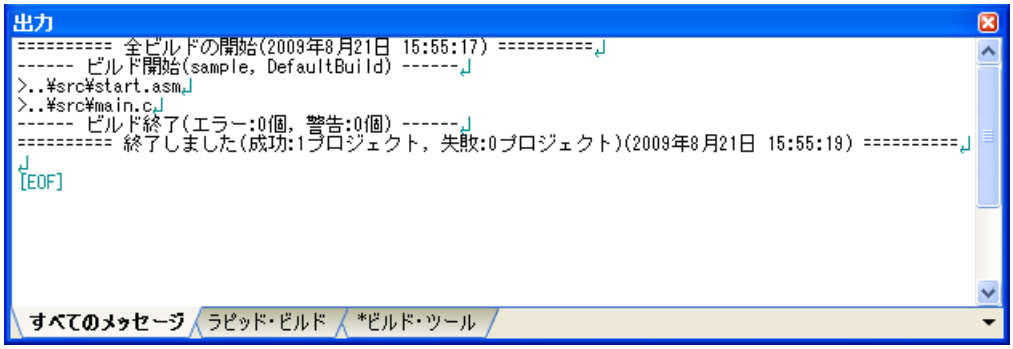
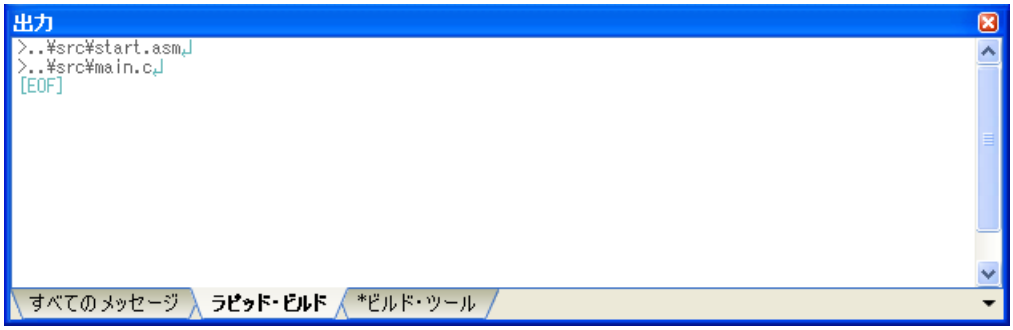


図 2—108 ビルドの実行結果（ラピッド・ビルドの場合）



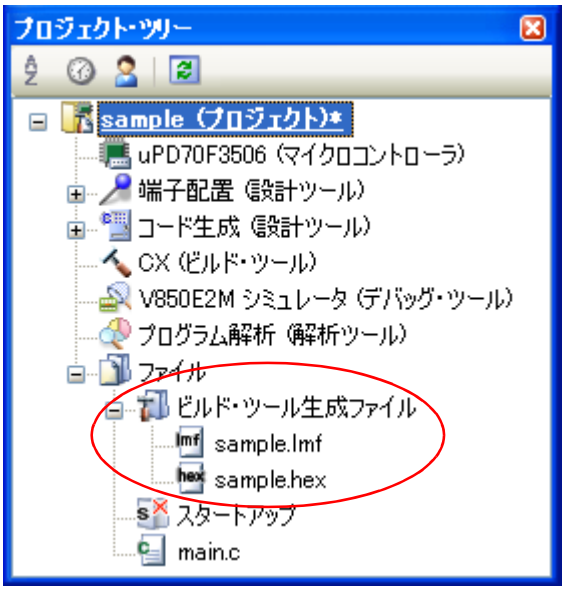
備考 1. ［ラピッド・ビルド］タブの表示文字列は，淡色表示になります。

2. 出力したメッセージからファイル名／行番号を獲得できる場合，メッセージ上でダブルクリックすると，ファイルの該当する行へジャンプすることができます。

3. 警告メッセージ，またはエラー・メッセージを表示している行にキャレットがある状態で，［F1］キーを押下すると，その行のメッセージに関するヘルプを表示することができます。

ビルド・ツールの生成ファイルは，プロジェクト・ツリー パネルのビルド・ツール生成ファイル・ノードに表示します。

図 2—109 ビルド・ツールの生成ファイル



備考 ビルド・ツール生成ファイル・ノードに表示するのは，以下のファイルです。

- ライブラリ用のプロジェクト以外の場合

ロード・モジュール・ファイル（*.lmf）

リンク・マップ・ファイル（*.map）

ヘキサ・ファイル（*.hex）

エラー・メッセージ・ファイル（*.err）

- ライブラリ用のプロジェクトの場合
  ライブラリ・ファイル (*.lib)
  エラー・メッセージ・ファイル（*.err）

注意 ビルド・ツール生成ファイル・ノードは，ビルド時に作成するノードです。
ビルド後にプロジェクトの再読み込みを行った場合，本ノードは表示されなくなります。

2. 16. 1 更新ファイルのビルドを実行する

ビルド対象ファイルのうち，更新したファイルのみビルドを実行します（以降，“ビルド”と呼びます）。

ビルドの実行は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト），またはアクティブ・プロジェクト（「2. 15. 5 ビルド対象プロジェクトを変更する」参照）に対して行います。

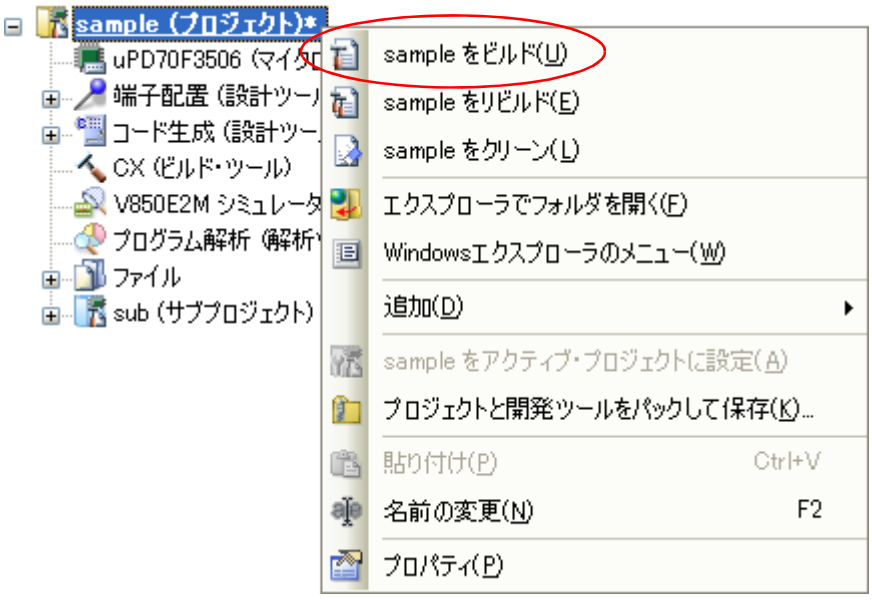
(1) プロジェクト全体のビルドを実行する場合

ツールバーの  ボタンをクリックしてください。

(2) アクティブ・プロジェクトのビルドを実行する場合

プロジェクトを選択し，コンテキスト・メニューの［*アクティブ・プロジェクト*をビルド］を選択してください。

図 2—110 ［*アクティブ・プロジェクト*をビルド］項目



備考 ヘッダ・ファイルを編集後にビルドを実行してもインクルードしているソース・ファイルがビルドされない場合は，ファイルの依存関係を更新してください（「2. 3. 8 ファイルの依存関係を更新する」参照）。

2. 16. 2 すべてのファイルのビルドを実行する

ビルド対象のすべてのファイルのビルドを実行します（以降，“リビルド”と呼びます）。

また，クロス・リファレンス・ファイルの削除も行います。

リビルドの実行は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト），またはアクティブ・プロジェクト（「2. 15. 5 ビルド対象プロジェクトを変更する」参照）に対して行います。

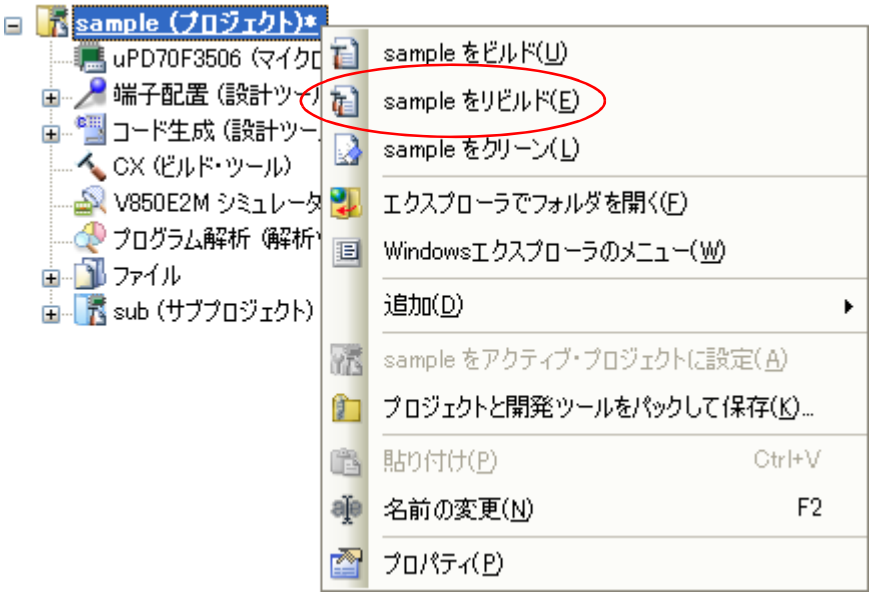
(1) プロジェクト全体のリビルドを実行する場合

ツールバーの  ボタンをクリックしてください。

(2) アクティブ・プロジェクトのリビルドを実行する場合

プロジェクトを選択し，コンテキスト・メニューの［*アクティブ・プロジェクト*をリビルド］を選択してください。

図 2―111 ［*アクティブ・プロジェクト*をリビルド］項目



2. 16. 3 他の処理と平行してビルドを実行する

以下のタイミングでビルドを自動で開始する機能があります（以降，“ラピッド・ビルド”と呼びます）。

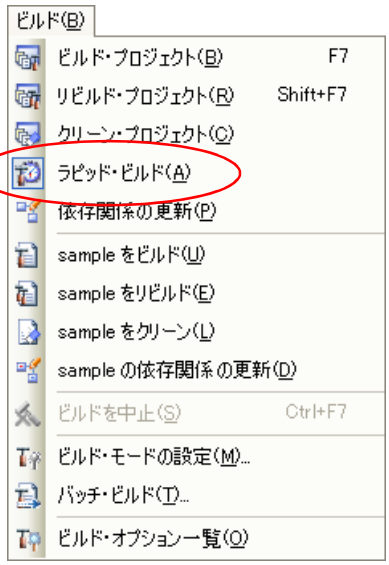
- プロジェクトに追加しているＣソース・ファイル，アセンブラ・ソース・ファイル，ヘッダ・ファイル，リンク・ディレクティブ・ファイル，シンボル情報ファイル，オブジェクト・モジュール・ファイル，およびライブラリ・ファイルを更新したとき
- プロジェクトにビルド対象ファイルを追加，または削除したとき
- オブジェクト・モジュール・ファイル，およびライブラリ・ファイルのリンク順を変更したとき
- ビルド・ツール，およびビルド対象ファイルのプロパティを変更したとき

ラピッド・ビルドを有効にすることにより，上記の操作と平行してビルドを行うことができます。

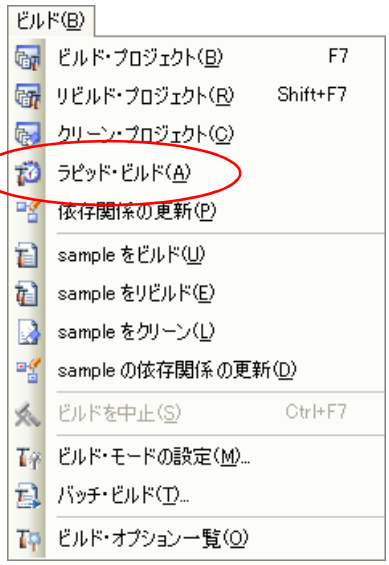
ラピッド・ビルドの有効／無効は，［ビルド］メニュー→［ラピッド・ビルド］の選択により，切り替えます。デフォルトでは，有効となっています。

図 2—112 ［ラピッド・ビルド］項目

【ラピッド・ビルドが有効の場合】



【ラピッド・ビルドが無効の場合】



備考 1. ソース・ファイル編集後，［Ctrl］＋［S］キーの押下により，こまめに上書き保存することを推奨します。

2. ラピッド・ビルドの有効／無効は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト）に対して設定します。

3. ラピッド・ビルドの実行中に，ラピッド・ビルドを無効に切り替えた場合は，その場でラピッド・ビルドの実行を中止します。

注意 この機能は，オプション ダイアログの［ビルド／デバッグ］カテゴリの［登録されたファイルの変更を監視する］をチェックした場合，外部テキスト・エディタでも有効となります。

2. 16. 4 ビルド・モードを一括してビルドを実行する

プロジェクト（メイン・プロジェクト，およびサブプロジェクト）が持つビルド・モードを一括して，ビルド／リビルド／クリーンを行うことができます（以降，“バッチ・ビルド”と呼びます）。

備考 ビルド，リビルド，クリーンについては，それぞれ以下を参照してください。

- ビルド → 「2. 16. 1 更新ファイルのビルドを実行する」参照
- リビルド → 「2. 16. 2 すべてのファイルのビルドを実行する」参照
- クリーン → 「2. 16. 9 中間ファイル，生成ファイルを削除する」参照

［ビルド］メニュー→［バッチ・ビルド ...］を選択すると，バッチ・ビルド ダイアログがオープンします。

図 2—113 バッチ・ビルド ダイアログ

ダイアログ上には，現在開いているプロジェクトが持つメイン・プロジェクト，およびサブプロジェクトの名前と，それらが持つビルド・モード，定義マクロの組み合わせの一覧を表示します。

バッチ・ビルドを行いたいメイン・プロジェクト，およびサブプロジェクトとビルド・モードの組み合わせをチェック・ボックスにより選択し，［ビルド］／［リビルド］／［クリーン］ボタンをクリックしてください。

備考 バッチ・ビルド順は，プロジェクトのビルド順に従い，サブプロジェクト，メイン・プロジェクトの順となります。

1 つのメイン・プロジェクト，またはサブプロジェクトについて複数のビルド・モードを選択した場合は，そのサブプロジェクトで選択しているすべてのビルド・モードでビルドを行ったのち，次のサブプロジェクト，またはメイン・プロジェクトのビルドを行います。

2. 16. 5 複数のファイルのコンパイル／アセンブル／リンクを同時に実行する

ビルド対象ファイルが複数存在する場合，cx コマンド 1 回の呼び出しでファイルを一括してコンパイル／アセンブル／リンクを行うことができます（以降，“一括ビルド”と呼びます）。

cx コマンドの呼び出しイメージを以下に示します。

例 ビルド対象ファイルが aaa.c，bbb.c，ccc.c の場合

- 一括ビルドを行う場合

```
>cx -CF3746 aaa.c bbb.c ccc.c ← a.lmf を生成
```

- 一括ビルドを行わない場合

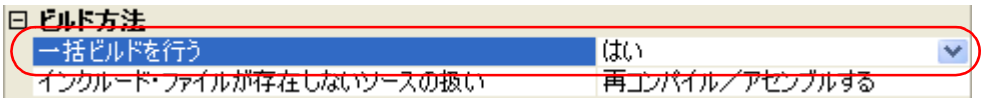
```
>cx -CF3746 aaa.c ← aaa.obj を生成
>cx -CF3746 bbb.c ← bbb.obj を生成
>cx -CF3746 ccc.c ← ccc.obj を生成
>cx -CF3746 aaa.obj bbb.obj ccc.obj ← a.lmf を生成
```

一括ビルドを行うかどうかは，プロパティで設定します。

プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［共通オプション］タブを選択します。

［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで，［はい］を選択してください。

図 2—114 ［一括ビルドを行う］プロパティ



このとき，［アセンブル・オプション］タブは非表示となります。

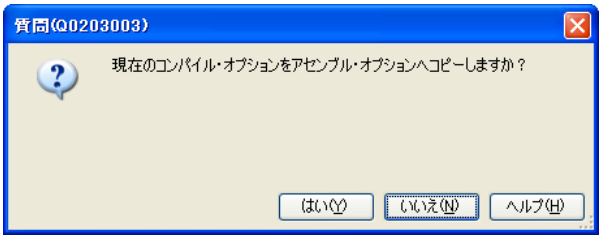
アセンブラ・ソース・ファイル（個別オプションを設定しているファイルを除く）のアセンブルには，［コンパイル・オプション］タブの設定を使用します。

備考 1. 個別オプションを設定しているファイル，およびビルド前実行の対象となっているファイルは，一括ビルドの対象となります。
一括ビルドの対象外となったファイルについては，個別にビルドを行います。

2. ソース・ファイルが，生成するオブジェクト・モジュール・ファイル，および関連するプロパティやプロジェクトなどより古い場合は，ソース・ファイルではなく，オブジェクト・モジュール・ファイルがビルド対象となります。

3. ［一括ビルドを行う］プロパティを［はい］から［いいえ］に変更すると，以下のメッセージ ダイアログがオープンします。

図 2—115 メッセージ ダイアログ



ダイアログ上で［はい］をクリックすると，［コンパイル・オプション］タブの設定を［アセンブル・オプション］タブにコピーします。

［いいえ］をクリックすると，［アセンブル・オプション］タブは非表示となる前の状態で表示します。

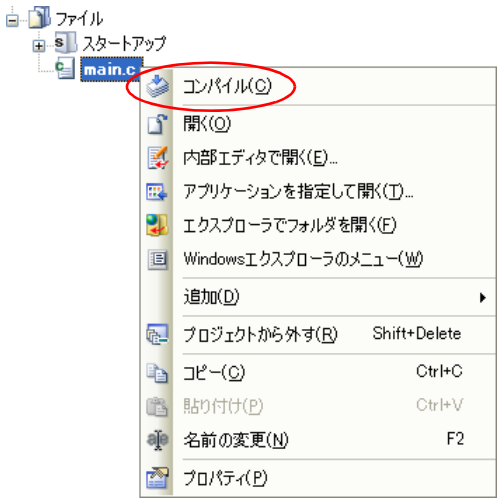
2. 16. 6 ファイル単位でコンパイル／アセンブルする

プロジェクトに追加している各ソース・ファイルに対して，コンパイル，またはアセンブルのみを行うことができます。

(1) C ソース・ファイルをコンパイルする場合

プロジェクト・ツリー で C ソース・ファイルを選択し，コンテキスト・メニューの［コンパイル］を選択してください。

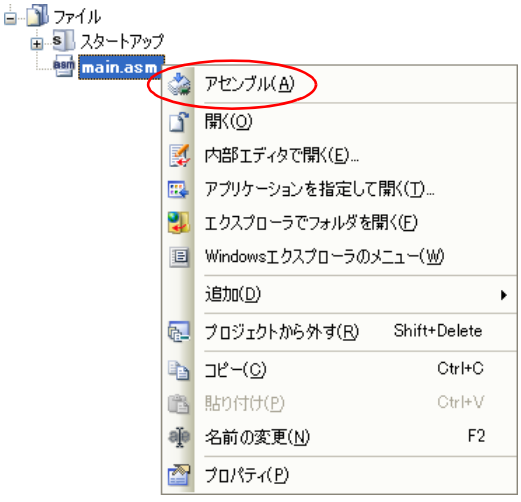
図 2—116 ［コンパイル］項目



(2) アセンブラ・ソース・ファイルをアセンブルする場合

プロジェクト・ツリー でアセンブラ・ソース・ファイルを選択し，コンテキスト・メニューの［アセンブル］を選択してください。

図 2—117 ［アセンブル］項目



2. 16. 7 ビルドの実行を中止する

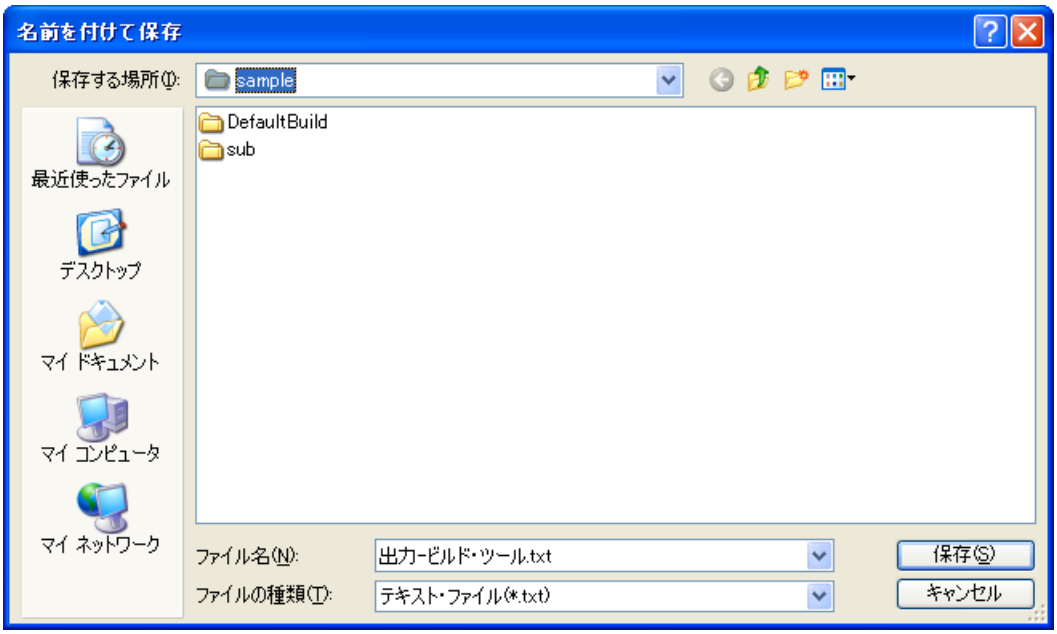
実行中のビルド，リビルド，バッチ・ビルドを中止するには，ツールバーの  ボタンをクリックしてください。

2. 16. 8 ビルド結果をファイルに保存する

出力 パネルに表示するビルドの実行結果（ビルド・ツールの出力メッセージ）をテキスト・ファイルに保存することができます。

パネル上で［ビルド・ツール］タブを選択し，［ファイル］メニュー→［名前を付けて 出力ービルド・ツール を保存 ...］を選択すると，名前を付けて保存 ダイアログがオープンします。

図 2—118 名前を付けて保存 ダイアログ



ダイアログ上で，保存するテキスト・ファイル名と保存場所を指定し，［保存］ボタンをクリックしてください。

2. 16. 9 中間ファイル，生成ファイルを削除する

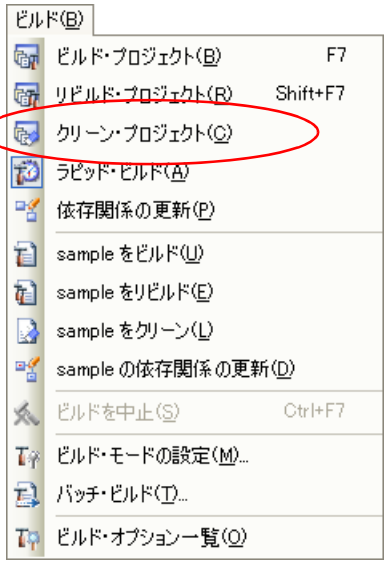
ビルドの実行により出力した中間ファイル，生成ファイルをすべて削除することができます（以降，“クリーン”と呼びます）。

クリーンの実行は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト），またはアクティブ・プロジェクト（「2. 15. 5 ビルド対象プロジェクトを変更する」参照）に対して行います。

(1) プロジェクト全体のクリーンを実行する場合

［ビルド］メニュー→［クリーン・プロジェクト］を選択してください。

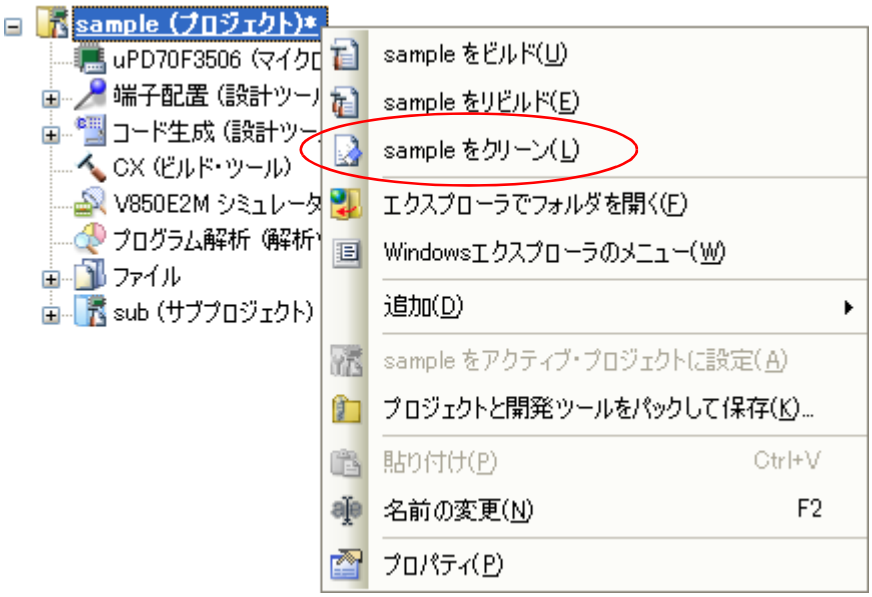
図 2—119 ［クリーン・プロジェクト］項目



(2) アクティブ・プロジェクトのクリーンを実行する場合

プロジェクトを選択し，コンテキスト・メニューの［*アクティブ・プロジェクト*をクリーン］を選択してください。

図 2—120 ［*アクティブ・プロジェクト*をクリーン］項目



2. 17 スタックを見積もる

スタックを見積もるには，スタック見積もりツールを使用します。

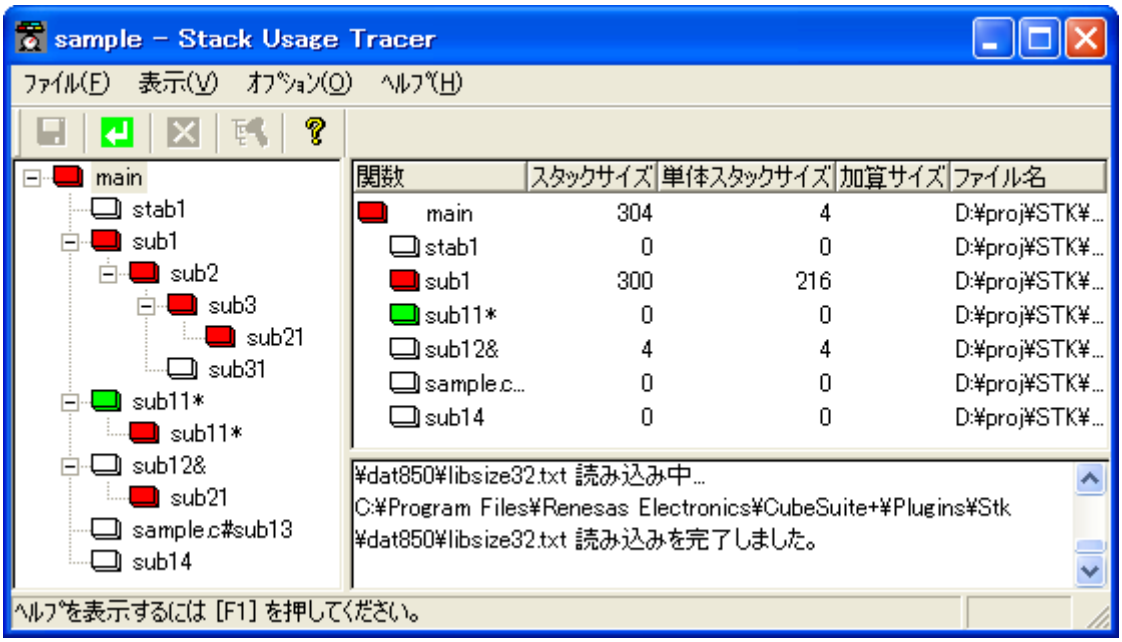
スタック見積もりツールでは，静的に解析処理を行うことにより，関数の呼び出し関係をツリー形式で表示するとともに，関数単位のスタック情報（関数名，スタック・サイズ，単体スタック・サイズ，加算サイズ，ファイル名）をリスト形式で表示します。

2. 17. 1 起動と終了

スタック見積もりツールの起動は，メイン・ウインドウの［ツール］メニュー→［スタック見積もりツールの起動］を選択することにより行います。

なお，スタック見積もりツールの起動が完了した際には，関数の呼び出し関係，および関数単位のスタック情報を Stack Usage Tracer ウインドウのツリー表示エリア／リスト表示エリアに表示します。

図 2—121 スタック見積もりツールの起動イメージ

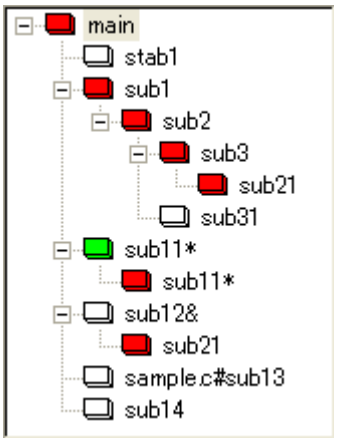


また，スタック見積もりツールの終了は，Stack Usage Tracer ウインドウの［ファイル］メニュー→［skcx の終了］を選択することにより行います。

2. 17. 2 呼び出し関係を確認する

関数の呼び出し関係については，Stack Usage Tracer ウインドウのツリー表示エリアで確認することができます。

図 2—122 ツリー表示エリア



備考 関数名の直前に表示しているアイコンは，以下の意味を持ちます。
なお，アイコンの表示優先度は，高い： ～ 低い：となります。

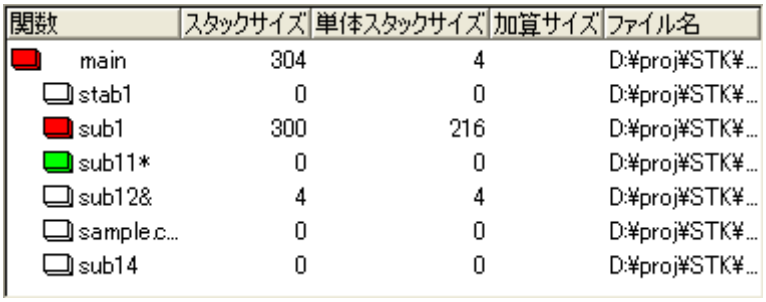
	同じ関数から直接呼び出す関数の中でスタック・サイズが最大となる関数
	スタックサイズ変更 ダイアログ，またはスタック・サイズ指定ファイルにより情報（加算サイズ，再帰回数，呼び出し関数）の変更が行われた関数
	再帰関数
	スタック見積もりツールがスタック情報を取得できていない関数
	上記以外の関数

2. 17. 3 スタック情報を確認する

関数単位のスタック情報（関数名，スタック・サイズ，単体スタック・サイズ，加算サイズ，ファイル名）については，Stack Usage Tracer ウインドウのリスト表示エリアで確認することができます。

- スタック・サイズ（呼び出し関数のスタック・サイズを含む）
- 単体スタック・サイズ（呼び出し関数のスタック・サイズを含まない）
- 加算サイズ（単体スタック・サイズに対して強制的に加算する値）

図 2—123 リスト表示エリア

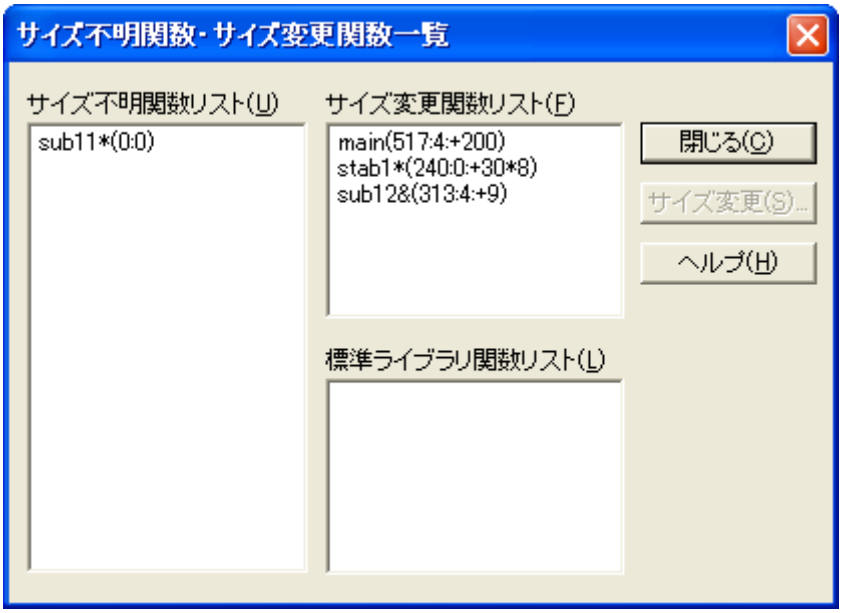


備考 スタック見積もりツールの起動中に，プロジェクトに登録しているファイルに対してスタック・サイズが変わるような記述変更などを行った際には，リビルド後，ボタンをクリックし，表示内容の更新を行ってください。

2. 17. 4 不明関数を確認する

スタック見積もりツールがスタック情報を取得できていない関数については，サイズ不明関数・サイズ変更関数一覧 ダイアログの［サイズ不明関数リスト］で確認することができます。

図 2—124 サイズ不明関数・サイズ変更関数一覧 ダイアログ



備考 以下の場合，［サイズ不明関数リスト］に該当関数を表示します。

- 単体スタック・サイズを計測することができなかった関数
- スタックサイズ変更 ダイアログで再帰回数の設定が行われていない再帰関数
- スタックサイズ変更 ダイアログで呼び出し関数の設定が行われていない間接関数呼び出しを含む関数

2. 17. 5 スタック・サイズを変更する

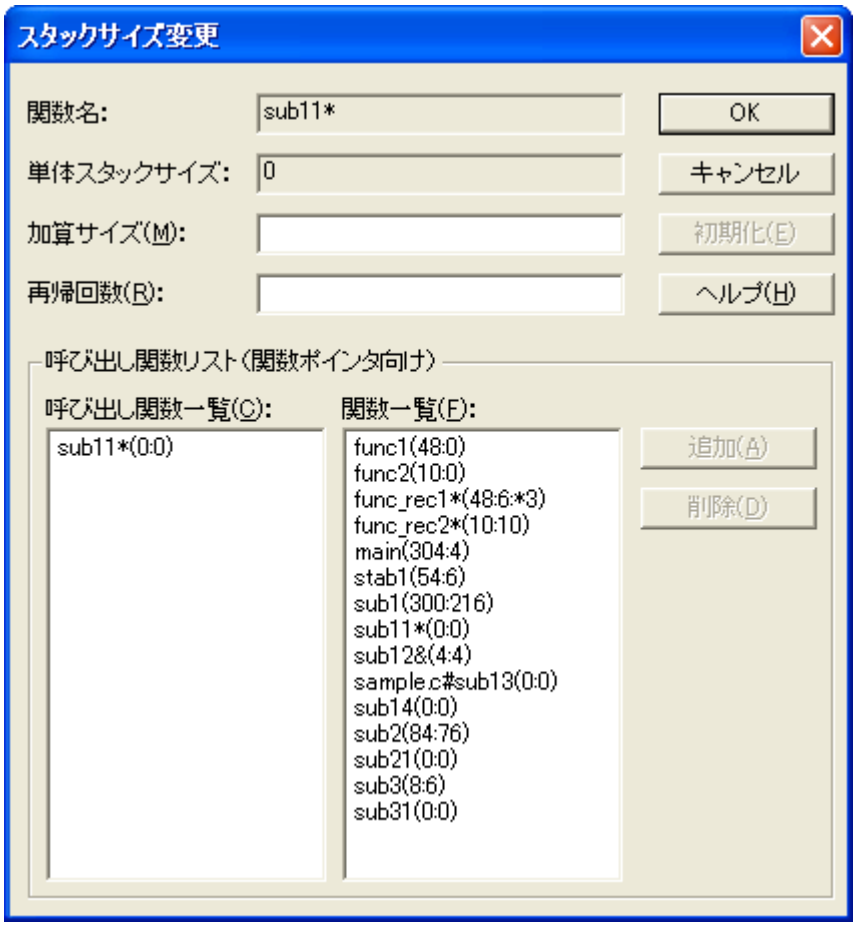
スタック見積もりツールがスタック情報を取得できていない関数，および意図的に情報を変更したい関数については，スタックサイズ変更 ダイアログ，またはスタック・サイズ指定ファイルを用いることにより，該当値を動的に設定することができます。

(1) スタックサイズ変更 ダイアログを用いる場合

スタックサイズ変更 ダイアログを用いる場合は，以下の操作手順となります。

- Stack Usage Tracer ウインドウのツリー表示エリアで該当関数を選択したのち，ツールバー→  ボタンをクリックし，スタックサイズ変更 ダイアログをオープン

図 2—125 スタックサイズ変更 ダイアログ



- ［加算サイズ］，［再帰回数］，［呼び出し関数一覧］を設定したのち，［OK］ボタンをクリック

(2) スタック・サイズ指定ファイルを用いる場合

スタック・サイズ指定ファイルを用いる場合は，以下の操作手順となります。

- スタック・サイズ指定ファイルの作成

スタック・サイズ指定ファイルでは，動的に設定したい関数を以下の形式で記述します。

関数名［, ADD= 加算サイズ］［, RECTIME= 再帰回数］［, CALL= 呼び出し関数］...

```
# アセンブリ言語で記述した関数 _flib の単体スタック・サイズを 50 に設定
[flib], ADD=50

#C 言語で記述した関数 sub2 の単体スタック・サイズを 100 に設定
sub2, ADD=100

#C 言語で記述した再帰関数 sub3 の再帰回数を 123 に設定
sub3, RECTIME=123
```

- Stack Usage Tracer ウインドウの［ファイル］メニュー→［スタックサイズ指定ファイルを開く］を選択することによりオープンするファイルを開く ダイアログで該当スタック・サイズ指定ファイルを指定したのち，［開く］ボタンをクリック

第３章　ビルドの出力リスト

この章では，ビルドにより CX が出力する各種ファイルのフォーマットなどについて説明します。

3.1　アセンブル・リスト・ファイル

ここでは，アセンブル・リスト・ファイルについて説明します。
アセンブル・リストとは，ソースをコンパイル，アセンブルして出力するコードをリスト形式にしたものです。
これにより，コンパイル，アセンブルした結果が，どのようなコードになっているかを確認することができます。

（1）出力方法

アセンブル・リスト・ファイルの出力方法を以下に示します。

（a）CubeSuite+ の場合

プロジェクト・ツリー パネルでビルド・ツール・ノードを選択したのち，プロパティ パネルの［コンパイル・オプション］タブを選択します。

アセンブル・リスト・ファイルを出力するには，［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択します。

出力先は，［アセンブル・リスト・ファイル出力フォルダ］プロパティで指定します。

ファイル名は，ソース・ファイルの拡張子を .prn で置き換えた名前となります。

なお，［出力コード］カテゴリの［アセンブラ・ソースにコメントを出力する］プロパティで［はい (-Xpass_source)］を選択すると，出力するアセンブル・リスト・ファイル中に，アセンブラ・ソース・プログラムに対応した C ソース・プログラムをコメントとして出力します。

（b）コマンド・ラインの場合

-Xprn_path オプションを指定すると，カレント・フォルダに，ソース・ファイル名の拡張子を .prn で置き換えた名前でアセンブル・リスト・ファイルを出力します。

-Xprn_path オプションのパラメータで，ファイル名を指定することもできます。

-Xpass_source オプションを同時に指定すると，出力するアセンブル・リスト・ファイル中に，アセンブラ・ソース・プログラムに対応した C ソース・プログラムをコメントとして出力します。

（2）出力例

以下の C ソース・ファイルをコンパイルし，さらに出力したアセンブラ・ソース・ファイルをアセンブルすることにより出力したアセンブル・リスト・ファイルの例を示します。

- C ソース・ファイル

```
void main(void) {
    int     a;
}
```

- アセンブル・リスト・ファイル

```
(1)  (2)      (3)          (4)          (5)
        :
A-X- 00000000              16           .file   “a.c“
A-X- 00000000              17           .align  4
A-X- 00000000              18           #@BF
A-X- 00000000              19           .func   _main, .F2-.F2.end, 20
A-X- 00000000              20           .public _main
A-X- 00000000              21  _main:
A-X- 00000000 0C8A         22           jbr     .L4
A-X- 00000002              23  .L5:
A-X- 00000002 4001         24           mov     r0, r10
A-X- 00000004 E3CF0000     25           ld.w    -4+.F2[sp], lp
A-X- 00000008 6444         26           add     .F2, sp
A-X- 0000000A 1F18         27           jmp     [lp]    --1
A-X- 0000000C              28  .L4:
A-X- 0000000C              29           sub     .F2, sp
A-X- 0000000C 2440             -- mov   0x4, r1
A-X- 0000000E 6108             -- sub   r1, sp
A-X- 00000010 E3DF0000     30           st.w    lp, -4+.F2[sp]
A-X- 00000014 EE8B         31           jbr     .L5
A-X- 00000016              32           #@FUNC_ARG
A-X- 00000016              33           .F2     .set   0x4
A-X- 00000016              34           .A2     .set   0x0
A-X- 00000016              35           .T2     .set   0x0
A-X- 00000016              36           #@EF
        :
```

項番	説明
(1)	セクション属性 その行のソース・プログラムに対して生成したコードを格納するセクションのセクション属性です。 セクション属性とその意味を以下に示します。 A：メモリを占有するセクション W：書き込み可能なセクション X：実行可能なセクション G：グローバル・ポインタ（gp）と 16 ビットのディスプレースメントを用いて参照することのできるメモリの範囲に割り付けるセクション

項番	説明
(2)	ロケーション・カウンタ値 その行のソース・プログラムに対して生成したコードの先頭に対するロケーション・カウンタ値です。
(3)	コード その行のソース・プログラムに対して生成したコード（機械語命令，またはデータ）です。 1 バイトごとに 2 桁の 16 進数で表記します。
(4)	行番号 その行の行番号です。 10 進数で表記します。
(5)	ソース・プログラム その行のソース・プログラムです。 その行の命令に対して命令展開が生じた場合，その命令展開によって生成した機械語命令の命令列を逆アセンブルした結果を，“--”以降に示します。

3.2　リンク・マップ・ファイル

ここでは，リンク・マップ・ファイルについて説明します。

リンク・マップとは，リンク結果の情報が書かれたもので，セクションの配置アドレスなどの情報を知ることができます。

（1）出力方法

リンク・マップ・ファイルの出力方法を以下に示します。

（a）CubeSuite+ の場合

プロジェクト・ツリー パネルでビルド・ツール・ノードを選択したのち，プロパティ パネルの［リンク・オプション］タブを選択します。

リンク・マップ・ファイルを出力するには，［リンク・マップ］カテゴリの［リンク・マップ・ファイルを出力する］プロパティで［はい (-Xmap)］を選択します。

出力先は，［リンク・マップ・ファイル出力フォルダ］プロパティ，および［リンク・マップ・ファイル名］プロパティで指定します。

また，プロジェクト・ツリーのビルド・ツール生成ファイル・ノードにも表示します。

（b）コマンド・ラインの場合

-Xmap オプションを指定すると，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .map に置き換えた名前でリンク・マップ・ファイルを出力します。

-Xmap オプションのパラメータで，ファイル名を指定することもできます。

なお，-Xno_romize オプションの有無により，出力するリンク・マップ・ファイルは以下のようになります。

- -Xno_romize オプションの指定あり
  リンク処理後までのリンク・マップ・ファイルを出力します。
- -Xno_romize オプションの指定なし
  ROM 化処理後までのリンク・マップ・ファイルを出力します。

（2）出力例

シンボル情報出力（-Xsymbol_dump オプションの指定あり），および ROM 化処理（-Xno_romize オプションの指定なし）を指定して，以下のオブジェクトをリンクした場合に出力するリンク・マップ・ファイルの例を示します。

- test.obj（ユーザ指定オブジェクト・モジュール・ファイル）
- cstart.obj（スタートアップ・ルーチン）
- libc.lib（標準ライブラリ）

```
            ********** MEMORY ALLOCATION MAP ***********
(1)         (2)                 (3)          (4)          (5)
OUTPUT      SEGMENT             VIRTUAL      SIZE(16)     SIZE(10)
SEGMENT     ATTRIBUTE           ADDRESS

INT         RX                  0x00000000   0x00000080        128
TEXT        RX                  0x00000080   0x00000338        824
DATA        RW                  0x03ffd000   0x00000200        512

            ******** LINK EDITOR ALLOCATION MAP ********
(6)         (7)                 (8)          (9)          (10)
OUTPUT      INPUT               VIRTUAL      SIZE         INPUT
SECTION     SECTION             ADDRESS                   FILE
RESET
                                0x00000000   0x00000004
            RESET
                                0x00000000   0x00000004   D:¥lib¥cstart.obj
SECURITY_ID
                                0x00000070   0x0000000a
            SECURITY_ID
                                0x00000070   0x0000000a   *(nil)*
OPTION_BYTES
                                0x0000007a   0x00000006
            OPTION_BYTES
                                0x0000007a   0x00000006   *(nil)*
.pro_epi_runtime
                                0x00000080   0x000001e0
            .pro_epi_runtime
                                0x00000080   0x000001e0   callt.obj(D:¥lib¥libc.lib)
.text
                                0x00000260   0x00000158
            .text
                                0x00000260   0x00000066   D:¥lib¥850e¥cstart.obj
            .text
                                0x000002c6   0x0000002a   D:¥work¥test.obj
            .text
                                0x000002f0   0x00000002   hdwinit.obj(D:¥lib¥850e¥libc.lib)
            .text
                                0x000002f2   0x000000c6   rcopy.obj(D:¥lib¥850e¥libc.lib)
            .text
                                0x000003b8   0x00000000   D:¥lib¥850e¥rompcrt.obj
.sbss
                                0x03ffd000   0x00000000
            .sbss
                                0x03ffd000   0x00000000   D:¥lib¥850e¥cstart.obj
```

```
.bss
                              0x03ffd000  0x00000200
            .bss
                              0x03ffd000  0x00000200  D:¥lib¥850e¥cstart.obj

            ***************** SYMBOL DUMP *****************
    (11)(12)        (13)        (14)    (15)        (16)
    No. Value       Size        Bind    Type        Name
      1 0x0         0x0         Local   File        cstart.asm
      2 0x2c0       0x0         Local   Object      __zeroclrw
      3 0x3ffd000   0x200       Local   Object      __stack
      4 0x0         0x0         Local   File        ..¥work¥nomal.c
      5 0x0         0x0         Local   Devfile     Df3737.800
                ........ 中略 ........
    106 0x3ffd200   0x0         Global  Object      __end
    107 0x70        0x0         Global  Object      __sSECURITY_ID
    108 0x7a        0x0         Global  Object      __eSECURITY_ID
    109 0x7a        0x0         Global  Object      __sOPTION_BYTES
    110 0x80        0x0         Global  Object      __eOPTION_BYTES

            ******** ROM PROCESSOR MEMORY MAP ********
(17)        (18)              (19)        (20)
OUTPUT      INPUT             VIRTUAL     SIZE
SECTION     SECTION           ADDRESS
RESET
            RESET
                              0x00000000  0x00000004
SECURITY_ID
            SECURITY_ID
                              0x00000070  0x0000000a
OPTION_BYTES
            OPTION_BYTES
                              0x0000007a  0x00000006
.pro_epi_runtime
            .pro_epi_runtime
                              0x00000080  0x000001e0
.text
            .text
                              0x00000260  0x00000158
rompsec
                              0x000003b8  0x00000008
            *(nil)*
                              0x000003b8  0x00000008  *(copy-info)*
```

項番	説明
(1)	出力セグメント 生成するロード・モジュール・ファイルを構成する出力セグメントの名前です。 生成するロード・モジュール・ファイルには，出力セグメントの名前は格納しません。
(2)	セグメント属性 セグメント属性とその意味を以下に示します。 R：読み出し可能 W：書き込み可能 X：実行可能
(3)	アドレス 出力セグメントの先頭アドレスです。
(4)	サイズ（16 進数） セクション間の整列条件やアラインホールを含めたメモリのサイズです。
(5)	サイズ（10 進数） セクション間の整列条件やアラインホールを含めたメモリのサイズです。
(6)	出力セクション 生成するロード・モジュール・ファイルを構成する出力セクションの名前です。
(7)	入力セクション 各出力セクションに割り付けられた入力セクション名前です。
(8)	アドレス 出力セクション，および入力セクションの先頭アドレスです。
(9)	サイズ 出力セクション，および入力セクションのサイズです。
(10)	入力ファイル 入力セクションの属しているオブジェクト・モジュール・ファイルの名前です。 リンク処理によって生成するセクション，およびアセンブル処理によって生成する .symtab，.strtab，.shstrtab などに対しては，“*(nil)*”と表示します。 また，オブジェクトを示す識別子を外部結合で仮定義した場合（アセンブリ言語では .comm 疑似命令を用いて領域確保した場合），全ファイル共通の領域となり，そのセクションに対しては，“*(Common)*”，または“*(GpCommon)*”と表示します。 入力セクションの属しているオブジェクト・モジュール・ファイルがライブラリ・ファイル内のオブジェクト・モジュール・ファイルである場合，“オブジェクト・モジュール・ファイル名（ライブラリ・ファイル名）”の形式で，ライブラリ・ファイルの名前も表示します。
(11)	インデックス シンボルのインデックスです（昇順）。
(12)	値 シンボルに関する値です。
(13)	サイズ シンボルに関するサイズです。 “0L”の場合は，サイズを持たない，またはサイズが未定です。

項番	説明
(14)	シンボル Bind 属性 シンボル Bind 属性とその意味を以下に示します。 Local　：ローカル・シンボル Global　：グローバル・シンボル Weak　：ウィーク・シンボル Num　：シンボル・バインディング・クラス定義数
(15)	シンボル・タイプ シンボル・タイプとその意味を以下に示します。 Notype　：未定義 Object　：オブジェクト（データ） Func　：手続き（関数） Section　：セクション名 File　：ファイル名 Devfile　：デバイス・ファイル名
(16)	シンボル名 シンボルの名前です。
(17)	出力セクション 生成するオブジェクト・ファイルを構成する出力セクションの名前です。
(18)	入力セクション 各出力セクションに割り付けられた入力セクション名前です。
(19)	アドレス 出力セクション，および入力セクションの先頭アドレスです。
(20)	サイズ 出力セクション，および入力セクションのサイズです。

3.3　シンボル情報ファイル

ここでは，シンボル情報ファイルについて説明します。

シンボル情報ファイルとは，C ソース・ファイル内で定義した変数（グローバル変数，ファイル内 static 変数，関数内 static 変数）の配置情報，および変数，関数の参照回数などの情報を記述したテキスト形式のファイルです。

(1) 出力方法

シンボル情報ファイルの出力方法を以下に示します。

(a) CubeSuite+ の場合

プロジェクト・ツリー パネルでビルド・ツール・ノードを選択したのち，プロパティ パネルの［リンク・オプション］タブを選択します。

シンボル情報ファイルを出力するには，［シンボル情報］カテゴリの［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択します。

出力先は，［シンボル情報ファイル出力フォルダ］プロパティ，および［シンボル情報ファイル名］プロパティで指定します。

また，プロジェクト・ツリーのファイル・ノードにも表示します。

(b) コマンド・ラインの場合

-Xsfg オプションを指定すると，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .sfg で置き換えた名前でシンボル情報ファイルを出力します。

-Xsfg オプションのパラメータで，ファイル名を指定することもできます。

(2) 出力例

C ソース・ファイルをコンパイルすることにより出力したシンボル情報ファイルの例を以下に示します。

```
(1)## Section and Variable Information ##
(2)// Global-Variable             : Name, ReferenceCount, ByteSize
(2)// Static-Variable in file     : Name, ReferenceCount, ByteSize, "FileName"
(2)// Static-Variable in function : Name, ReferenceCount, ByteSize, "FileName", FunctionName

(3)[tidata.byte]:Size_128bytes
(11)func_static_UC,(12)5,(13)1,(14)"D:¥develop¥tp¥xx_2.c",(15)sub2
(11)func_static_UC,(12)4,(13)1,(14)"..¥tp¥xx_1.c",(15)main
(4)ti_data_byte,(5)1,(6)1
(4)ti_bss_byte,(5)1,(6)1
(11)@func_static_unused_UC,(12)0,(13)1,(14)"D:¥develop¥tp¥xx_2.c",(15)sub2

(3)[tidata.word]:Size_128bytes
(4)global_int_module_xx1_symbol,(5)6,(6)4
(4)global_const_int_module_xx1_symbol,(5)5,(6)4
(11)func_static_UL,(12)5,(13)4,(14)"D:¥develop¥tp¥xx_2.c",(15)sub2
(7)file_static_int_module_xx2,(8)5,(9)4,(10)"D:¥develop¥tp¥xx_2.c"
```

```
(11)func_static_UL,(12)4,(13)4,(14)"..¥tp¥xx_1.c",(15)main
(7)file_static_int_module_xx1,(8)4,(9)4,(10)"..¥tp¥xx_1.c"
(4)global_const_int_module_xx2_symbol,(5)4,(6)4
(7)static_module_xx1_symbol,(8)1,(9)2,(10)"..¥tp¥xx_1.c"
(4)si_data,(5)1,(6)4
(4)si_bss,(5)1,(6)4
(4)se_data,(5)1,(6)4
(4)se_bss,(5)1,(6)4
(4)struct_xx2_sub2,(5)1,(6)24
(11)@func_static_unused_UC,(12)0,(13)4,(14)"D:¥develop¥tp¥xx_2.c",(15)sub2
(4)@struct_xx2_unused,(5)0,(6)24
(4)@global_const_int_module_xx1_unusedsymbol,(5)0,(6)4

(3)[sidata]:Size_32512bytes

(3)[sedata]:Size_32768bytes

(3)[sdata]:Size_65536bytes


(16)## Function Information ##
(2)// FunctionName, StartAddress, FunctionByteSize, ReferenceCount, StackByteSize, "FileName"

(17)sub1,(18)0x00000210,(19)24,(20)2,(21)0,(22)"..¥tp¥xx_1.c"
(17)sub2,(18)0x00000080,(19)174,(20)2,(21)0,(22)"D:¥tp¥xx_2.c"
(17)@unusedfunc2,(18)0x0000012E,(19)12,(20)0,(21)0,(22)"D:¥tp¥xx_2.c"
(17)@main,(18)0x0000013A,(19)214,(20)0,(21)0,(22)"..¥tp¥xx_1.c"
(17)@unusedfunc1,(18)0x00000228,(19)12,(20)0,(21)0,(22)"..¥tp¥xx_1.c"
```

項番	説明
(1)	変数のセクション配置情報の開始 コンパイラは，次の行以降に記述された変数のセクション配置情報を参照します。
(2)	コメント
(3)	セクション名とセクション・サイズ 変数の配置セクションの名前とサイズを“[*セクション名*]:Size_*10 進数のバイト・サイズ*bytes”の形式で出力します。 -Xsfg_opt オプション指定時のみ出力します。
(4)	変数名 グローバル変数の名前です。 参照回数が 0 回の場合は，先頭に @ を付与し，未使用変数であることを示します。
(5)	参照回数 グローバル変数の参照回数です。
(6)	サイズ グローバル変数のバイト・サイズです。

項番	説明
(7)	変数名 ファイル内 static 変数の名前です。 参照回数が 0 回の場合は，先頭に @ を付与し，未使用変数であることを示します。
(8)	参照回数 ファイル内 static 変数の参照回数です。
(9)	サイズ ファイル内 static 変数のバイト・サイズです。
(10)	ファイル名 static 変数を定義しているファイルの名前をパス付きで出力します。
(11)	変数名 関数内 static 変数の名前です。 参照回数が 0 回の場合は，先頭に @ を付与し，未使用変数であることを示します。
(12)	参照回数 関数内 static 変数の参照回数です。
(13)	サイズ 関数内 static 変数のバイト・サイズです。
(14)	ファイル名 関数内 static 変数を定義しているファイルの名前をパス付きで出力します。
(15)	関数名 static 変数を定義している関数の名前です。
(16)	関数情報の開始
(17)	関数名 関数の名前です。 参照回数が 0 回の場合は，先頭に @ を付与し，未使用関数であることを示します。
(18)	開始アドレス 関数を定義している開始アドレスです。
(19)	関数サイズ 関数のバイト・サイズです。
(20)	参照回数 関数の参照回数です。 -Xdelete_func オプション指定時に実行する未使用関数の削除において，参照回数が 0 でない関数を削除対象から除外します。
(21)	スタック・サイズ 関数のスタック使用量のバイト・サイズです。
(22)	ファイル名 関数を定義しているファイルの名前をパス付きで出力します。

3.4　ヘキサ・ファイル

ここでは，ヘキサ・ファイルについて説明します。

ヘキサ・ファイルとは，実行可能なロード・モジュール・ファイルをヘキサ・フォーマットで変換したファイルです。

ヘキサ・フォーマットとして，以下を指定することができます。

- インテル拡張ヘキサ・フォーマット
- モトローラ S タイプ・ヘキサ・フォーマット
- 拡張テクトロニクス・ヘキサ・フォーマット

（1）出力方法

CX では，ヘキサ・ファイルをデフォルトで出力します。

ヘキサ・フォーマットの設定方法を以下に示します。

（a）CubeSuite+ の場合

プロジェクト・ツリー パネルでビルド・ツール・ノードを選択したのち，プロパティ パネルの［ヘキサ出力オプション］タブを選択します。

ヘキサ・フォーマットは，［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイル・フォーマット］プロパティで設定します。

なお，ヘキサ・ファイルは，プロジェクト・ツリーのビルド・ツール生成ファイル・ノードにも表示します。

（b）コマンド・ラインの場合

-Xhex_format オプションのパラメータでフォーマットを指定します。

指定形式を以下に示します。

```
-Xhex_format=format
```

format に指定可能なものを以下に示します。

I	インテル拡張ヘキサ・フォーマット（1M バイトまで）
i	インテル拡張ヘキサ・フォーマット（32 ビット・アドレス）（4G バイトまで）
S	モトローラ S タイプ・ヘキサ・フォーマット（スタンダード・アドレス）（16M バイトまで）
s	モトローラ S タイプ・ヘキサ・フォーマット（32 ビット・アドレス）（4G バイトまで）
T	拡張テクトロニクス・ヘキサ・フォーマット（4G バイトまで）

3. 4. 1 インテル拡張ヘキサ・フォーマット

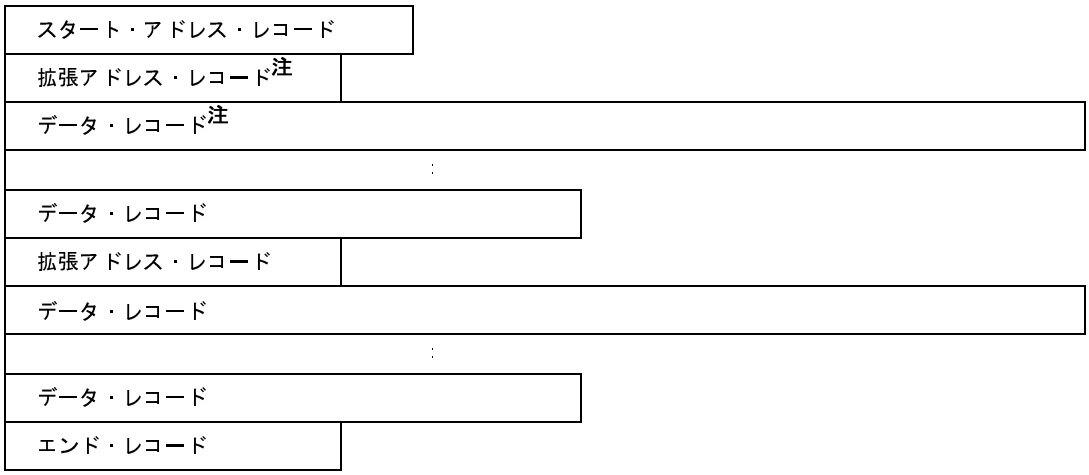
インテル拡張ヘキサ・フォーマット（20 ビット）のファイルは，スタート・アドレス・レコード，拡張アドレス・レコード，データ・レコード，およびエンド・レコードの 4 種類のレコード[注]により構成されます。

インテル拡張ヘキサ・フォーマット（32 ビット）のファイルは，スタート・リニア・アドレス・レコード，拡張リニア・スタート・アドレス・レコード，スタート・アドレス・レコード，拡張アドレス・レコード，データ・レコード，およびエンド・レコードの 6 種類のレコード[注]により構成されます。

注 各レコードは，ASCII コードで出力します。

インテル拡張ヘキサ・フォーマットのファイル構成を以下に示します。

図 3—1 インテル拡張ヘキサ・フォーマットのファイル構成



注 拡張アドレス・レコード，およびデータ・レコードは繰り返されます。

各レコードは，各種フィールドにより以下の形で構成されます。

```
:   XX  XXXX  XX  DD......DD SS  NL
(1) (2) (3)   (4)     (5)    (6) (7)
```

項番	説明
(1)	レコード・マーク
(2)	バイト数 (5) の 2 桁ずつの 16 進数で表されるバイトのバイト数です。
(3)	ロケーション・アドレス

項番	説明
(4)	レコード・タイプ 05 ： スタート・リニア・アドレス・レコード 04 ： 拡張リニア・アドレス・レコード 03 ： スタート・アドレス・レコード 02 ： 拡張アドレス・レコード 00 ： データ・レコード 01 ： エンド・レコード
(5)	コード コードの 1 バイトごとを 2 桁の 16 進数で表したものです。
(6)	チェック・サム ：，SS，NL を除くレコード内の 2 桁ずつの 16 進数で表される値を初期値 0 から順に減算し，その下位 1 バイトを 2 桁の 16 進数で表したものです。
(7)	ニュー・ライン（¥ n）

- スタート・リニア・アドレス・レコード（32 ビット・アドレス指定時）

リニア・アドレスを示します。

```
:   04  0000  05  XXXXXXXX  SS  NL
(1) (2) (3)   (4)    (5)
```

項番	説明
(1)	レコード・マーク
(2)	04 固定
(3)	0000 固定
(4)	05 固定
(5)	リニア・アドレス値

- 拡張リニア・アドレス・レコード（32 ビット・アドレス指定時）

ビット 32 ～ビット 16 の上位 16 ビット・アドレスを示します。

```
:   02  0000  04  0000  SS  NL
(1) (2) (3)   (4)   (5)
```

項番	説明
(1)	レコード・マーク
(2)	02 固定
(3)	0000 固定
(4)	04 固定
(5)	ビット 32 ～ビット 16 の上位 16 ビット・アドレス値

注　下位 16 ビットは，データ・レコードのロケーション・アドレスを用います。

- スタート・アドレス・レコード

エントリ・ポイント・アドレスを示します。

```
:   04  0000  03  PPPP  OOOO  SS  NL
(1) (2) (3)   (4)  (5)   (6)
```

項番	説明
(1)	レコード・マーク
(2)	04 固定
(3)	0000 固定
(4)	03 固定
(5)	エントリ・ポイント・アドレスのパラグラフ値注
(6)	エントリ・ポイント・アドレスのオフセット値

注 アドレスは（パラグラフ値 <<4）＋オフセット値で求められます。

- 拡張アドレス・レコード

ロード・アドレスのパラグラフ値を示します注。

注 （データ・レコードを出力する際）セグメントの先頭で，またはデータ・レコードのロード・アドレスのオフセット値が最大値 0xffff を越えてセグメントが新しくなる際，出力します。

```
:   02  0000  02  PPPP  SS  NL
(1) (2) (3)   (4) (5)
```

項番	説明
(1)	レコード・マーク
(2)	02 固定
(3)	0000 固定
(4)	02 固定
(5)	セグメントのパラグラフ値

- データ・レコード

コードの値を示します。

```
:   XX  XXXX  00  DD......DD  SS  NL
(1) (2) (3)   (4)     (5)
```

項番	説明
(1)	レコード・マーク

項番	説明
(2)	バイト数[注]
(3)	ロケーション・アドレス
(4)	00 固定
(5)	コード コードの 1 バイトごとを 2 桁の 16 進数で表したものです。

注 0x1 ～ 0xff の範囲に限られます（1 つのデータ・レコードで示されるコードのバイト数の最小値は 1 で最大値は 255 です）。

例

```
:   04  0100  00  3C58E01B  6C  NL
(1) (2) (3)   (4)    (5)    (6) (7)
```

項番	説明
(1)	レコード・マーク
(2)	3C58E01B の 2 桁ずつの 16 進数で表されるバイトのバイト数
(3)	ロケーション・アドレス
(4)	レコード・タイプ 00
(5)	コードの 1 バイトごとを 2 桁の 16 進数で表したもの
(6)	チェック・サム 04 ＋ 01 ＋ 00 ＋ 00 ＋ 3C ＋ 58 ＋ E0 ＋ 1B ＝ 194 の 2 の補数 E6C の下位 1 バイトを 2 桁の 16 進数で表したもの
(7)	ニュー・ライン（¥n）

- エンド・レコード

コードの終わりを示します。

```
:   00  0000  01  FF  NL
(1) (2) (3)   (4) (5)
```

項番	説明
(1)	レコード・マーク
(2)	00 固定
(3)	0000 固定
(4)	01 固定
(5)	FF 固定

備考 インテル・ヘキサ・フォーマットのロケーション・アドレスは 2 バイト（16 ビット）です。
したがって，64K の空間しか直接指定はできません。
それを拡張するために，16 ビットの拡張アドレスを追加して 1M（20 ビット）の空間まで扱えるようにし

たのがインテル拡張ヘキサ・フォーマットです。
具体的には，16 ビットの拡張アドレスを指定するレコードタイプを追加しています。
この追加した拡張アドレスの 4 ビットをシフトしてロケーション・アドレスと加算することで，20 ビットのアドレスを表現できるようになっています。
たとえば，FFFFFH を示す場合には，拡張アドレスに F000H を設定し，ロケーション・アドレスに FFFFH を指定します。
このように，インテル拡張ヘキサ・フォーマットでは，0 ～ FFFFFH までしかアドレッシングできません。
100000H のような場合には，別のオブジェクト形式を使用する必要があります。
CX では，このアドレス，サイズにおいて，このフォーマットの規定に違反していた場合，メッセージを出力します。
インテル拡張ヘキサ・フォーマットの場合，表現可能な値は 20 ビット，つまり，1M（0x100000）バイトになります。

```
W0562022: 先頭アドレスが、インテル拡張ヘキサ・フォーマット形式で表現可能なアドレスの最大値（20 ビット）を越えています。表現可能な範囲をアドレスとして出力する形で処理を続行します。
```

“W0562022”のメッセージを出力した場合は，ヘキサ変換する領域の開始アドレスが，1 M バイトを越えている場合になります。

```
W0562020: アドレスが、インテル拡張ヘキサ・フォーマット形式で表現可能なアドレスの最大値（20 ビット）を越えています。表現可能な範囲をアドレスとして出力する形で処理を続行します。
```

“W0562020”のメッセージを出力した場合は，ヘキサ変換を行おうとするアドレスが 1M（20 ビット）を越えた場合になります。
以下の例のような場合には，1M を越えなくても，上記のエラーが発生します。

例 1. -Xhex_offset オプションで指定したアドレスを起点とするオフセットとしない
→ 絶対アドレスをヘキサ・フォーマットに格納する

2. 20 ビットで表現可能なアドレスの上限付近にセクションを配置
→開始アドレスは 20 ビットに収まっているが，セクションの途中から 20 ビットを越える

この 2 パターンに合致する場合には，変換する領域がわずかに 4 バイトであるとしても，“W0562020”のメッセージが発生します。

3. 4. 2　モトローラ S タイプ・ヘキサ・フォーマット

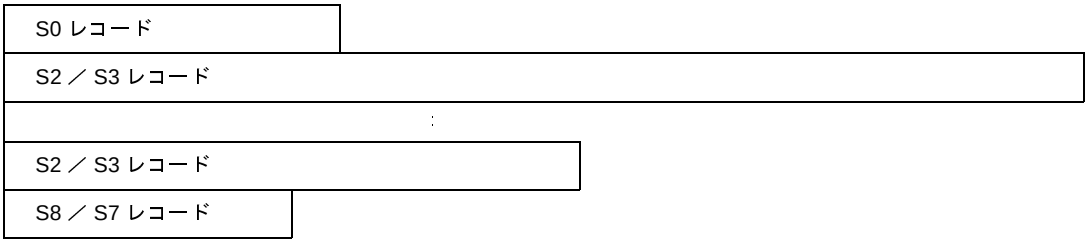
モトローラ S タイプ・ヘキサ・フォーマットのファイルは，ヘッダ・レコードである S0 レコード，データ・レコードである S2 ／ S3 レコード，エンド・レコードである S8 ／ S7 レコードの 5 種類のレコード[注 1] により構成されます[注 2]。

注 1. 各レコードは，ASCII コードで出力します。

2. モトローラ S タイプ・ヘキサ・フォーマットには，（24 ビット）スタンダード・アドレスのものと，32 ビット・アドレスのものとが存在し，スタンダード・アドレスのフォーマットは S0，S2，および S8 レコード，32 ビット・アドレスのフォーマットは S0，S3，および S7 レコードによって構成されます。

モトローラ S タイプ・ヘキサ・フォーマットのファイル構成を以下に示します。

図 3―2　モトローラ S タイプ・ヘキサ・フォーマットのファイル構成



各レコードは，各種フィールドにより以下の形で構成されます。

```
Sx  XX  YY......YY  SS  NL
(1) (2)     (3)     (4) (5)
```

項番	説明
(1)	レコード・タイプ
(2)	レコード長 (3) の 2 桁ずつの 16 進数で表されるバイトのバイト数＋ SS で表されるバイト数[注]
(3)	フィールド
(4)	チェック・サム Sx，SS，NL を除くレコード内の 2 桁ずつの 16 進数で表されるバイトの値を合計したものの 1 の補数を取り，その下位 1 バイトを 2 桁の 16 進数で表したもの
(5)	ニュー・ライン（¥ n）

注　1 です。

- S0 レコード

ファイル名を示します。

```
S0  XX  XX......XX  SS  NL
(1) (2)     (3)
```

項番	説明
(1)	S0 固定
(2)	レコード長
(3)	ファイル名 指定したファイル名の ASCII コード表示

- S2 レコード

コードの値を示します。

```
S2  XX  YYYYYY  ZZ......ZZ  SS  NL
(1) (2)   (3)       (4)
```

項番	説明
(1)	S2 固定
(2)	レコード長
(3)	ロード・アドレス 24 ビット（0x0 ～ 0xffffff）
(4)	コード コードの 1 バイトごとを 2 桁の 16 進数で表したもの

- S3 レコード

コードの値を示します。

```
S3  XX  YYYYYY  ZZ......ZZ  SS  NL
(1) (2)   (3)       (4)
```

項番	説明
(1)	S3 固定
(2)	レコード長
(3)	ロード・アドレス 32 ビット（0x0 ～ 0xffffffff）
(4)	コード コードの 1 バイトごとを 2 桁の 16 進数で表したもの

- S7 レコード

エントリ・ポイント・アドレスを示します。

```
S7  XX  YYYYYY  SS  NL
(1) (2)   (3)
```

項番	説明
(1)	S7 固定
(2)	レコード長
(3)	エントリ・ポイント・アドレス 32 ビット（0x0 ～ 0xffffffff）

- S8 レコード

エントリ・ポイント・アドレスを示します。

```
S8  XX  YYYYYY  SS  NL
(1) (2)   (3)
```

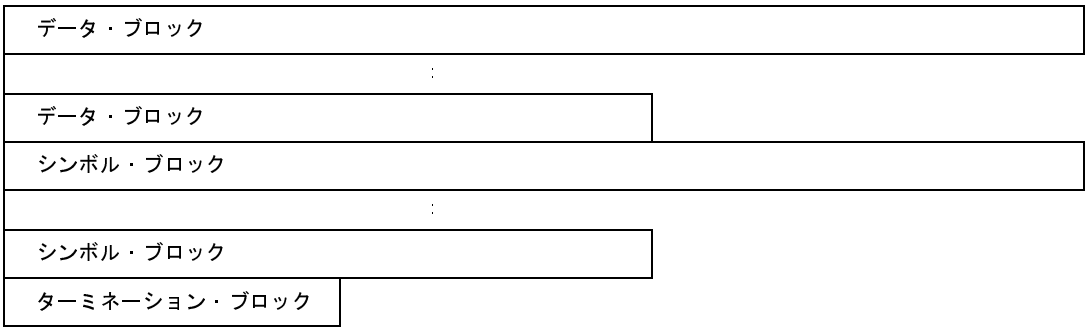
項番	説明
(1)	S8 固定
(2)	レコード長
(3)	エントリ・ポイント・アドレス 24 ビット（0x0 ～ 0xffffff）

3. 4. 3　拡張テクトロニクス・ヘキサ・フォーマット

拡張テクトロニクス・ヘキサ・フォーマットのファイルは，データ・ブロック，シンボル・ブロック，およびターミネーション・ブロックの 3 種類のブロックにより構成されます。

拡張テクトロニクス・ヘキサ・フォーマットのファイル構成を以下に示します。

図 3―3　拡張テクトロニクス・ヘキサ・フォーマットのファイル構成



各ブロックは，各種フィールドにより以下の形で構成されます。

```
%   XX   X  SS  FF[FF......]  NL
(1) (2) (3) (4)      (5)      (6)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長 %，NL を除くブロック内のすべての文字の数です。
(3)	ブロックの種類 6 ： データ・ブロック 3 ： シンボル・ブロック 8 ： ターミネーション・ブロック
(4)	チェック・サム %，SS，NL を除くブロック内のすべての文字に対する値[注]を合計したものを 256 で割った余りの値を，2 桁の 16 進数で表したものです。
(5)	各ブロックにより，仕様が異なります。
(6)	ニュー・ライン（¥ n）

注　各文字に対する値は，以下のように定められています。

0 ～ 9 : 0 ～ 9，A ～ Z : 10 ～ 35，$: 36，% : 37，. : 38，- : 39，a ～ z : 40 ～ 65

- データ・ブロック

コードの値を示します。

```
%   XX  6   SS  XXXX  DD......DD  NL
(1) (2) (3) (4)  (5)      (6)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	6 固定
(4)	チェック・サム
(5)	ロード・アドレスの桁数とロード・アドレス 2 ～ 17 バイト
(6)	コード コードの 1 バイトごとを 2 桁の 16 進数で表したもの

例

```
%   15  6   1C  3 100 020202020202  NL
(1) (2) (3) (4)  (5)       (6)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	ブロックの種類 6
(4)	チェック・サム 1＋5＋6＋3＋1＋0＋0＋0＋2＋0＋2＋0＋2＋0＋2＋0＋2＋0＋2＝28 を 256 で割った余りの値を 2 桁の 16 進数で表したもの
(5)	ロード・アドレスの桁数は 3，ロード・アドレスは 100
(6)	コードの 1 バイトごとを 2 桁の 16 進数で表したもの

- シンボル・ブロック

シンボルの値を示します。

```
%   XX  3   SS  XX...XX XX...XX XX...XX NL
(1) (2) (3) (4)   (5)     (6)     (7)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	3 固定
(4)	チェック・サム
(5)	セクション名の文字数とセクション名 2 ～ 17 バイト
(6)	セクション定義フィールド（SEDF）**注 1** 5 ～ 35 バイト

項番	説明
(7)	シンボル定義フィールド（SYDF）注2 5 ～ 35 バイト

注 1. セクション定義フィールド

セクション定義フィールドはセクションごとに 1 つ存在しなければならず，任意の数のシンボル定義フィールドの前，または後ろに続けることができます。

```
0   XX...XX XX...XX
(1)   (2)     (3)
```

項番	説明
(1)	0 固定 このフィールドがセクション定義フィールドであることを示します。
(2)	セクションのベース・アドレスの桁数とセクションのベース・アドレス 2 ～ 17 バイト
(3)	セクションの長さの桁数とセクションの長さ 2 ～ 17 バイト

2. シンボル定義フィールド

```
T   XX...XX XX...XX
(1)   (2)     (3)
```

項番	説明
(1)	シンボルの種類 1 : グローバル・アドレス（バインディング・クラス GLOBAL と ABS 以外のタイプを持つシンボル） 2 : グローバル・スカラ（バインディング・クラス GLOBAL とタイプ ABS を持つシンボル） 3 : グローバル・コード・アドレス 4 : グローバル・コード・アドレス 5 : ローカル・アドレス（バインディング・クラス LOCAL と ABS 以外のタイプを持つシンボル） 6 : ローカル・スカラ（バインディング・クラス LOCAL とタイプ ABS を持つシンボル） 7 : グローバル・コード・アドレス 8 : グローバル・コード・アドレス
(2)	シンボルの文字数とシンボル 2 ～ 17 バイト
(3)	シンボルの値の桁数とシンボルの値 2 ～ 17 バイト

例 1.

```
%   37  3   60  8SVCSTUFF 02402C6 22CR1D14OPEN25014READ25815WRITE260  NL
(1) (2) (3) (4)    (5)      (6)                  (7)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	ブロックの種類 3
(4)	チェック・サム
(5)	セクション名の文字数は 8，セクション名は SVCSTUFF
(6)	セクション定義フィールド セクションのベース・アドレスの桁数は 2，セクションのベース・アドレスは 40，セクションの長さの桁数は 2，セクションの長さは C6
(7)	シンボル定義フィールド 22CR1D ／ 14OPEN250 ／ 14READ258 ／ 15WRITE260

2.

```
%   37  3   C8  8SVCSTUFF   15CLOSE26814EXIT27029BUFLENGTH28013BUF278 NL
(1) (2) (3) (4)    (5)                          (6)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	ブロックの種類 3
(4)	チェック・サム
(5)	セクション名の文字数は 8，セクション名は SVCSTUFF
(6)	シンボル定義フィールド 15CLOSE268 ／ 14EXIT270 ／ 29BUFLENGTH280 ／ 13BUF278

- ターミネーション・ブロック

エントリ・ポイント・アドレスを示します。

```
%   XX  8   SS  YY......YY  NL
(1) (2) (3) (4)     (5)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	8 固定
(4)	チェック・サム
(5)	エントリ・ポイント・アドレスの桁数とエントリ・ポイント・アドレス 2 ～ 17 バイト

例

```
%   08  8   1A  280   NL
(1) (2) (3) (4) (5)
```

項番	説明
(1)	ヘッダ文字
(2)	ブロック長
(3)	ブロックの種類 8
(4)	チェック・サム
(5)	エントリ・ポイント・アドレスの桁数は 2，エントリ・ポイント・アドレスは 80

付録 A　ウインドウ・リファレンス

ここでは，ビルドに関するウインドウ／パネル／ダイアログについての詳細を説明します。

A. 1　説　　明

以下に，ビルドに関するウインドウ／パネル／ダイアログの一覧を示します。

表 A―1　ウインドウ／パネル／ダイアログ一覧

ウインドウ／パネル／ダイアログ名	機能概要
メイン・ウインドウ	CubeSuite+ を起動した際，最初にオープンするウインドウ
プロジェクト・ツリー パネル	プロジェクトの構成要素をツリー表示
プロパティ パネル	プロジェクト・ツリー パネルで選択しているビルド・ツール・ノード，ファイル，カテゴリ・ノードについて，詳細情報を表示，および設定を変更
エディタ パネル	テキスト・ファイル／ソース・ファイルを表示／編集
出力 パネル	ビルド・ツールから出力するメッセージを表示
ファイル追加 ダイアログ	新規ファイルを作成，およびプロジェクトに追加
フォルダとファイル追加 ダイアログ	既存のファイルとフォルダ構成をプロジェクトに追加
文字列入力 ダイアログ	1 行分の文字列を入力，編集
テキスト編集 ダイアログ	複数行のテキストを入力，編集
パス編集 ダイアログ	パス，またはパスを含むファイル名を編集，追加
システム・インクルード・パス順設定 ダイアログ	コンパイラに対して指定するシステム・インクルード・パスを参照，および指定順を設定
ファイルの保存設定 ダイアログ	エディタ パネルで編集中のファイルのエンコードと改行コードを設定
リンク・ディレクティブ生成 ダイアログ	リンク・ディレクティブ・ファイルを生成
オブジェクト・ファイル指定 ダイアログ	本ダイアログの呼び出し元に設定するオブジェクト・ファイルを選択
セグメント指定 ダイアログ	本ダイアログの呼び出し元に設定するセグメントを選択
リンク順設定 ダイアログ	リンカに入力するファイルを参照，およびリンク順を設定
ビルド・モード設定 ダイアログ	ビルド・モードの追加と削除，および現在のビルド・モードを一括設定
バッチ・ビルド ダイアログ	プロジェクトが持つビルド・モードを一括して，ビルド，リビルド，クリーンを実行
処理中表示 ダイアログ	処理の進捗状況を表示
オプション ダイアログ	各種環境を設定
既存のファイルを追加 ダイアログ	プロジェクトに既存のファイルを追加

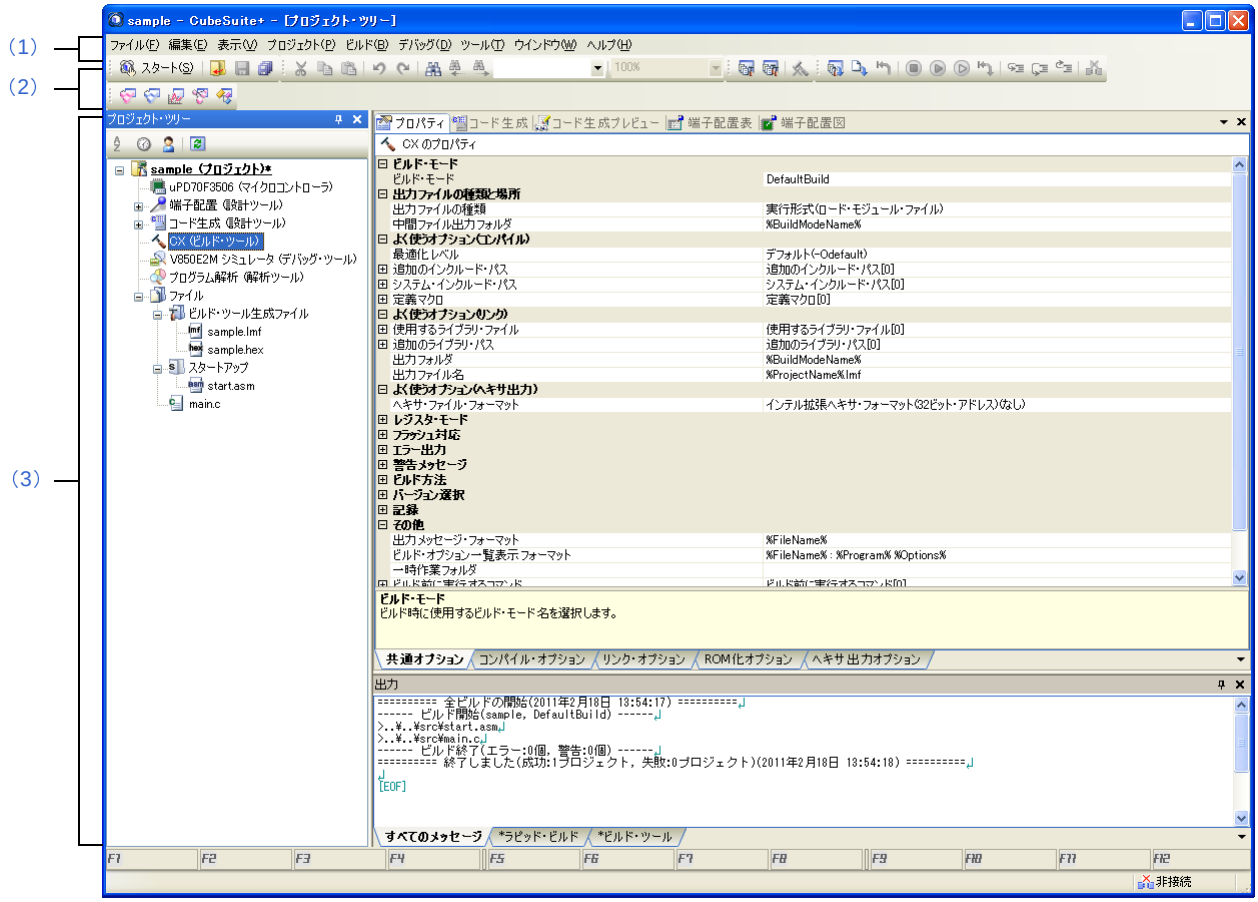
ウインドウ／パネル／ダイアログ名	機能概要
ビルド・オプションのインポート ダイアログ	ビルド・オプションのインポート対象となるプロジェクト・ファイルを選択
フォルダの参照 ダイアログ	本ダイアログの呼び出し元に設定するフォルダを選択
ブート領域用ロード・モジュール・ファイルを指定 ダイアログ	本ダイアログの呼び出し元に設定するブート領域用ロード・モジュール・ファイルを選択
Far Jump ファイルを指定 ダイアログ	本ダイアログの呼び出し元に設定する Far Jump ファイルを選択
ROM 化用領域確保コード・ファイルを指定 ダイアログ	本ダイアログの呼び出し元に設定する ROM 化用領域確保コード・ファイルを選択
名前を付けて保存 ダイアログ	編集中のファイル，または各パネルの内容をファイルに保存
プログラムから開く ダイアログ	ファイルを開くアプリケーションを選択
インポートするファイルを選択 ダイアログ	リンク順設定 ダイアログにインポートするリンク順指定ファイルを選択
エクスポートするファイルを選択 ダイアログ	リンク順指定ファイルを生成
Stack Usage Tracer ウインドウ	スタック見積もりツールを起動した際，最初にオープンするウインドウ
サイズ不明関数・サイズ変更関数一覧 ダイアログ	スタック見積もりツールがスタック情報を取得できていない関数，意図的に情報の変更が行われた関数，およびスタック見積もりツールが強制的に加算サイズの設定を行った関数を一覧表示
スタックサイズ変更 ダイアログ	選択関数に対する情報を変更
ファイルを開く ダイアログ	既存のスタック・サイズ指定ファイルをオープン

メイン・ウインドウ

CubeSuite+ を起動した際，最初にオープンするウインドウです。

ビルドを行う際は，本ウインドウからユーザ・プログラムの実行制御，および各パネルのオープンを行います。

図 A―1　メイン・ウインドウ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］

［オープン方法］

- Windows の［スタート］→［すべてのプログラム］→［Renesas Electronics CubeSuite+］→［CubeSuite+］を選択

［各エリアの説明］

(1) メニューバー

ビルド関連のメニューを示します。

(a)［プロジェクト］

［プロジェクト］メニューでは，プロジェクト関連を操作するメニュー項目を表示します。

新しいプロジェクトを作成 ...	現在のプロジェクトを閉じて，新しいプロジェクトを作成するために，プロジェクト作成 ダイアログをオープンします。 開いているプロジェクト，またはファイルを変更し，保存していない場合は，それらを保存するかどうかの確認を行います。
プロジェクトを開く ...	現在のプロジェクトを閉じて，既存のプロジェクトを開くために，プロジェクトを開く ダイアログをオープンします。 開いているプロジェクト，またはファイルを変更し，保存していない場合は，それらを保存するかどうかの確認を行います。
お気に入りのプロジェクト	お気に入りのプロジェクトを開く，または登録するためのカスケード・メニューを表示します。
1 パス	［お気に入りのプロジェクト］→［1 お気に入りのプロジェクトに登録］で登録したプロジェクトを開きます。 プロジェクトを登録していない場合は，“お気に入りのプロジェクト”を表示します。
2 パス	［お気に入りのプロジェクト］→［2 お気に入りのプロジェクトに登録］で登録したプロジェクトを開きます。 プロジェクトを登録していない場合は，“お気に入りのプロジェクト”を表示します。
3 パス	［お気に入りのプロジェクト］→［3 お気に入りのプロジェクトに登録］で登録したプロジェクトを開きます。 プロジェクトを登録していない場合は，“お気に入りのプロジェクト”を表示します。
4 パス	［お気に入りのプロジェクト］→［4 お気に入りのプロジェクトに登録］で登録したプロジェクトを開きます。 プロジェクトを登録していない場合は，“お気に入りのプロジェクト”を表示します。
1 お気に入りのプロジェクトに登録	現在開いているプロジェクトのパスを［お気に入りのプロジェクト］→［1 パス］に登録します。
2 お気に入りのプロジェクトに登録	現在開いているプロジェクトのパスを［お気に入りのプロジェクト］→［2 パス］に登録します。
3 お気に入りのプロジェクトに登録	現在開いているプロジェクトのパスを［お気に入りのプロジェクト］→［3 パス］に登録します。
4 お気に入りのプロジェクトに登録	現在開いているプロジェクトのパスを［お気に入りのプロジェクト］→［4 パス］に登録します。

追加	プロジェクトにサブプロジェクトを追加するためのカスケード・メニューを表示します。
既存のサブプロジェクトを追加 ...	プロジェクトに既存のサブプロジェクトを追加するために，既存のサブプロジェクトを追加 ダイアログをオープンします。
新しいサブプロジェクトを追加 ...	プロジェクトに新しいサブプロジェクトを追加するために，プロジェクト作成 ダイアログをオープンします。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	ファイル・ノードの直下にカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
*選択しているプロジェクト，またはサブプロジェクト名*をアクティブ・プロジェクトに設定	選択しているプロジェクト，またはサブプロジェクトをアクティブ・プロジェクトに設定します。
プロジェクトを閉じる	現在開いているプロジェクトを閉じます。 開いているプロジェクト，またはファイルを変更し，保存していない場合は，それらを保存するかどうかの確認を行います。
プロジェクトを保存	現在開いているプロジェクトの設定情報をプロジェクト・ファイルに保存します。
名前を付けてプロジェクトを保存 ...	現在開いているプロジェクトの設定情報を別名のプロジェクト・ファイルに保存するために，名前を付けてプロジェクトを保存 ダイアログをオープンします。
プロジェクトから外す	選択しているサブプロジェクト，またはファイルをプロジェクトから外します。 サブプロジェクト・ファイル，およびファイル自体はファイル・システム上から削除しません。
プロジェクトと開発ツールをパックして保存 ...	本製品一式とプロジェクト一式を指定したフォルダにコピーして，１つのフォルダにまとめて保存します。 また，プロジェクトのみをパックして保存することも可能です。

(b) [ビルド]

[ビルド]メニューでは，ビルド関連を操作するメニュー項目を表示します。

ビルド・プロジェクト	プロジェクトのビルドを行います。サブプロジェクトを追加している場合は，サブプロジェクトのビルドも行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。

リビルド・プロジェクト	プロジェクトのリビルドを行います。サブプロジェクトを追加している場合は，サブプロジェクトのリビルドも行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
クリーン・プロジェクト	プロジェクトのクリーンを行います。サブプロジェクトを追加している場合は，サブプロジェクトのクリーンも行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
ラピッド・ビルド	ラピッド・ビルド機能の有効（デフォルト）／無効を選択します（トグル）。
依存関係の更新	プロジェクトのビルド対象ファイルの依存関係を更新します。サブプロジェクトを追加している場合は，サブプロジェクトのビルド対象ファイルの依存関係も更新します。
*アクティブ・プロジェクト*をビルド	アクティブ・プロジェクトのビルドを行います。 アクティブ・プロジェクトがメイン・プロジェクトの場合，サブプロジェクトのビルドは行いません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
*アクティブ・プロジェクト*をリビルド	アクティブ・プロジェクトのリビルドを行います。 アクティブ・プロジェクトがメイン・プロジェクトの場合，サブプロジェクトのリビルドは行いません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
*アクティブ・プロジェクト*をクリーン	アクティブ・プロジェクトのクリーンを行います。 アクティブ・プロジェクトがメイン・プロジェクトの場合，サブプロジェクトのクリーンは行いません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
*アクティブ・プロジェクト*の依存関係の更新	アクティブ・プロジェクトのビルド対象ファイルの依存関係を更新します。
ビルドを中止	実行中のビルド，リビルド，バッチ・ビルド，クリーンを中止します。
ビルド・モードの設定 ...	ビルド・モードの変更，追加等を行うために，ビルド・モード設定 ダイアログをオープンします。
バッチ・ビルド ...	バッチ・ビルドを行うために，バッチ・ビルド ダイアログをオープンします。
ビルド・オプション一覧	現在設定しているビルド・オプションを出力 パネルに一覧表示します。

(2) ツールバー

ビルド関連のボタン群を示します。

(a) ビルド・ツールバー

ビルド・ツールバーでは，ビルド関連を操作するボタン群を表示します。

	プロジェクトのビルドを行います。サブプロジェクトを追加している場合は，サブプロジェクトのビルドも行います。 なお，本ボタンは，ビルド・ツールが実行中の場合は無効となります。

	プロジェクトのリビルドを行います。サブプロジェクトを追加している場合は，サブプロジェクトのリビルドも行います。 なお，本ボタンは，ビルド・ツールが実行中の場合は無効となります。
	実行中のビルド，リビルド，バッチ・ビルド，クリーンを中止します。

（3）パネル表示エリア

以下のパネルを表示するエリアです。

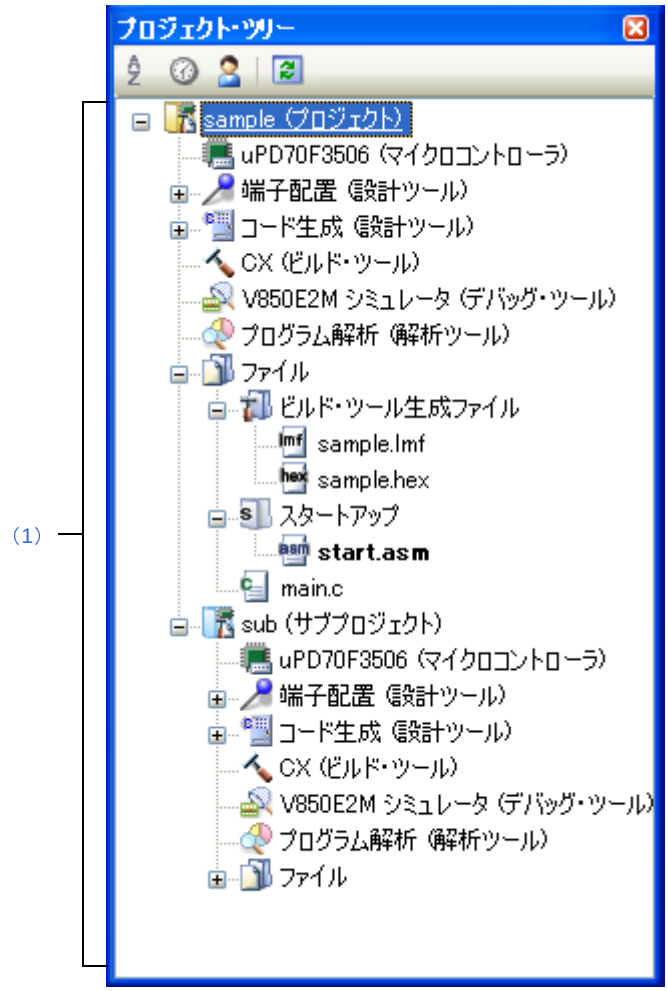
- プロジェクト・ツリー パネル
- プロパティ パネル
- エディタ パネル
- 出力 パネル

表示内容についての詳細は，各パネルの項を参照してください。

プロジェクト・ツリー パネル

プロジェクトを構成するビルド・ツール，ソース・ファイル等の構成要素をツリー表示します。

図 A―2　プロジェクト・ツリー パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［［編集］ メニュー（プロジェクト・ツリー パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］ メニュー→［プロジェクト・ツリー］ を選択

[各エリアの説明]

(1) プロジェクト・ツリー エリア

プロジェクトの構成要素を以下のノードでツリー表示します。

ノード	説明
プロジェクト名 (プロジェクト) (以降, “プロジェクト・ノード”と呼びます。)	プロジェクトの名前です。
ビルド・ツール名 (ビルド・ツール) (以降, “ビルド・ツール・ノード”と呼びます。)	使用するビルド・ツール (CX) です。
ファイル (以降, “ファイル・ノード”と呼びます。)	プロジェクトに追加している以下のファイルを, 直下に表示します。 - C ソース・ファイル (*.c) - アセンブラ・ソース・ファイル (*.asm, *.s) - ヘッダ・ファイル (*.h, *.inc) - オブジェクト・モジュール・ファイル (*.obj, *.o) - ライブラリ・ファイル (*.lib) - リンク・ディレクティブ・ファイル (*.dir, *.dr) - シンボル情報ファイル (*.sfg) - その他のファイル (*.doc, *.xml 等)
ビルド・ツール生成ファイル (以降, “ビルド・ツール生成ファイル・ノード”と呼びます。)	ビルド時に生成するノードで, ビルド・ツールによって生成したファイルのうち, 以下のものを直下に表示します。 - ライブラリ用のプロジェクト以外の場合 ロード・モジュール・ファイル (*.lmf) リンク・マップ・ファイル (*.map) ヘキサ・ファイル (*.hex) エラー・メッセージ・ファイル (*.err) - ライブラリ用のプロジェクトの場合 ライブラリ・ファイル (*.lib) エラー・メッセージ・ファイル (*.err) 本ノードに表示しているファイルは, 名前の変更, 削除, 移動を行うことができません。 なお, 本ノードは常にファイル・ノード以下に生成します。 ビルド後にプロジェクトの再読み込みを行った場合, 本ノードは表示されなくなります。
スタートアップ (以降, “スタートアップ・ノード”と呼びます。)	プロジェクトに標準以外のスタートアップ・ルーチンを追加するためのノードです。 なお, 本ノードは常にファイル・ノード以下に表示します。
カテゴリ名 (以降, “カテゴリ・ノード”と呼びます。)	ファイルを分類するためにユーザが作成するカテゴリです (「2. 3. 6 ファイルをカテゴリに分類する」参照)。 なお, 本ノードは常にファイル・ノード以下に作成します。

ノード	説明
サブプロジェクト名 (サブプロジェクト) （以降，“サブプロジェクト・ノード”と呼びます。）	プロジェクトに追加しているサブプロジェクトです。

各構成要素（ノード，またはファイル）を選択すると，その詳細情報（プロパティ）をプロパティ パネルに表示し，設定の変更を行うことができます。

備考 複数の構成要素を選択している場合は，その構成要素に共通するタブのみ表示します。
なお，複数のファイルを選択し，共通するプロパティの値が異なる場合，その値は空欄となります。

本エリアは，次の機能を備えています。

(a) ファイルの追加

以下のいずれかの方法により，ファイルの追加を行うことができます。
ファイルの追加先はファイル・ノード以下となります。

- 既存のファイルを追加する場合
  - プロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノード，ファイルのいずれかを選択し，［ファイル］メニュー→［追加］→［既存のファイルを追加 ...］を選択する。既存のファイルを追加 ダイアログがオープンし，追加するファイルを選択する。
  - プロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノード，ファイルのいずれかのコンテキスト・メニューの［追加］→［既存のファイルを追加 ...］を選択する。既存のファイルを追加 ダイアログがオープンし，追加するファイルを選択する。
  - エクスプローラなどでファイルをコピーし，本エリアにフォーカスを移動したのち，［編集］メニュー→［貼り付け］を選択する。
  - エクスプローラなどからファイルをドラッグし，本エリア上のファイルを追加したい位置にドロップする。

備考 エクスプローラなどからファイルをドラッグし，プロジェクト・ツリー下部の空白部分にドロップした場合は，メイン・プロジェクト上にドロップしたものとみなします。

- 新しいファイルを追加する場合
  - プロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノードのいずれかを選択し，［ファイル］メニュー→［追加］→［新しいファイルを追加 ...］を選択する。ファイル追加 ダイアログがオープンし，新しく作成するファイルを指定する。
  - プロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノードのいずれかのコンテキスト・メニューの［追加］→［新しいファイルを追加 ...］を選択する。ファイル追加 ダイアログがオープンし，新しく作成するファイルを指定する。

備考 ファイル追加 ダイアログで指定した場所に，空のファイルを作成します。

(b) プロジェクトからファイルを外す

以下のいずれかの方法により，プロジェクトからファイルを外すことができます。

ファイル自体はファイル・システム上から削除しません。

- プロジェクトから外すファイルを選択し，［プロジェクト］メニュー→［プロジェクトから外す］を選択する。
- プロジェクトから外すファイルを選択し，コンテキスト・メニューの［プロジェクトから外す］を選択する。

(c) ファイルの移動

以下の方法により，ファイルの移動を行うことができます。

ファイルの移動先はファイル・ノード以下となります。

- 移動するファイルをドラッグし，移動先でドロップする。

備考 1. メイン・プロジェクト内，またはサブプロジェクト内でドロップした場合，ファイルに設定した個別オプションは保持します。

2. 異なるプロジェクト間，または同一プロジェクトのメイン・プロジェクトかサブプロジェクトにドロップした場合，ファイルは移動ではなく，コピーします。
なお，ファイルに設定した個別オプションは保持しません。

(d) カテゴリの追加

以下のいずれかの方法により，カテゴリ・ノードの追加を行うことができます。

カテゴリ・ノードの追加先はファイル・ノード以下となります。

- ［プロジェクト］メニュー→［新しいカテゴリを追加］を選択する。
- プロジェクト・ノード，サブプロジェクト・ノード，またはファイル・ノードのコンテキスト・メニューの［新しいカテゴリを追加］を選択する。

備考 1. カテゴリ名は，デフォルトで“新しいカテゴリ”となります。

2. すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。

(e) カテゴリの移動

以下の方法により，カテゴリ・ノードの移動を行うことができます。

カテゴリ・ノードの移動先はファイル・ノード以下となります。

- 移動するカテゴリ・ノードをドラッグし，移動先でドロップする。

備考 1. メイン・プロジェクト内，またはサブプロジェクト内でドロップした場合，カテゴリ・ノードに含まれるファイルに設定した個別オプションは保持します。

2. 異なるプロジェクト間，または同一プロジェクトのメイン・プロジェクトかサブプロジェクトにドロップした場合，カテゴリ・ノードは移動ではなく，コピーします。
なお，カテゴリ・ノードに含まれるファイルに設定した個別オプションは保持しません。

(f) フォルダの追加

以下の方法により，エクスプローラなどからフォルダの追加を行うことができます。

フォルダの追加先はファイル・ノード以下となります。

なお，フォルダはカテゴリとして追加します。

- エクスプローラなどからフォルダをドラッグし，移動先でドロップする。フォルダとファイル追加ダイアログがオープンし，フォルダに含まれているファイルのうち追加するファイルの種類と，フォルダの階層を指定する。

注意　フォルダとファイルを同時にドラッグして，本エリアにドロップすることはできません。

(g) サブプロジェクトのビルド順の表示編集

サブプロジェクトは，ビルド順に上から表示します。

そのため，サブプロジェクトの表示位置を変更することで，ビルド順も変更することができます。

なお，プロジェクトのビルドは，すべてのサブプロジェクト，メイン・プロジェクトの順で行います。

(h) 標準ビルド・オプションの設定

プロパティ パネルにおいて，標準ビルド・オプションの設定に変更を加えると，プロパティの値を太字表示します。

以下の方法により，現在設定しているビルド・オプションを標準ビルド・オプションとする（太字表示を解除する）ことができます。

- ビルド・ツール・ノードを選択し，コンテキスト・メニューの［現在のビルド・オプションをプロジェクトの標準に設定する］を選択する。

備考　標準ビルド・オプションの設定は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト）に対して行います。

(i) ファイル，およびカテゴリのソート

以下の方法により，ファイル，およびカテゴリ・ノードをファイル名順，タイムスタンプ順，ユーザ指定順でソートすることができます。

- ツールバーのいずれかのボタンを選択する。

以下に，各ボタンの説明を示します。

なお，デフォルトでは  を選択します。

ボタン	説明
	カテゴリ・ノード，およびファイルを名前順でソートします。  : 昇順  : 降順  : 昇順

ボタン	説明
	カテゴリ・ノード，およびファイルをタイムスタンプ順でソートします。  : 降順  : 昇順  : 降順
	カテゴリ・ノードとファイル（依存関係ファイルは除く）をユーザが指定した順で表示します（デフォルト）。 カテゴリ・ノード，およびファイルをドラッグ・アンド・ドロップすることにより，表示順を任意に変更することができます。

(j) 依存関係ファイルの表示

プロジェクトに追加しているソース・ファイルに依存関係ファイルが存在する場合，その依存関係ファイルをソース・ファイルにぶら下げて表示します。

依存関係ファイルが存在しないソース・ファイル	 main.c
依存関係ファイルが存在するソース・ファイル	 main.c  io.h

なお，依存関係ファイルの表示は，以下のタイミングで更新します。

- プロジェクトを読み込んだのち，初めてビルドを実行したとき
- ツールバーの  をクリックしたとき
- ［ビルド］メニュー→［依存関係の更新］を選択したとき
- ［ビルド］メニュー→［*アクティブ・プロジェクト*の依存関係の更新］を選択したとき

備考 1. 本機能は，オプション ダイアログの［全般 - ビルド／デバッグ］カテゴリの［プロジェクト・ツリーに依存関係ファイルを表示する］がチェック状態の場合のみ有効となります。

2. プロジェクト・ツリーに表示している依存関係ファイルの情報は，プロジェクト・ファイルには保存しません。

(k) 編集中ファイルの表示

プロジェクトに追加しているファイル（依存関係ファイルは除く）をエディタ パネルで編集し，未保存の場合，ファイル名の後ろに“*”を表示します。

ファイルを保存すると，“*”を消去します。

通常のファイル	 main.c
編集後，未保存のファイル	 main.c*

(l) 個別ビルド・オプションを設定しているソース・ファイルの強調表示

プロジェクトの全体オプションとは異なるビルド・オプション（個別コンパイル・オプション，個別アセンブル・オプション）を設定しているソース・ファイルのアイコンは，通常とは異なるアイコンに変更します。

通常のファイル	 main.c
個別ビルド・オプションを設定しているファイル	 main.c

（m）読み取り専用属性ファイルの強調表示

プロジェクトに追加している読み取り専用属性ファイルは，イタリック表示します。

通常のファイル	 main.c
読み取り専用属性ファイル	 *main.c*

（n）存在しないファイルの強調表示

プロジェクトに追加しているファイルで，存在しないファイルは，グレー表示し，アイコンは淡色表示となります。

通常のファイル	 main.c
存在しないファイル	 main.c

（o）ビルド対象ファイルの強調表示

- ビルド（ラピッド・ビルド），リビルド，コンパイル，アセンブル時にエラーを検出したファイルは，以下の例のように強調表示します。

エラーもワーニングも検出しなかったファイル	 main.c
エラーを検出したファイル	 main.c
ワーニングを検出したファイル	 main.c

備考 1. エラーとワーニングの両方を検出したファイルは，赤色表示します。

2. 強調表示は，ビルド・オプション（全体オプション，または個別オプション）の変更，およびビルド・モードの変更により解除します。

- 以下のファイルは，太字表示します。
  - 編集後，コンパイルしていないソース・ファイル
  - クリーンを実行した場合のソース・ファイル
  - ビルド・ツールのオプションを変更した場合のソース・ファイル
  - ビルド・モードを変更した場合のソース・ファイル

備考 プロジェクトを開いた直後は，すべて太字表示となり，ビルドを行うことで太字表示を解除します。

（p）ビルド対象外ファイルの強調表示

ビルドの対象外に設定しているファイルは，以下の例のように強調表示します。

通常のファイル	main.c
ビルド対象外ファイル	main.c

（q）コア番号指定ファイルの強調表示

マルチコア対応デバイスを使用するプロジェクトの場合，コア番号を指定しているファイルのアイコンは，コア番号を付加したものに変更します。

通常のファイル	main.c
コア番号１を指定しているファイル	main.c

備考 1. コア番号は，１から４までに対応しています。

2. ビルド対象外のファイルは，本機能の対象外となります。

（r）オーバーレイ・アイコンの強調表示

プロジェクト，およびプロジェクトに追加しているファイル，カテゴリ（フォルダへのショートカットを設定している場合のみ）に設定されている Windows エクスプローラのオーバーレイ・アイコンは，以下の例のように通常のアイコンの左側に表示します。

通常のプロジェクト	sample (プロジェクト)
オーバーレイ・アイコンを表示したプロジェクト	sample (プロジェクト)

注意 上記のオーバーレイ・アイコンは，サンプルとして掲載しています。
お使いのツールによって表示されるアイコンは異なりますので，ご注意ください。

なお，オーバーレイ・アイコンの表示は，以下のタイミングで更新します。

- プロジェクトを読み込んだとき
- ツールバーの をクリックしたとき
- ［編集］メニュー→［最新の情報に更新］を選択したとき

備考 本機能は，オプション ダイアログの［全般 - 表示］カテゴリの［プロジェクト・ツリーに Windows エクスプローラのオーバーレイ・アイコンを表示する］がチェック状態で本製品を起動した場合のみ有効となります。

（s）フォルダへのショートカットを設定しているカテゴリの強調表示

フォルダへのショートカットを設定しているカテゴリは，以下の例のように強調表示します。

通常のカテゴリ	source
フォルダへのショートカットを設定しているカテゴリ	source

(t) 変更したプロジェクトの強調表示

プロジェクトに追加しているファイル構成を変更した場合，およびプロジェクトの構成要素のプロパティを変更した場合，プロジェクト名に“*”を付加し，太字表示します。

強調表示は，プロジェクトを保存すると解除します。

通常のプロジェクト	sample (プロジェクト)
変更したプロジェクト	**sample (プロジェクト)***

(u) アクティブ・プロジェクトの強調表示

アクティブ・プロジェクトには，下線を付加します。

通常のプロジェクト	sample (プロジェクト)
アクティブ・プロジェクト	**sample (プロジェクト)***

(v) ファイルの強調表示の状態を更新

以下の方法により，ファイル，読み取り専用属性ファイル，存在しないファイル，およびオーバーレイ・アイコンの強調表示の状態を最新の情報に更新することができます。

- ツールバーの  を選択する。

(w) エディタの起動

特定の拡張子を持つファイルをエディタ パネルでオープンします。オプション ダイアログで，外部テキスト・エディタを使用する設定をしている場合は，設定している外部テキスト・エディタでオープンします。それ以外のファイルは，ホスト OS で関連付けられているアプリケーションで起動します。

注意　ホスト OS で関連付けられていない拡張子のファイルは表示しません。

以下のいずれかの方法により，エディタをオープンすることができます。

- ファイルをダブルクリックする。
- ファイルを選択し，コンテキスト・メニューの［開く］を選択する。
- ファイルを選択し，［Enter］キーを押下する。

以下に，エディタ パネルでオープンできるファイルを示します。

- C ソース・ファイル（*.c）
- アセンブラ・ソース・ファイル（*.asm，*.s）
- ヘッダ・ファイル（*.h，*.inc）
- シンボル情報ファイル（*.sfg）
- リンク・ディレクティブ・ファイル（*.dir，*.dr）
- リンク順指定ファイル（*.mtls）
- リンク・マップ・ファイル（*.map）
- ヘキサ・ファイル（*.hex）
- テキスト・ファイル（*.txt）

備考　以下のいずれかの方法により，上記以外のファイルもエディタ パネルでオープンすることができます。

- ファイルをドラッグし，エディタ パネルにドロップする。
- ファイルを選択し，コンテキスト・メニューの［内部エディタで開く ...］を選択する。

[[編集］メニュー（プロジェクト・ツリー パネル専用部分)]

コピー	選択しているファイル，カテゴリ・ノードをクリップ・ボードにコピーします。 ファイル名，カテゴリ名を編集中の場合は，選択している文字列をクリップ・ボードにコピーします。 なお，本メニューは，ファイル（依存関係ファイルは除く），カテゴリ・ノードを選択している場合のみ有効となります。
貼り付け	クリップ・ボードの内容をプロジェクト・ツリー上で選択しているノードの直下に挿入します。 ファイル名，カテゴリ名を編集中の場合は，クリップ・ボードの内容を挿入します。 なお，本メニューは，クリップボートの内容が同一プロジェクトに存在する場合，ファイル，カテゴリ・ノードを複数選択している場合，およびビルド・ツールが実行中の場合は無効となります。
名前の変更	選択しているプロジェクト，サブプロジェクト，ファイル，カテゴリ・ノードの名前が編集可能な状態になります。［Enter］キーの押下により編集を確定し，［ESC］キーの押下により編集をキャンセルすることができます。 ファイルを選択している場合は，実際のファイル名も変更します。 ファイルを選択し，そのファイルをほかのプロジェクトにも追加している場合は，それらの名前も変更します。 なお，本メニューは，プロジェクト，サブプロジェクト，ファイル（依存関係ファイルは除く），カテゴリ・ノードを選択している場合のみ有効となります。ただし，ビルド・ツールが実行中の場合は無効となります。

[コンテキスト・メニュー]

(1) プロジェクト・ノードを選択している場合

アクティブ・プロジェクトをビルド	アクティブ・プロジェクトのビルドを行います。 アクティブ・プロジェクトがメイン・プロジェクトの場合，サブプロジェクトのビルドは行いません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
アクティブ・プロジェクトをリビルド	アクティブ・プロジェクトのリビルドを行います。 アクティブ・プロジェクトがメイン・プロジェクトの場合，サブプロジェクトのリビルドは行いません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
アクティブ・プロジェクトをクリーン	アクティブ・プロジェクトのクリーンを行います。 アクティブ・プロジェクトがメイン・プロジェクトの場合，サブプロジェクトのクリーンは行いません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
エクスプローラでフォルダを開く	選択しているプロジェクトのプロジェクト・ファイルが存在しているフォルダをエクスプローラでオープンします。
Windows エクスプローラのメニュー	選択しているプロジェクトのプロジェクト・ファイルに対応する Windows エクスプローラでのコンテキスト・メニューを表示します。
追加	プロジェクトにサブプロジェクト，ファイルを追加するためのカスケード・メニューを表示します。
既存のサブプロジェクトを追加 ...	既存のサブプロジェクトを追加 ダイアログをオープンし，選択したサブプロジェクトをプロジェクトに追加します。
新しいサブプロジェクトを追加 ...	プロジェクト作成 ダイアログをオープンし，作成したサブプロジェクトをプロジェクトに追加します。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	ファイル・ノードの直下にカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，200 文字まで指定可能です。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合，およびカテゴリのネスト数が 20 の場合は無効となります。
選択しているプロジェクトをアクティブ・プロジェクトに設定	選択しているプロジェクトをアクティブ・プロジェクトに設定します。

プロジェクトと開発ツールをパックして保存 ...	本製品一式とプロジェクト一式を指定したフォルダにコピーして，1 つのフォルダにまとめて保存します。 また，プロジェクトのみをパックして保存することも可能です。
貼り付け	本メニューは常に無効です。
名前の変更	選択しているプロジェクトの名前が編集可能な状態になります。
プロパティ	選択しているプロジェクトのプロパティをプロパティ パネルに表示します。

(2) サブプロジェクト・ノードを選択している場合

*アクティブ・プロジェクト*をビルド	アクティブ・プロジェクトのビルドを行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
*アクティブ・プロジェクト*をリビルド	アクティブ・プロジェクトのリビルドを行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
*アクティブ・プロジェクト*をクリーン	アクティブ・プロジェクトのクリーンを行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
エクスプローラでフォルダを開く	選択しているサブプロジェクトのサブプロジェクト・ファイルが存在しているフォルダをエクスプローラでオープンします。
Windows エクスプローラのメニュー	選択しているサブプロジェクトのサブプロジェクト・ファイルに対応するWindows エクスプローラでのコンテキスト・メニューを表示します。
追加	プロジェクトにサブプロジェクト，ファイル，カテゴリ・ノードを追加するためのカスケード・メニューを表示します。
既存のサブプロジェクトを追加 ...	既存のサブプロジェクトを追加 ダイアログをオープンし，選択したサブプロジェクトをプロジェクトに追加します。 サブプロジェクトにサブプロジェクトを追加することはできません。
新しいサブプロジェクトを追加 ...	プロジェクト作成 ダイアログをオープンし，作成したサブプロジェクトをプロジェクトに追加します。 サブプロジェクトにサブプロジェクトを追加することはできません。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	ファイル・ノードの直下にカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，200 文字まで指定可能です。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合，およびカテゴリのネスト数が 20 の場合は無効となります。
*選択しているサブプロジェクト*をアクティブ・プロジェクトに設定	選択しているサブプロジェクトをアクティブ・プロジェクトに設定します。

プロジェクトから外す	選択しているサブプロジェクトをプロジェクトから外します。 サブプロジェクト・ファイル自体はファイル・システム上から削除しません。 選択しているサブプロジェクトがアクティブ・プロジェクトの場合は，プロジェクトから外すことはできません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
貼り付け	本メニューは常に無効です。
名前の変更	選択しているサブプロジェクトの名前が編集可能な状態になります。
プロパティ	選択しているサブプロジェクトのプロパティをプロパティ パネルに表示します。

(3) ビルド・ツール・ノードを選択している場合

ビルド・プロジェクト	選択しているプロジェクト（メイン・プロジェクト，またはサブプロジェクト）のビルドを行います。サブプロジェクトを追加している場合は，サブプロジェクトのビルドも行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
リビルド・プロジェクト	選択しているプロジェクト（メイン・プロジェクト，またはサブプロジェクト）のリビルドを行います。サブプロジェクトを追加している場合は，サブプロジェクトのリビルドも行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
クリーン・プロジェクト	選択しているプロジェクト（メイン・プロジェクト，またはサブプロジェクト）のクリーンを行います。サブプロジェクトを追加している場合は，サブプロジェクトのクリーンも行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
現在のビルド・オプションを*選択しているプロジェクト*の標準に設定する	現在のビルド・オプションを選択しているプロジェクトの標準に設定します。サブプロジェクトを追加している場合，サブプロジェクトの設定は行いません。 標準と異なるビルド・オプションを設定した場合，そのプロパティは太字表示します。
ビルド・オプションのインポート ...	ビルド・オプションのインポート ダイアログをオープンし，選択したプロジェクト・ファイルからビルド・オプションをインポートします。**注**
リンク順を設定する ...	リンク順設定 ダイアログをオープンし，オブジェクト・モジュール・ファイル，ライブラリ・ファイルの表示，およびリンク順の設定を行います。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
リンク・ディレクティブ・ファイルを生成する ...	リンク・ディレクティブ生成 ダイアログをオープンし，リンク・ディレクティブ・ファイルの生成を行います。
プロパティ	選択しているビルド・ツールのプロパティをプロパティ パネルに表示します。

注 ビルド・オプションのインポート機能についての詳細は，「2. 15. 1 他のプロジェクトのビルド・オプションをインポートする」を参照してください。

(4) ファイル・ノードを選択している場合

追加	プロジェクトにファイル，カテゴリ・ノードを追加するためのカスケード・メニューを表示します。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	本ノードの直下にカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，200 文字まで指定可能です。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合，およびカテゴリのネスト数が 20 の場合は無効となります。
エクスプローラでフォルダを開く	本メニューは常に無効です。
Windows エクスプローラのメニュー	本メニューは常に無効です。
プロジェクトから外す	本メニューは常に無効です。
コピー	本メニューは常に無効です。
貼り付け	クリップ・ボードの内容を本ノードの直下に挿入します。 ただし，クリップボートの内容が同一プロジェクトに存在する場合は，無効となります。
名前の変更	本メニューは常に無効です。
プロパティ	本ノードのプロパティをプロパティ パネルに表示します。

(5) ファイルを選択している場合

コンパイル	選択している C ソース・ファイルをコンパイルします。 なお，本メニューは，C ソース・ファイル（ビルド対象外のファイルを除く）を選択している場合のみ表示します。 ただし，ビルド・ツールが実行中の場合は無効となります。
アセンブル	選択しているアセンブラ・ソース・ファイルをアセンブルします。 なお，本メニューは，アセンブラ・ソース・ファイル（ビルド対象外のファイルを除く）を選択している場合のみ表示します。 ただし，ビルド・ツールが実行中の場合は無効となります。
開く	ファイルの拡張子に割り当てられたアプリケーションで選択しているファイルをオープンします（「(w) エディタの起動」参照）。 なお，本メニューは，複数のファイルを選択している場合は無効となります。

内部エディタで開く ...	エディタ パネルで選択しているファイルをオープンします。 なお，本メニューは，複数のファイルを選択している場合は無効となります。
アプリケーションを指定して開く ...	プログラムから開く ダイアログをオープンし，指定したアプリケーションで選択しているファイルをオープンします。 ただし，複数のファイルを選択している場合は無効となります。
エクスプローラでフォルダを開く	選択しているファイルが存在しているフォルダをエクスプローラでオープンします。
Windows エクスプローラのメニュー	選択しているファイルに対応する Windows エクスプローラでのコンテキスト・メニューを表示します。
追加	プロジェクトにファイル，カテゴリ・ノードを追加するためのカスケード・メニューを表示します。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。追加先は，選択しているファイルと同じレベルです。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。追加先は，選択しているファイルと同じレベルです。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	選択しているファイルと同じレベルにカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，200 文字まで指定可能です。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合，およびカテゴリのネスト数が 20 の場合は無効となります。
プロジェクトから外す	選択しているファイルをプロジェクトから外します。 ファイル自体はファイル・システム上から削除しません。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
コピー	選択しているファイルをクリップ・ボードにコピーします。 ファイル名を編集中の場合は，選択している文字列をクリップ・ボードにコピーします。
貼り付け	本メニューは常に無効です。
名前の変更	選択しているファイルの名前が編集可能な状態になります。 実際のファイル名も変更します。 選択しているファイルをほかのプロジェクトにも追加している場合は，それらの名前も変更します。
プロパティ	選択しているファイルのプロパティをプロパティ パネルに表示します。

(6) ビルド・ツール生成ファイル・ノードを選択している場合

プロパティ	本ノードのプロパティをプロパティ パネルに表示します。

(7) スタートアップ・ノードを選択している場合

追加	プロジェクトにファイル，カテゴリ・ノードを追加するためのカスケード・メニューを表示します。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	本ノードの直下にカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，200 文字まで指定可能です。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合，およびカテゴリのネスト数が 20 の場合は無効となります。
エクスプローラでフォルダを開く	本メニューは常に無効です。
Windows エクスプローラのメニュー	本メニューは常に無効です。
プロジェクトから外す	本メニューは常に無効です。
コピー	本メニューは常に無効です。
貼り付け	クリップ・ボードの内容を本ノードの直下に挿入します。 ただし，クリップボートの内容が同一プロジェクトに存在する場合は，無効となります。
名前の変更	本メニューは常に無効です。
プロパティ	本ノードのプロパティをプロパティ パネルに表示します。

(8) カテゴリ・ノードを選択している場合

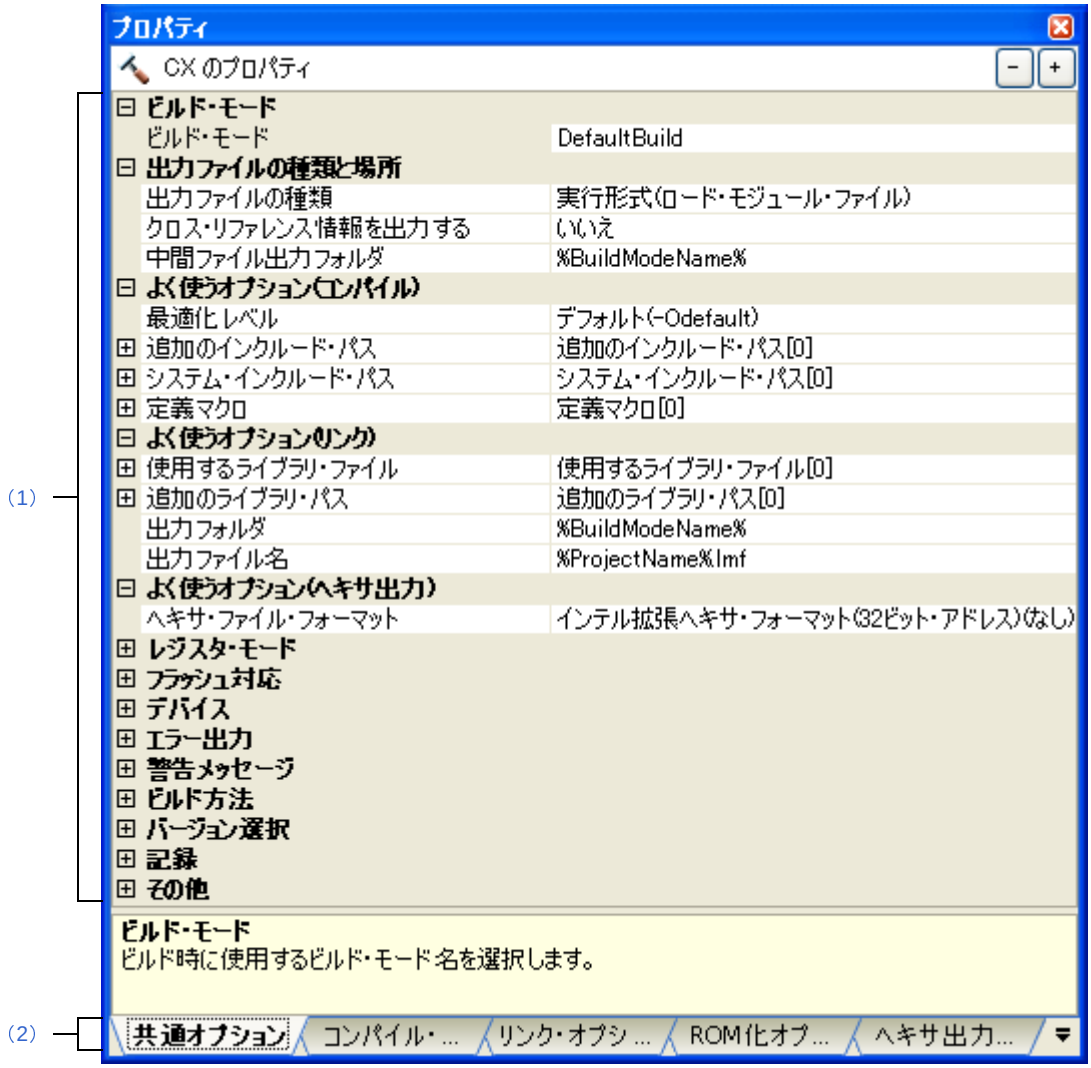
追加	プロジェクトにファイル，カテゴリ・ノードを追加するためのカスケード・メニューを表示します。

既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンします。
新しいカテゴリを追加	本ノードの直下にカテゴリ・ノードを追加し，カテゴリ名が編集可能な状態になります。 カテゴリ名は，200 文字まで指定可能です。 カテゴリ名は，デフォルトで“新しいカテゴリ”となります。すでに存在するカテゴリ・ノードと同名のカテゴリ・ノードを追加することもできます。 なお，本メニューは，ビルド・ツールが実行中の場合，およびカテゴリのネスト数が 20 の場合は無効となります。
エクスプローラでフォルダを開く	選択しているカテゴリに設定しているフォルダへのショートカット先をエクスプローラでオープンします。 なお，本メニューは，フォルダへのショートカットを設定していない場合は無効となります。
Windows エクスプローラのメニュー	選択しているカテゴリに設定しているフォルダへのショートカット先に対応する Windows エクスプローラでのコンテキスト・メニューを表示します。 なお，本メニューは，フォルダへのショートカットを設定していない場合は無効となります。
プロジェクトから外す	選択しているカテゴリ・ノードをプロジェクトから外します。 なお，本メニューは，ビルド・ツールが実行中の場合は無効となります。
コピー	選択しているカテゴリ・ノードをクリップ・ボードにコピーします。 カテゴリ名を編集中の場合は，選択している文字列をクリップ・ボードにコピーします。
貼り付け	クリップ・ボードの内容を本ノードの直下に挿入します。 ただし，クリップボートの内容が同一プロジェクトに存在する場合は，無効となります。 カテゴリ名を編集中の場合は，クリップ・ボードの内容を挿入します。
名前の変更	選択しているカテゴリ・ノードの名前が編集可能な状態になります。
プロパティ	選択しているカテゴリ・ノードのプロパティをプロパティ パネルに表示します。

プロパティ パネル

プロジェクト・ツリー パネルで選択しているビルド・ツール・ノード，ファイル，カテゴリ・ノードについて，カテゴリ別に詳細情報の表示，および設定の変更を行います。

図 A―3 プロパティ パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［［編集］ メニュー（プロパティ パネル専用部分）］
- ［コンテキスト・メニュー］

[オープン方法]

- プロジェクト・ツリー パネルにおいて，ビルド・ツール・ノード，ファイル，カテゴリ・ノードを選択したのち，［表示］メニュー→［プロパティ］を選択，またはコンテキスト・メニュー→［プロパティ］を選択

備考 すでにプロパティ パネルがオープンしている場合，プロジェクト・ツリー パネルにおいて，ビルド・ツール・ノード，ファイル，カテゴリ・ノードを選択することで，選択した項目の詳細情報を表示します。

[各エリアの説明]

(1) 詳細情報表示／変更エリア

プロジェクト・ツリー パネルで選択しているビルド・ツール・ノード，ファイル，カテゴリ・ノードの詳細情報を，カテゴリ別のリスト形式で表示し，設定の変更を直接行うことができるエリアです。

⊟マークは，そのカテゴリ内に含まれているすべての項目を展開表示していることを示し，また，⊞マークは，カテゴリ内の項目を折りたたみ表示していることを示します。展開／折りたたみ表示の切り替えは，このマークのクリック，またはカテゴリ名のダブルクリックにより行うことができます。

HEXマークは，そのプロパティのテキスト・ボックスが 16 進数入力専用であることを示します。

カテゴリ，およびそれに含まれる項目の表示内容／設定方法についての詳細は，該当するタブの項を参照してください。

(2) タブ選択エリア

タブを選択することにより，詳細情報を表示するカテゴリが切り替わります。

本パネルには，次のタブが存在します（各タブ上における表示内容／設定方法についての詳細は，該当するタブの項を参照してください）。

(a) プロジェクト・ツリー パネルでビルド・ツール・ノードを選択している場合

- ［共通オプション］タブ
- ［コンパイル・オプション］タブ
- ［アセンブル・オプション］タブ
- ［リンク・オプション］タブ
- ［ROM 化オプション］タブ
- ［ヘキサ出力オプション］タブ
- ［ライブラリ生成オプション］タブ

(b) プロジェクト・ツリー パネルでファイルを選択している場合

- ［ビルド設定］タブ（C ソース・ファイル，アセンブラ・ソース・ファイル，オブジェクト・モジュール・ファイル，リンク・ディレクティブ・ファイル，シンボル情報ファイル，ライブラリ・ファイルの場合）
- ［個別コンパイル・オプション］タブ（C ソース・ファイルの場合）
- ［個別アセンブル・オプション］タブ（アセンブラ・ソース・ファイルの場合）
- ［ファイル情報］タブ

(c) プロジェクト・ツリー パネルでカテゴリ・ノード，ファイル・ノード，ビルド・ツール生成ファイル・ノード，スタートアップ・ノードを選択している場合

- ［カテゴリ情報］タブ

備考　プロジェクト・ツリー パネルで複数の構成要素を選択している場合は，その構成要素に共通するタブのみ表示します。
プロパティの値の変更は，選択している複数の構成要素に共通に反映します。

［[編集］メニュー（プロパティ パネル専用部分）］

元に戻す	直前に行ったプロパティの値の編集作業を取り消します。
切り取り	プロパティの値を編集中の場合，選択している文字列を切り取ってクリップ・ボードに移動します。
コピー	選択しているプロパティの値文字列をクリップ・ボードにコピーします。
貼り付け	プロパティの値を編集中の場合，クリップ・ボードの内容を挿入します。
削除	プロパティの値を編集中の場合，選択している文字列を削除します。
すべて選択	プロパティの値を編集中の場合，選択しているプロパティの値文字列をすべて選択します。

［コンテキスト・メニュー］

元に戻す	直前に行ったプロパティの値の編集作業を取り消します。
切り取り	プロパティの値を編集中の場合，選択している文字列を切り取ってクリップ・ボードに移動します。
コピー	選択しているプロパティの値文字列をクリップ・ボードにコピーします。
貼り付け	プロパティの値を編集中の場合，クリップ・ボードの内容を挿入します。
削除	プロパティの値を編集中の場合，選択している文字列を削除します。
すべて選択	プロパティの値を編集中の場合，選択しているプロパティの値文字列をすべて選択します。
デフォルトに戻す	選択している項目の設定値をプロジェクトに設定しているデフォルト値に戻します。 ただし，［個別コンパイル・オプション］タブ，［個別アセンブル・オプション］タブにおいては，全体オプションの設定値に戻します。
すべてデフォルトに戻す	現在表示しているタブの設定値をすべてプロジェクトに設定しているデフォルト値に戻します。 ただし，［個別コンパイル・オプション］タブ，［個別アセンブル・オプション］タブにおいては，全体オプションの設定値に戻します。

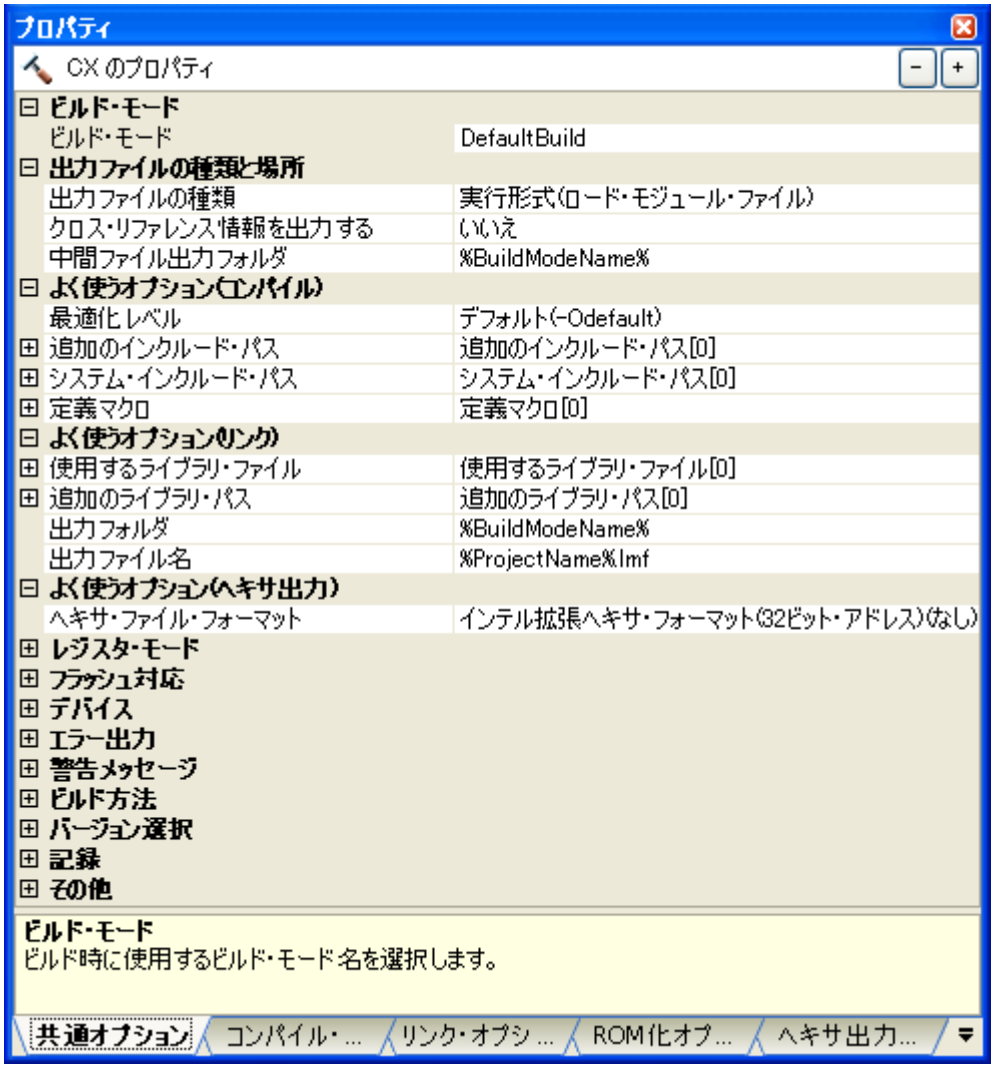
［共通オプション］タブ

本タブでは，ビルド・ツールに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

備考　［よく使うオプション］カテゴリのプロパティを変更した場合，それらに対応するタブの同名のプロパティの値も連動して変更します。

［共通オプション］タブのカテゴリ	対応するタブ
［よく使うオプション（コンパイル）］カテゴリ	［コンパイル・オプション］タブ
［よく使うオプション（アセンブル）］カテゴリ	［アセンブル・オプション］タブ
［よく使うオプション（リンク）］カテゴリ	［リンク・オプション］タブ
［よく使うオプション（ヘキサ出力）］カテゴリ	［ヘキサ出力オプション］タブ

図 A―4　プロパティ パネル：［共通オプション］タブ



［各カテゴリの説明］

(1) ［ビルド・モード］

ビルド・モードに関する詳細情報の表示，および設定の変更を行います。

ビルド・モード	ビルド時に使用するビルド・モードを選択します。 なお，本プロパティには，コンテキスト・メニュー→［すべてデフォルトに戻す］は適用されません。		
	デフォルト	DefaultBuild	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	DefaultBuild	プロジェクトの新規作成時にデフォルトで設定するビルド・モードでビルドを行います。
		プロジェクトに登録しているビルド・モード	プロジェクトに登録しているビルド・モード（DefaultBuild 以外）でビルドを行います。

(2)［出力ファイルの種類と場所］

出力ファイルの種類と場所に関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td rowspan="6">出力ファイルの種類</td><td colspan="3">ビルド時に生成するファイルの種類を選択します。
ここで設定したファイルの種類がデバッグ対象となります。
なお，ライブラリ用のプロジェクト以外の場合は［実行形式（ロード・モジュール・ファイル）］，［実行形式（ROM 化処理前ロード・モジュール・ファイル）］，［実行形式（ヘキサ・ファイル）］のみを表示します。
ライブラリ用のプロジェクトの場合は［ライブラリ形式］のみを表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">- ライブラリ用のプロジェクト以外の場合
実行形式（ロード・モジュール・ファイル）
- ライブラリ用のプロジェクトの場合
ライブラリ形式</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="4">指定可能値</td><td>実行形式（ロード・モジュール・ファイル）</td><td>ビルド時にロード・モジュール・ファイル，およびヘキサ・ファイルを生成します。
ロード・モジュール・ファイルがデバッグ対象となります。</td></tr>
<tr><td>実行形式（ROM 化処理前ロード・モジュール・ファイル）</td><td>ビルド時にロード・モジュール・ファイル，およびヘキサ・ファイルを生成します。
ROM 化処理を行う場合，ROM 化処理前のロード・モジュール・ファイルがデバッグ対象となります。
なお，本項目は，［ROM 化オプション］タブの［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］，および［ROM 化処理前のロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合のみ表示します。</td></tr>
<tr><td>実行形式（ヘキサ・ファイル）</td><td>ビルド時にロード・モジュール・ファイル，およびヘキサ・ファイルを生成します。
ヘキサ・ファイルがデバッグ対象となります。</td></tr>
<tr><td>ライブラリ形式</td><td>ビルド時にライブラリ・ファイルを生成します。</td></tr>
</table>

<table>
<tr><td rowspan="6">デバイス共通オブジェクト・モジュールを出力する</td><td colspan="3">デバイス共通のオブジェクト・モジュール・ファイルを出力するかどうかを選択します。
cx コマンドの -Xcommon オプションに相当します。
なお，本プロパティは，ライブラリ用のプロジェクトの場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ (デバイス固有)(なし)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="4">指定可能値</td><td>はい (V850E/ES コア共通)(-Xcommon=v850e)</td><td>命令セット・アーキテクチャが V850E より上位（V850E，V850E2，V850E2V3）の品種に対応したオブジェクト・モジュール・ファイルを出力します。
なお，本項目は，命令セット・アーキテクチャが V850E2V3 以外の V850 ファミリを使用するプロジェクトの場合のみ表示します。</td></tr>
<tr><td>はい (V850E2 コア共通)(-Xcommon=v850e2)</td><td>命令セット・アーキテクチャが V850E2 より上位（V850E2，および V850E2V3）の品種に対応したオブジェクト・モジュール・ファイルを出力します。
なお，本項目は，命令セット・アーキテクチャが V850E2V3 以外の V850 ファミリを使用するプロジェクトの場合のみ表示します。</td></tr>
<tr><td>はい (V850E2V3 アーキテクチャ共通)(-Xcommon=v850e2v3)</td><td>命令セット・アーキテクチャが V850E2V3 である品種に対応したオブジェクト・モジュール・ファイルを出力します。</td></tr>
<tr><td>いいえ (デバイス固有)(なし)</td><td>指定したデバイス固有の情報を持つオブジェクト・モジュール・ファイルを出力します。</td></tr>
<tr><td rowspan="5">クロス・リファレンス情報を出力する</td><td colspan="3">クロス・リファレンス情報を出力するかどうかを選択します。
なお，［リンク・オプション］タブの［シンボル情報］カテゴリの［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択した場合，本プロパティは［はい (-Xcref)］となります。
また，解析ツールの［強制的にクロス・リファレンス・ファイルを出力する］プロパティで［はい］を選択した場合，本プロパティは［はい (-Xcref)］となります。［強制的にクロス・リファレンス・ファイルを出力する］プロパティが［はい］の場合に，本プロパティを［いいえ］に変更した場合は，ビルド時に［はい (-Xcref)］に変更されます。
cx コマンドの -Xcref オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xcref)</td><td>クロス・リファレンス情報を出力します。
ビルド処理速度は低下しますが，関数ジャンプ等の機能を使用することができます。</td></tr>
<tr><td>いいえ</td><td>クロス・リファレンス情報を出力しません。</td></tr>
</table>

中間ファイル出力フォルダ	中間ファイルを出力するフォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xobj_path オプションに相当します。	
	デフォルト	%BuildModeName%
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集
	指定可能値	247 文字までの文字列

(3)［よく使うオプション（コンパイル）］

コンパイル時によく使うオプションに関する詳細情報の表示，および設定の変更を行います。

最適化レベル	コンパイルの最適化レベルを選択します。 cx コマンドの -O オプションに相当します。		
	デフォルト	デフォルト (-Odefault)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	デフォルト (-Odefault)	デバッグに影響しない範囲の最適化（式の最適化，およびレジスタ割り付けなど）を行います。
		サイズ優先 (-Osize)	オブジェクト・サイズ優先の最適化を行います。 ROM/RAM 容量の削減を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
		実行速度優先 (-Ospeed)	実行速度優先の最適化を行います。 実行速度の短縮を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
		デバッグ優先 (-Onothing)	デバッグを優先して最適化を行います。 デバッグのしやすさを重視し，デフォルトで実行する最適化を含むすべての最適化を抑止します。
		詳細指定	［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリにプロパティを追加表示し，そこで選択した最適化項目を優先して最適化を行います。

<table>
<tr><td>追加のインクルード・パス</td><td>コンパイル時の追加のインクルード・パスを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
指定したインクルード・パスは，CX の標準インクルード・ファイル・フォルダよりも優先して検索します。
パスはプロジェクト・フォルダを基点とします。
本プロパティを省略した場合は，CX の標準インクルード・ファイル・フォルダのみ検索します。
cx コマンドの -I オプションに相当します。
指定したインクルード・パスはサブプロパティとして表示します。
なお，プロジェクト・ツリーにインクルード・ファイルを追加すると，インクルード・パスをサブプロパティの一番最初に追加します。
インクルード・パスに大文字，小文字の区別はありません。
<table><tr><td>デフォルト</td><td>追加のインクルード・パス [定義数]</td></tr><tr><td>変更方法</td><td>[...] ボタンをクリックし，パス編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr><tr><td>指定可能値</td><td>259 文字までの文字列
256 個まで指定可能です。</td></tr></table></td></tr>
</table>

<table>
<tr><td>システム・インクルード・パス</td><td colspan="2">コンパイル時にシステムが設定するインクルード・パスの指定順を変更します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
システム・インクルード・パスは，追加のインクルード・パスより低い優先度で検索します。
パスはプロジェクト・フォルダを基点とします。
cx コマンドの -I オプションに相当します。
インクルード・パスはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>システム・インクルード・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，システム・インクルード・パス順設定 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>変更不可（インクルード・パスの設定順の変更のみ可能）</td></tr>
<tr><td>定義マクロ</td><td colspan="2">定義したいマクロ名を指定します。
「マクロ名 = 定義値」の形式で 1 行に 1 つずつ指定します。
「= 定義値」の部分は省略可能で，省略した場合，定義値を 1 とします。
cx コマンドの -D オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>定義マクロ [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>256 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

(4) [よく使うオプション（アセンブル）]

アセンブル時によく使うオプションに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。

<table>
<tr><td>追加のインクルード・パス</td><td>アセンブル時の追加のインクルード・パスを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
指定したインクルード・パスは，CX の標準インクルード・ファイル・フォルダよりも優先して検索します。
パスはプロジェクト・フォルダを基点とします。
本プロパティを省略した場合は，CX の標準インクルード・ファイル・フォルダのみ検索します。
cx コマンドの -I オプションに相当します。
指定したインクルード・パスはサブプロパティとして表示します。
なお，プロジェクト・ツリーにインクルード・ファイルを追加すると，インクルード・パスをサブプロパティの一番最初に追加します。
インクルード・パスに大文字，小文字の区別はありません。
<table>
<tr><td>デフォルト</td><td>追加のインクルード・パス [定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，パス編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td>システム・インクルード・パス</td><td colspan="2">アセンブル時にシステムが設定するインクルード・パスの指定順を変更します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
システム・インクルード・パスは，追加のインクルード・パスより低い優先度で検索します。
パスはプロジェクト・フォルダを基点とします。
cx コマンドの -I オプションに相当します。
インクルード・パスはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>システム・インクルード・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，システム・インクルード・パス順設定 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>変更不可（インクルード・パスの設定順の変更のみ可能）</td></tr>
<tr><td>定義マクロ</td><td colspan="2">定義したいマクロ名を指定します。
「マクロ名 = 定義値」の形式で 1 行に 1 つずつ指定します。
「= 定義値」の部分は省略可能で，省略した場合，定義値を 1 とします。
cx コマンドの -D オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>定義マクロ [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>256 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

(5) [よく使うオプション（リンク）]

リンク時によく使うオプションに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，ライブラリ用のプロジェクトの場合は表示しません。

<table>
<tr><td>使用するライブラリ・ファイル</td><td colspan="2">標準ライブラリ以外の使用するライブラリ・ファイル（libxxx.lib，または libxxx.a）を指定します。
libxxx.lib を優先して検索して，見つからなかった場合に libxxx.a を検索します。
xxx のみを指定してください（例：“user”と指定すると，libuser.lib，または libuser.a を指定したものとみなします）。
1 行に 1 ファイルずつ指定します。
ライブラリ・ファイルはライブラリ・パスから検索します。
cx コマンドの -l オプションに相当します。
指定したライブラリ・ファイル名はサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>使用するライブラリ・ファイル [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>249 文字までの文字列
256 個まで指定可能です。</td></tr>
<tr><td>追加のライブラリ・パス</td><td colspan="2">標準ライブラリ以外の使用するライブラリ・ファイルの検索フォルダを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
相対パスを指定した場合は，プロジェクト・フォルダを基点とします。
cx コマンドの -L オプションに相当します。
指定したライブラリ・パス名はサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>追加のライブラリ・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，パス編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>259 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

<table>
<tr><td>出力フォルダ</td><td colspan="2">生成するロード・モジュールの出力先フォルダを指定します。
相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。
絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。
次のプレースホルダに対応しています。
　%BuildModeName%　：ビルド・モード名に置換します。
空欄の場合は，プロジェクト・フォルダを指定したものとみなします。
cx コマンドの -o オプションに相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td>%BuildModeName%</td></tr>
<tr><td></td><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>247 文字までの文字列</td></tr>
<tr><td>出力ファイル名</td><td colspan="2">生成するロード・モジュールのファイル名を指定します。
.lmf 以外の拡張子を指定することはできません。
拡張子を省略した場合は，.lmf を自動的に付加します。
ターゲットがマルチコア CPU の場合は，共通部用ロード・モジュールとコア n 用ロード・モジュールを生成し，それらを元に最終ロード・モジュールを生成します（n：ターゲット CPU が持つコアの数）。
　共通部用ロード・モジュール：拡張子を除いた入力文字列 _cmn.lmf
　コア n 用ロード・モジュール：拡張子を除いた入力文字列 _pen.lmf
次のプレースホルダに対応しています。
　%ActiveProjectName%　：アクティブ・プロジェクト名に置換します。
　%MainProjectName%　：メイン・プロジェクト名に置換します。
　%ProjectName%　：プロジェクト名に置換します。
cx コマンドの -o オプションに相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td>%ProjectName%.lmf</td></tr>
<tr><td></td><td>変更方法</td><td>テキスト・ボックスによる直接入力</td></tr>
<tr><td></td><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

（6）［よく使うオプション（ヘキサ出力）］

ヘキサ出力時によく使うオプションに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，ライブラリ用のプロジェクトの場合は表示しません。

ヘキサ・ファイル・フォーマット	出力するヘキサ・ファイルのフォーマットを選択します。 cx コマンドの -Xhex_format オプションに相当します。		
	デフォルト	インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	インテル拡張ヘキサ・フォーマット (-Xhex_format=I)	ヘキサ・ファイルをインテル拡張ヘキサ・フォーマットとします。
		インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(なし)	ヘキサ・ファイルをインテル拡張ヘキサ・フォーマット（32 ビット・アドレス）とします。
		モトローラ S タイプ・フォーマット (スタンダード・アドレス)(-Xhex_format=S)	ヘキサ・ファイルをモトローラ S タイプ・フォーマット（スタンダード・アドレス）とします。
		モトローラ S タイプ・フォーマット (32 ビット・アドレス)(-Xhex_format=s)	ヘキサ・ファイルをモトローラ S タイプ・フォーマット（32 ビット・アドレス）とします。
		拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)	ヘキサ・ファイルを拡張テクトロニクス・ヘキサ・フォーマットとします。

(7) [レジスタ・モード]

レジスタ・モードに関する詳細情報の表示，および設定の変更を行います。

レジスタ・モード	ソフトウエア・レジスタ・バンク機能のレジスタ・モード（コンパイラが使用するレジスタの本数）注を選択します。 cx コマンドの -Xreg_mode オプションに相当します。		
	デフォルト	32 レジスタ・モード (なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	32 レジスタ・モード (なし)	レジスタ・モードを 32 とします。
		26 レジスタ・モード (-Xreg_mode=26)	レジスタ・モードを 26 とします。
		22 レジスタ・モード (-Xreg_mode=22)	レジスタ・モードを 22 とします。
		汎用レジスタ・モード (-Xreg_mode=common)	レジスタ・モードを 22 とします。 レジスタ・モードに依存しないオブジェクト・モジュール・ファイルを生成するために使用します。

注 CX が提供しているレジスタ・モードを以下に示します。

レジスタ・モード	作業用レジスタ	レジスタ変数用レジスタ
common	r10 ～ r14	r25 ～ r29
22 レジスタ・モード	r10 ～ r14	r25 ～ r29
26 レジスタ・モード	r10 ～ r16	r23 ～ r29
32 レジスタ・モード	r10 ～ r19	r20 ～ r29

(8)［フラッシュ対応］

フラッシュ対応に関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，ライブラリ用のプロジェクトの場合は表示しません。

<table>
<tr><td rowspan="4">ブート－フラッシュの再リンク機能を利用する</td><td colspan="3">ブート－フラッシュ再リンク機能を利用するかどうかを選択します。
フラッシュ領域側とブート領域側の双方での指定が必要です。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td>指定可能値</td><td>はい</td><td>ブート－フラッシュ再リンク機能を利用します。</td></tr>
<tr><td></td><td>いいえ</td><td>ブート－フラッシュ再リンク機能を利用しません。</td></tr>
<tr><td rowspan="5">生成するロード・モジュール・ファイルの種類</td><td colspan="3">生成するロード・モジュール・ファイルの種類を選択します。
cx コマンドの -Xflash オプションに相当します。
なお，本プロパティは，［ブート－フラッシュの再リンク機能を利用する］プロパティで［はい］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">ブート領域用ロード・モジュール・ファイル (なし)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>ブート領域用ロード・モジュール・ファイル (なし)</td><td>ブート領域側のロード・モジュール・ファイルを生成します。
［ROM 化オプション］タブの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［いいえ］を選択，または［ROM 化処理前のロード・モジュール・ファイルを出力する］プロパティで［はい］を選択してください。</td></tr>
<tr><td>フラッシュ領域用ロード・モジュール・ファイル (-Xflash)</td><td>フラッシュ領域側のロード・モジュール・ファイルを生成します。
［ブート領域用ロード・モジュール・ファイル名］プロパティにブート領域側のロード・モジュール・ファイル名を指定してください。</td></tr>
</table>

ブート領域用ロード・モジュール・ファイル名	ブート領域側のロード・モジュール・ファイルの名前を指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 ［生成するロード・モジュール・ファイルの種類］プロパティで［フラッシュ領域用ロード・モジュール・ファイル (-Xflash)］を選択した場合は，本プロパティを必ず指定してください。 cx コマンドの -Xflash オプションに相当します。 なお，本プロパティは，［生成するロード・モジュール・・ファイルの種類］プロパティで［フラッシュ領域用ロード・モジュール・ファイル (-Xflash)］を選択した場合のみ表示します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，ブート領域用ロード・モジュール・ファイルを指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
分岐テーブルのアドレス	分岐テーブルの先頭アドレスを指定します。 フラッシュ領域側とブート領域側の双方で同じアドレスを指定します。 奇数を指定した場合は，偶数に補正します。 cx コマンドの -Xflash_ext_table オプションに相当します。 なお，本プロパティは，［ブート－フラッシュの再リンク機能を利用する］プロパティで［はい］を選択した場合のみ表示します。	
	デフォルト	0（16 進数）
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ FFFFFFFF（16 進数）

(9)［デバイス］

デバイスに関する詳細情報の表示，および設定の変更を行います。

セキュリティ ID を設定する	フラッシュ・メモリ搭載デバイスのセキュリティ ID 設定するかどうかを選択します。 cx コマンドの -Xsecurity_id オプションに相当します。 なお，本プロパティは，セキュリティ ID 機能を持たないデバイスを使用するプロジェクトの場合，およびライブラリ用のプロジェクトの場合は表示しません。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	［セキュリティ ID］プロパティでセキュリティ ID を設定します。
		いいえ (-Xsecurity_id=none)	セキュリティ ID を設定しません。

セキュリティ ID	フラッシュ・メモリ搭載デバイスのセキュリティ ID を指定します。 cx コマンドの -Xsecurity_id オプションに相当します。 空欄の場合は，0x*f が最大桁*を指定したものとみなします。 なお，本プロパティは，［セキュリティ ID を設定する］プロパティで［はい］を選択した場合のみ表示します。	
	デフォルト	0x*f が最大桁*
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0 ～ *デバイスで対応する最大桁数*（16 進数）
プログラマブル I/O 領域開始アドレス	プログラマブル I/O 領域の使用と開始アドレスを指定します。 アドレスはデバイス固有のサイズでアラインします。 デバイス固有幅の下位ビットに値が入っている場合は，0 に補正します。 空欄の場合は，各デバイスで既定しているアドレスを指定したものとみなします。 cx コマンドの -Xprogrammable_io オプションに相当します。 なお，本プロパティは，プログラマブル I/O 機能を持たないデバイスを使用するプロジェクトの場合，およびライブラリ用のプロジェクトのとき，［出力ファイルの種類と場所］カテゴリの［デバイス共通オブジェクト・モジュールを出力する］プロパティで［いいえ (デバイス固有)(なし)］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	16 進数（選択したデバイスに依存します）

（10）［エラー出力］

エラー出力に関する詳細情報の表示，および設定の変更を行います。

エラー・メッセージ・ファイルを出力する	エラー・メッセージ・ファイルを出力するかどうかを選択します。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティの選択にかかわらず，エラー・メッセージは出力 パネルに表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xerror_file)	エラー・メッセージ・ファイルを出力します。
		いいえ	エラー・メッセージ・ファイルを出力しません。

エラー・メッセージ・ファイル出力フォルダ	エラー・メッセージ・ファイルの出力先フォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティは，［エラー・メッセージ・ファイルを出力する］プロパティで［はい (-Xerror_file)］を選択した場合のみ表示します。	
	デフォルト	%BuildModeName%
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集
	指定可能値	247 文字までの文字列
エラー・メッセージ・ファイル名	エラー・メッセージ・ファイルの名前を指定します。 拡張子は自由に指定可能です。 次のプレースホルダに対応しています。 %ActiveProjectName% ：アクティブ・プロジェクト名に置換します。 %MainProjectName% ：メイン・プロジェクト名に置換します。 %ProjectName% ：プロジェクト名に置換します。 空欄の場合は，%ProjectName%.err を指定したものとみなします。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティは，［エラー・メッセージ・ファイルを出力する］プロパティで［はい (-Xerror_file)］を選択した場合のみ表示します。	
	デフォルト	%ProjectName%.err
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	259 文字までの文字列

（11）［警告メッセージ］

警告メッセージに関する詳細情報の表示，および設定の変更を行います。

必ず表示させる警告メッセージ	必ず表示させたい警告メッセージの番号を指定します。 複数指定する場合は，メッセージ番号をカンマで区切って指定します（例：02042,02107）。 また，ハイフンを使用して，区間設定を行うこともできます（例：02222-02554,02699-02782）。 ［表示させない警告メッセージ］プロパティと同一の番号を指定した場合は，本プロパティを優先します。 cx コマンドの -Xwarning オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	2048 文字までの文字列

表示させない警告メッセージ	表示させない警告メッセージの番号を指定します。 複数指定する場合は，メッセージ番号をカンマで区切って指定します（例：02042,02107）。 また，ハイフンを使用して，区間設定を行うこともできます（例：02222-02554,02699-02782）。 ［必ず表示させる警告メッセージ］プロパティと同一の番号を指定した場合は，［必ず表示させる警告メッセージ］プロパティを優先します。 cx コマンドの -Xno_warning オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	2048 文字までの文字列

（12）［ビルド方法］

ビルド方法に関する詳細情報の表示，および設定の変更を行います。

一括ビルドを行う	複数のファイルを同時にコンパイル／アセンブル／リンクしてロード・モジュール・ファイルを生成するかどうかを選択します。 ただし，個別オプションを設定しているファイル，およびビルド前実行の対象となっているファイルは，一括ビルドの対象から除きます。 一括ビルドについての詳細は，「2. 16. 5　複数のファイルのコンパイル／アセンブル／リンクを同時に実行する」を参照してください。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	複数のファイルを同時にコンパイル／アセンブル／リンクします。 アセンブラ・ソース・ファイル（個別オプションを設定しているファイルを除く）のアセンブルには，［コンパイル・オプション］タブの設定を使用します。
		いいえ	ファイル単位でコンパイル／アセンブル／リンクを行います。
インクルード・ファイルが存在しないソースの扱い	ソース・ファイルがインクルードしているファイルが存在しない場合，そのソース・ファイルを再コンパイル／アセンブルするかどうかを選択します。		
	デフォルト	再コンパイル／アセンブルする	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	再コンパイル／アセンブルする	インクルード・ファイルが存在しない場合，ソース・ファイルを再コンパイル／アセンブルします。
		再コンパイル／アセンブルしない	インクルード・ファイルが存在しない場合でも，ソース・ファイル再コンパイル／アセンブルしません。

コアごとにビルド・オプションを指定する	マルチコア対応デバイスを使用するプロジェクトにおいて，コアごとにビルド・オプションを指定するかどうかを選択します。 なお，本プロパティは，マルチコア対応デバイスを使用するプロジェクトの場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	［コンパイル・オプション］タブ，および［アセンブル・オプション］タブの［コア *n* 用オプション］プロパティで，コアごとにビルド・オプションを指定します。 コアごとに指定可能なオプションは，-Xr，および -Xsdata オプションです。
		いいえ	コアごとにはビルド・オプションを指定しません。

（13）［バージョン選択］

ビルド・ツールのバージョン選択に関する詳細情報の表示，および設定の変更を行います。

使用するコンパイラ・パッケージのインストール・フォルダ	使用するコンパイラ・パッケージをインストールしているフォルダを表示します。		
	デフォルト	*インストール・フォルダ名*	
	変更方法	変更不可	
使用するコンパイラ・パッケージのバージョン	使用するコンパイラ・パッケージのバージョンを選択します。 この設定はすべてのビルド・モードで共通です。		
	デフォルト	常にインストール済みの最新版	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	常にインストール済みの最新版	インストールしているコンパイラ・パッケージの内，最新バージョンを使用します。
		インストールしているコンパイラ・パッケージのバージョン	選択したバージョンのコンパイラ・パッケージを使用します。
インストール済みのコンパイラ・パッケージの最新バージョン	［使用するコンパイラ・パッケージのバージョン］プロパティで［常にインストール済みの最新版］を選択した場合に使用するコンパイラ・パッケージのバージョンを表示します。 この設定はすべてのビルド・モードで共通です。 なお，本プロパティは，［使用するコンパイラ・パッケージのバージョン］プロパティで［常にインストール済みの最新版］を選択した場合のみ表示します。		
	デフォルト	*インストールしているコンパイラ・パッケージの最新バージョン*	
	変更方法	変更不可	

（14）［記録］

記録に関する詳細情報の表示，および設定の変更を行います。

メモ	このビルド・ツールにメモを追加します。 1 行に 1 項目ずつ指定します。 この設定はすべてのビルド・モードで共通です。 追加したメモはサブプロパティとして表示します。	
	デフォルト	メモ [*項目数*]
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	256 文字までの文字列 256 個まで指定可能です。

（15）［その他］

ビルド・ツールに関するその他の詳細情報の表示，および設定の変更を行います。

出力メッセージ・フォーマット	ビルド中のメッセージのフォーマットを指定します。 対象となるのは，使用するビルド・ツール，およびプラグインによって追加されたコマンドの出力メッセージです。 ［ビルド前に実行するコマンド］プロパティ，および［ビルド後に実行するコマンド］プロパティなどで指定したコマンドの出力メッセージは対象外です。 次のプレースホルダに対応しています。 %Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。 %Program% ：実行中のプログラム名に置換します。 %TargetFiles% ：コンパイル／アセンブル中のファイル名，またはリンク後の出力ファイル名に置換します。 空欄の場合は，%Program% %Options% を自動的に設定します。		
	デフォルト	%TargetFiles%	
	変更方法	テキスト・ボックスによる直接入力（256 文字までの文字列），またはドロップダウン・リストによる選択	
	指定可能値	%TargetFiles%	出力メッセージにファイル名を表示します。
		%TargetFiles%: %Options%	出力メッセージにファイル名とコマンド・ライン・オプションを表示します。
		%Program% %Options%	出力メッセージにプログラム名とコマンド・ライン・オプションを表示します。

<table>
<tr><td rowspan="4">ビルド・オプション一覧表示フォーマット</td><td colspan="2">ビルド・オプション一覧（「2. 15. 4 ビルド・オプションを一覧表示する」参照）の表示フォーマットを指定します。
対象となるのは，使用するビルド・ツール，およびプラグインによって追加されたコマンドのオプションです。
［ビルド前に実行するコマンド］プロパティ，および［ビルド後に実行するコマンド］プロパティなどで指定したコマンドのオプションは対象外です。
次のプレースホルダに対応しています。
%Options% ： ビルド実行時のコマンド・ライン・オプションに置換します。
%Program% ： 実行中のプログラム名に置換します。
%TargetFiles% ： コンパイル／アセンブル中のファイル名，またはリンク後の出力ファイル名に置換します。
空欄の場合は，%TargetFiles% : %Program% %Options% を自動的に設定します。</td></tr>
<tr><td>デフォルト</td><td>%TargetFiles% : %Program% %Options%</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>256 文字までの文字列</td></tr>
<tr><td rowspan="4">一時作業フォルダ</td><td colspan="2">cx コマンドが実行中に生成する一時的なファイルの格納先フォルダを指定します。
相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。
絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。
空欄の場合は，以下の順番で作業用フォルダを決定します。
（1）環境変数 TEMP で指定したフォルダ
（2）環境変数 TMP で指定したフォルダ
（3）メイン・プロジェクト，またはサブプロジェクトのフォルダ
cx コマンドの -Xtemp_path オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td>空欄</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>200 文字までの文字列</td></tr>
</table>

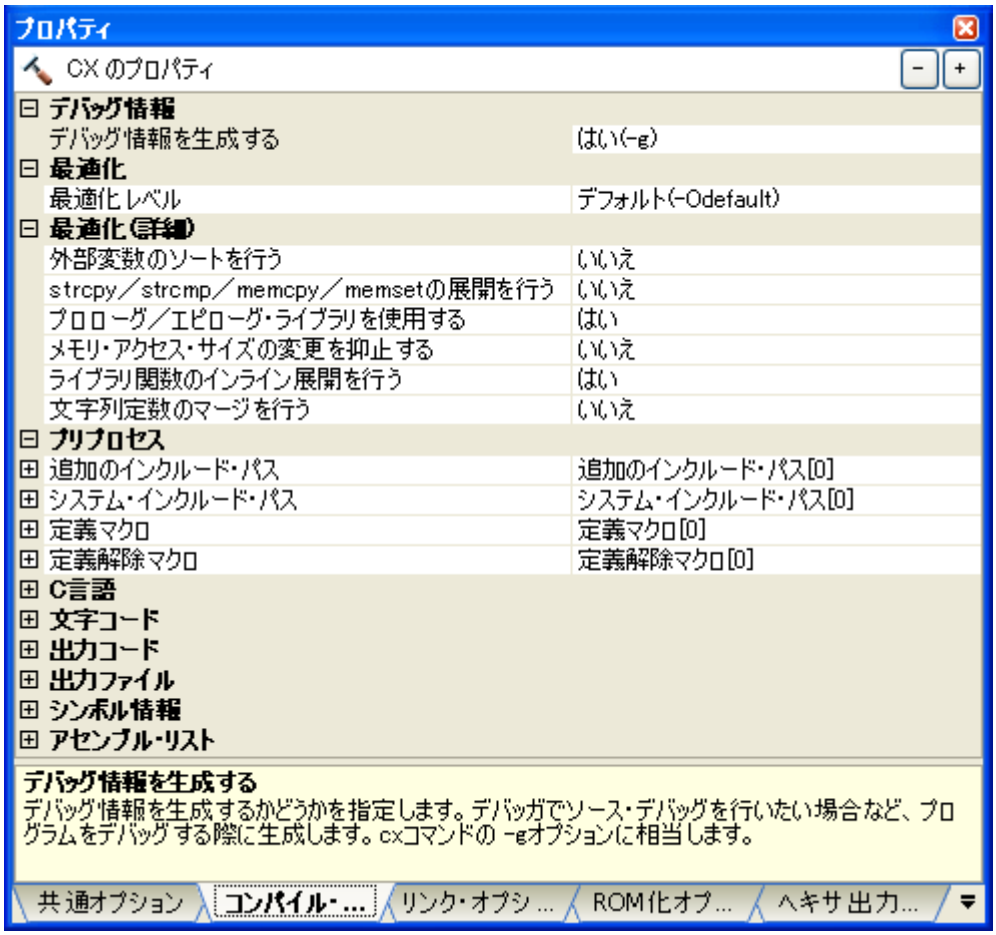
<table>
<tr><td>ビルド前に実行するコマンド</td><td>ビルド処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，ビルド処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
<table><tr><td>デフォルト</td><td>ビルド前に実行するコマンド [定義数]</td></tr><tr><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr><tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr></table></td></tr>
</table>

<table>
<tr><td rowspan="4">ビルド後に実行するコマンド</td><td colspan="2">ビルド処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，ビルド処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>ビルド後に実行するコマンド [定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td rowspan="4">その他の追加オプション</td><td colspan="2">その他に追加するオプションを入力します。
なお，ここで設定したオプションは，cx のオプション群の最後に付加します。</td></tr>
<tr><td>デフォルト</td><td>空欄</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

［コンパイル・オプション］タブ

本タブでは，コンパイル・フェーズに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

図 A―5　プロパティ パネル：［コンパイル・オプション］タブ



[各カテゴリの説明]

(1) [デバッグ情報]

デバッグ情報に関する詳細情報の表示，および設定の変更を行います。

デバッグ情報を生成する	デバッグ情報を生成するかどうかを選択します。 出力ファイル中にソース・デバッグ用の情報を出力することにより，デバッガでのソース・デバッグが可能となります。 cx コマンドの -g オプションに相当します。		
	デフォルト	はい (-g)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-g)	デバッグ情報を生成します。
		いいえ	デバッグ情報を生成しません。

(2) [最適化]

最適化に関する詳細情報の表示，および設定の変更を行います。

最適化レベル	コンパイルの最適化レベルを選択します。 cx コマンドの -O オプションに相当します。		
	デフォルト	デフォルト (-Odefault)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	デフォルト (-Odefault)	デバッグに影響しない範囲の最適化（式の最適化，およびレジスタ割り付けなど）を行います。
		サイズ優先 (-Osize)	オブジェクト・サイズ優先の最適化を行います。 ROM/RAM 容量の削減を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
		実行速度優先 (-Ospeed)	実行速度優先の最適化を行います。 実行速度の短縮を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
		デバッグ優先 (-Onothing)	デバッグを優先して最適化を行います。 デバッグのしやすさを重視し，デフォルトで実行する最適化を含むすべての最適化を抑止します。
		詳細指定	[最適化 (詳細)] カテゴリにプロパティを追加表示し，そこで選択した最適化項目を優先して最適化を行います。

(3) [最適化 (詳細)]

最適化に関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td rowspan="4">ループ展開最大数</td><td colspan="3">for，while などのループを展開する最大数を指定します。
0，または 1 を指定した場合は，展開を抑止します。
空欄の場合は，4 を指定したものとみなします。
cx コマンドの -Ounroll オプションに相当します。
なお，本プロパティは，［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">0</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td colspan="2">0 ～ 999（10 進数），または空欄</td></tr>
<tr><td rowspan="4">未使用 static 関数の削除を行う</td><td colspan="3">呼び出されない static 関数の削除を行うかどうかを選択します。
cx コマンドの -Odelete_static_func オプションに相当します。
なお，本プロパティは，［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Odelete_static_func=on)</td><td>呼び出されない static 関数の削除を行います。</td></tr>
<tr><td>いいえ</td><td>呼び出されない static 関数の削除を指定しません。</td></tr>
<tr><td rowspan="5">関数のインライン展開を行う</td><td colspan="3">関数を呼び出し箇所にインライン展開するかどうかを選択します。
cx コマンドの -Oinline オプションに相当します。
なお，本プロパティは，［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい (指定関数のみ)(なし)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="4">指定可能値</td><td>はい (指定関数のみ)(なし)</td><td>#pragma inline 指定した関数を呼び出し箇所にインライン展開します。</td></tr>
<tr><td>はい (自動判別)(-Oinline=2)</td><td>自動的にインライン展開対象の関数を判別して展開します。</td></tr>
<tr><td>はい (コード・サイズが増加しないよう自動判別)(-Oinline=3)</td><td>コード・サイズがなるべく増加しない範囲で，自動的にインライン展開対象の関数を判別して展開します。</td></tr>
<tr><td>いいえ (-Oinline=0)</td><td>関数のインライン展開を指定しません。</td></tr>
</table>

パイプライン最適化を行う	機械語レベルで命令の並べ替えを行い，プログラムの実行性能を引き出すかどうかを選択します。 cx コマンドの -Opipeline オプションに相当します。 なお，本プロパティは，V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合，かつ［最適化］カテゴリの［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Opipeline=on)	パイプライン最適化を行います。
		いいえ	パイプライン最適化を指定しません。
共通コードのサブルーチン化を行う	共通コードのサブルーチン化を行うかどうかを選択します。 cx コマンドの -Osubroutine オプションに相当します。 なお，本プロパティは，［最適化］カテゴリの［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Osubroutine=1)	共通コードのサブルーチン化を行います。
		いいえ	共通コードのサブルーチン化を行いません。
関数末尾の関数呼び出しに jr 命令を使用する	関数の末尾が関数呼び出しの場合に，jarl 命令の代わりに jr 命令を優先的に使用するかどうかを選択します。 cx コマンドの -Otail_call オプションに相当します。 なお，本プロパティは，［最適化］カテゴリの［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Otail_call=on)	関数の末尾が関数呼び出しの場合に，jarl 命令の代わりに jr 命令を優先的に使用します。 lp の退避／復帰コードを削除し，コード・サイズを小さくすることができます。 ただし，一部のデバッグ機能を使用することができなくなります。
		いいえ	関数の末尾が関数呼び出しの場合に，jarl 命令を使用します。

未使用関数の削除を行う	未使用関数の削除を行うかどうかを選択します。 本プロパティは，インライン展開後に不要になった関数を削除するのに有効です。 cx コマンドの -Xdelete_func オプションに相当します。 なお，本プロパティは，［シンボル情報］カテゴリの［使用するシンボル情報ファイル］プロパティに値が表示されている場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xdelete_func)	未使用関数の削除を行います。
		いいえ	未使用関数の削除を指定しません。
未使用関数の検索開始関数	未使用関数の削除を行う際，未使用関数の検索を開始する関数の名前を指定します。 複数指定する場合は，関数名をカンマで区切って指定します（例：main,init)。 空欄の場合は，main を指定したものとみなします。 cx コマンドの -Xdelete_func オプションに相当します。 なお，本プロパティは，［未使用関数の削除を行う］プロパティで［はい］を選択した場合のみ表示します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	1022 文字までの文字列	
外部変数のソートを行う	外部変数のソートを行うかどうかを選択します。 本プロパティは，RAM 容量を削減するためのものです。 cx コマンドの -Xsort_var オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xsort_var)	const/sconst セクション以外のセクションに配置している外部変数をアライメントが大きい順に並び替えます。
		いいえ	外部変数のソートを指定しません。

<table>
<tr><td rowspan="5">strcpy ／ strcmp ／ memcpy ／ memset の展開を行う</td><td colspan="3">配列（文字列を含む），および構造体の整列条件を 4 バイトとし，関数 strcpy()，strcmp()，memcpy()，memset() の呼び出しをインライン展開するかどうかを選択します。
生成するプログラムの実行速度は高速になりますが，コード・サイズは増大します。
cx コマンドの -Xinline_strcpy オプションに相当します。
なお，本プロパティは，［出力コード］カテゴリの［構造体パッキングを行う］プロパティで［いいえ］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xinline_strcpy)</td><td>関数 strcpy()，strcmp()，memcpy()，memset() の呼び出しをインライン展開します。</td></tr>
<tr><td>いいえ</td><td>関数 strcpy()，strcmp()，memcpy()，memset() のインライン展開を指定しません。</td></tr>
<tr><td rowspan="5">プロローグ／エピローグ・ライブラリを使用する</td><td colspan="3">関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にするかどうかを選択します。
cx コマンドの -Xpro_epi_runtime オプションに相当します。
なお，本プロパティは，［最適化］カテゴリの［最適化レベル］プロパティで［実行速度優先 (-Ospeed)］を選択した場合は表示しません。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい</td><td>関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にします。</td></tr>
<tr><td>いいえ (-Xpro_epi_runtime=off)</td><td>関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理に指定しません。</td></tr>
<tr><td rowspan="5">メモリ・アクセス・サイズの変更を抑止する</td><td colspan="3">ロード／ストア命令を 1 ビット操作命令（set1，clr1，tst1，not1）へ置き換える動作を禁止するかどうかを選択します。
cx コマンドの -Xkeep_access_size オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xkeep_access_size)</td><td>ロード／ストア命令を 1 ビット操作命令へ置き換える動作を禁止します。</td></tr>
<tr><td>いいえ</td><td>ロード／ストア命令の代わり 1 ビット操作命令を生成することがあります。</td></tr>
</table>

<table>
<tr><td rowspan="5">ライブラリ関数のインライン展開を行う</td><td colspan="3">ライブラリ関数の呼び出し箇所において，インライン展開を行うかどうかを選択します。
cx コマンドの -Xcall_lib オプションに相当します。
なお，本プロパティは，［strcpy ／ strcmp ／ memcpy ／ memset の展開を行う］プロパティで［いいえ］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい</td><td>sqrtf（FPU を持つデバイスのみ），memcmp，memcpy，memset 関数を呼び出す代わりに，適切な命令（命令列）を生成します。</td></tr>
<tr><td>いいえ (-Xcall_lib)</td><td>常にライブラリ関数を呼び出す命令を生成します。</td></tr>
<tr><td rowspan="5">文字列定数のマージを行う</td><td colspan="3">ソース・ファイル内で同じ文字列定数が複数存在する場合，これらをまとめて 1 つの領域に割り付けるかどうかを選択します。
cx コマンドの -Xmerge_string オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xmerge_string)</td><td>ソース・ファイル内で複数存在する同じ文字列定数をまとめて 1 つの領域に割り付けます。</td></tr>
<tr><td>いいえ</td><td>ソース・ファイル内で複数存在する同じ文字列定数をそれぞれ別々の領域に割り付けます。</td></tr>
</table>

(4) ［プリプロセス］

プリプロセスに関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td>追加のインクルード・パス</td><td>コンパイル時の追加のインクルード・パスを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
指定したインクルード・パスは，CX の標準インクルード・ファイル・フォルダよりも優先して検索します。
パスはプロジェクト・フォルダを基点とします。
本プロパティを省略した場合は，CX の標準インクルード・ファイル・フォルダのみ検索します。
cx コマンドの -I オプションに相当します。
指定したインクルード・パスはサブプロパティとして表示します。
なお，プロジェクト・ツリーにインクルード・ファイルを追加すると，インクルード・パスをサブプロパティの一番最初に追加します。
インクルード・パスに大文字，小文字の区別はありません。
<table>
<tr><td>デフォルト</td><td>追加のインクルード・パス [定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，パス編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td>システム・インクルード・パス</td><td colspan="2">コンパイル時にシステムが設定するインクルード・パスの指定順を変更します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
システム・インクルード・パスは，追加のインクルード・パスより低い優先度で検索します。
パスはプロジェクト・フォルダを基点とします。
cx コマンドの -I オプションに相当します。
インクルード・パスはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>システム・インクルード・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，システム・インクルード・パス順設定 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>変更不可（インクルード・パスの設定順の変更のみ可能）</td></tr>
<tr><td>定義マクロ</td><td colspan="2">定義したいマクロ名を指定します。
「マクロ名 = 定義値」の形式で 1 行に 1 つずつ指定します。
「= 定義値」の部分は省略可能で，省略した場合，定義値を 1 とします。
cx コマンドの -D オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>定義マクロ [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>256 文字までの文字列
256 個まで指定可能です。</td></tr>
<tr><td>定義解除マクロ</td><td colspan="2">定義解除したいマクロ名を指定します。
「マクロ名」の形式で 1 行に 1 つずつ指定します。
cx コマンドの -U オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>定義解除マクロ [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>256 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

プリプロセス結果にCソース・コメントを出力する	プリプロセス結果のファイルに，Cソースのコメントを出力するかどうかを選択します。 cxコマンドの -Xpreprocess オプションに相当します。 なお，本プロパティは，［出力ファイル］カテゴリの［プリプロセス処理したソースを出力する］プロパティで［はい (-P)］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xpreprocess=comment)	プリプロセス結果のファイルに，Cソースのコメントを出力します。
		いいえ	プリプロセス結果のファイルに，Cソースのコメントを出力しません。
プリプロセス結果に行番号情報を出力する	プリプロセス結果のファイルに，Cソースの行番号情報を出力するかどうかを選択します。 cxコマンドの -Xpreprocess オプションに相当します。 なお，本プロパティは，［出力ファイル］カテゴリの［プリプロセス処理したソースを出力する］プロパティで［はい (-P)］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xpreprocess=line)	プリプロセス結果のファイルに，Cソースの行番号情報を出力します。
		いいえ	プリプロセス結果のファイルに，Cソースの行番号情報を出力しません。

(5)［C言語］

C言語に関する詳細情報の表示，および設定の変更を行います。

ANSI規格に厳密に合わせてコンパイルする	Cソース・プログラムをANSI規格に厳密にあわせて処理し，規格に反する記述に対してエラーや警告を出力するかどうかを選択します。 cxコマンドの -Xansi オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xansi)	Cソース・プログラムをANSI規格に厳密にあわせて処理し，規格に反する記述に対してエラーや警告を出力します。
		いいえ	従来のC言語の仕様との両立性を持たせ，警告を出力して処理を続行します。

char の符号	型指定子（signed，unsigned）の付かない char 型に対し，符号付きとするか，符号なしとするかを選択します。 cx コマンドの -Xchar オプションに相当します。		
	デフォルト	符号付き	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	符号付き	型指定子の付かない char 型に対し，符号付きとします。
		符号なし (-Xchar=unsigned)	型指定子の付かない char 型に対し，符号なしとします。
enum の型	列挙型に対して，どの整数型として扱うかを選択します。 cx コマンドの -Xenum_type オプションに相当します。		
	デフォルト	signed int(なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	signed int(なし)	列挙型に対して，int 型として扱います。
		自動 (-Xenum_type=auto)	各列挙型について，その型のすべての列挙子の値を表現可能な最小の整数型として扱います。
仮定義を定義に変更する	変数の仮定義を定義として扱うかどうかを選択します。 cx コマンドの -Xdef_var オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xdef_var)	変数の仮定義を定義として扱います。
		いいえ	変数の仮定義を定義として扱いません。

（6）［文字コード］

文字コードに関する詳細情報の表示，および設定の変更を行います。

文字コード	ソース・ファイル中の日本語／中国語のコメント，文字列に対して，使用する文字コードを選択します。 cx コマンドの -Xcharacter_set オプションに相当します。		
	デフォルト	SJIS(なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	SJIS(なし)	ソース・ファイル中の日本語の文字コードを SJIS と解釈します。
		EUC(-Xcharacter_set=euc_jp)	ソース・ファイル中の日本語の文字コードを EUC と解釈します。
		UFT-8(-Xcharacter_set=utf8)	ソース・ファイル中の日本語の文字コードを UFT-8 と解釈します。
		Big5(-Xcharacter_set=big5)	ソース・ファイル中の中国語の文字コードを繁体字中国語と解釈します。
		GB2312(-Xcharacter_set=gb2312)	ソース・ファイル中の中国語の文字コードを簡体字中国語と解釈します。
		処理しない (-Xcharacter_set=none)	ソース・ファイル中の日本語の文字コードを処理しません。

(7) [出力コード]

出力コードに関する詳細情報の表示，および設定の変更を行います。

構造体パッキングを行う	構造体パッキングを行うかどうかを選択します。 構造体メンバをメンバ型に応じてアライメントすることなく，指定したアライメント値を用いることができます。 cx コマンドの -Xpack オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	1 バイト (-Xpack=1)	構造体メンバを 1 バイトのアライメントで整列します。
		2 バイト (-Xpack=2)	構造体メンバを 2 バイトのアライメントで整列します。
		4 バイト (-Xpack=4)	構造体メンバを 4 バイトのアライメントで整列します。
		8 バイト (-Xpack=8)	構造体メンバを 8 バイトのアライメントで整列します。
		いいえ	構造体パッキングを行いません。

<table>
<tr><td rowspan="5">アセンブラ・ソースにコメントを出力する</td><td colspan="3">出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力するかどうかを選択します。
cx コマンドの -Xpass_source オプションに相当します。
なお，本プロパティは，［出力ファイル］カテゴリの［アセンブラ・ソース・ファイルを出力する］プロパティで［はい (-Xasm_path)］を選択した場合，または［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xpass_source)</td><td>出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力します。</td></tr>
<tr><td>いいえ</td><td>出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力しません。</td></tr>
<tr><td rowspan="6">switch 文の出力コードの選択</td><td colspan="3">プログラム中の switch 文のコード出力形式を選択します。
cx コマンドの -Xswitch オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">自動選択 (なし)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="4">指定可能値</td><td>自動選択 (なし)</td><td>cx コマンドが最適な出力形式を自動的に選択します。</td></tr>
<tr><td>if-else(-Xswitch=ifelse)</td><td>プログラム中の switch 文を case 文の並びに沿った if-else 文と同じ形式で出力します。
頻度が多い順に case 文を書いているときやラベル数が少ないときに，本項目を選択します。
上から順に比較するので，頻繁に合致する case 文を先に記述すると余計な比較が減り，実行速度向上につながります。</td></tr>
<tr><td>バイナリ・サーチ (-Xswitch=binary)</td><td>プログラム中の switch 文をバイナリ・サーチ形式で出力します。
バイナリ・サーチ・アルゴリズムに用いて合致する case 文を探します。
ラベル数が多いときに本項目を選択すると，どの case 文も同じくらいの速さで見つけることができます。</td></tr>
<tr><td>テーブル分岐 (-Xswitch=table)</td><td>プログラム中の switch 文をテーブル・ジャンプ形式で出力します。
case 文の値を基にインデックス化したテーブルを参照し，switch 文の値により case ラベルを選択して処理を行います。
どの case 文にも同じくらい速く分岐します。
ただし，case 値が連続していないときは無駄な領域ができます。</td></tr>
</table>

<table>
<tr><td rowspan="5">switch テーブルのラベル・サイズ (バイト)</td><td colspan="3">switch 文の case ラベルに対する分岐テーブルの 1 ラベルあたりのサイズを選択します。
cx コマンドの -Xword_case オプションに相当します。
なお，本プロパティは，［switch 文の出力コードの選択］プロパティで［自動選択 (なし)］，または［テーブル分岐 (-Xswitch=table)］を選択した場合のみ表示します。
［switch 文の出力コードの選択］プロパティで［自動選択 (なし)］を選択したときに cx コマンドが［if-else(-Xswitch=ifelse)］，または［バイナリ・サーチ (-Xswitch=binary)］を自動選択した場合，本プロパティの設定は無効（2 バイト固定）となります。</td></tr>
<tr><td>デフォルト</td><td colspan="2">2 バイト (なし)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>2 バイト (なし)</td><td>switch 文の case ラベルに対する分岐テーブルを 1 ラベルあたり 2 バイトで生成します。</td></tr>
<tr><td>4 バイト (-Xword_case)</td><td>switch 文の case ラベルに対する分岐テーブルを 1 ラベルあたり 4 バイトで生成します。
長い switch 文のためにコンパイル・エラーとなる場合に選択します。</td></tr>
<tr><td rowspan="4">sdata/sbss セクションに配置するデータ長の上限値 (バイト)</td><td colspan="3">.sdata/.sbss セクションに配置するデータ長の上限サイズ（バイト）を指定します。
ただし，#pragma section 指令で .sdata/.sbss セクションを指定したデータは，そのサイズに関係なく .sdata/.sbss セクションに配置します。
マルチコア対応デバイスを使用している場合，［ビルド設定］タブの［対象コア番号］プロパティで［共通 (-Xmulti=cmn)］を選択したファイルは，本プロパティに 0 を指定したものとみなしてビルドを実行します。
cx コマンドの -Xsdata オプションに相当します。
なお，本プロパティは，［共通オプション］タブの［ビルド方法］カテゴリの［コアごとにビルド・オプションを指定する］プロパティで［いいえ］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">空欄</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td colspan="2">0 ～ 65535（10 進数）</td></tr>
<tr><td rowspan="4">sconst セクションのデータ長の上限値 (バイト)</td><td colspan="3">const 属性のデータ，文字列リテラルを .sconst セクションに配置する上限サイズ（バイト）を指定します。
cx コマンドの -Xsconst オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">32767</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td colspan="2">0 ～ 32767（10 進数）</td></tr>
</table>

浮動小数点演算方法	浮動小数点演算に対して，ランタイム・ライブラリの呼び出し命令を生成するか，FPU（浮動小数点ユニット）の浮動小数点演算命令を生成するかを選択します。 cx コマンドの -Xfloat【V850E2V3】オプションに相当します。 なお，本プロパティは，FPU を持つ V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合のみ表示します。		
	デフォルト	自動選択 (なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	自動選択 (なし)	ターゲット・デバイスが FPU を持たない場合は，浮動小数点演算命令を生成せず，ランタイム・ライブラリの呼び出し命令を生成します。 ターゲット・デバイスが FPU を持つ場合は，浮動小数点演算命令を生成します。
		ソフトウェアで行う (-Xfloat=soft)	浮動小数点演算に対して，ランタイム・ライブラリの呼び出し命令を生成します。
		FPU で行う (-Xfloat=fpu)	浮動小数点演算に対して，FPU（浮動小数点ユニット）の浮動小数点演算命令を生成します。
div ／ divu 除算命令を生成する	除算に対して，divq，および divqu 命令を生成する代わりに，div，および divu 命令を生成するかどうかを選択します。 divq，および divqu 命令は高速ですが，実行サイクル数がオペランドの値により変わります。 cx コマンドの -Xdiv【V850E2V3】オプションに相当します。 なお，本プロパティは，FPU を持つ V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xdiv)	除算に対して，div，および divu 命令を生成します。
		いいえ	除算に対して，divq，および divqu 命令を生成します。
32 ビット分岐命令を使用する	jarl，および jr 命令に対して，far jump 機能を使用するかどうかを選択します。 far jump 機能を使用することにより，jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。 cx コマンドの -Xasm_far_jump【V850E2】【V850E2V3】オプションに相当します。 なお，本プロパティは，V850E2 コア，および V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合，および［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xasm_far_jump)	jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。
		いいえ	jarl，および jr 命令としてアセンブルを行います。

Far Jump ファイル名	Far Jump ファイル名を指定します。 複数指定した場合は，最後に指定したものが有効となります。 Far Jump ファイルには，ファイルに記述した関数への分岐命令に対して，jmp 命令（V850E/ES コアの場合），または jarl32，および jr32 命令（V850E2 コア，および V850E2V3 アーキテクチャの場合）を使用したコードを出力します。 関数本体が jarl，および jr 命令では分岐できない範囲（± 2M バイト以上）にあり，cx コマンドがエラーを出力する場合，Far Jump ファイルを用いてコンパイルし直します。 なお，拡張子は .fjp としてください。 cx コマンドの -Xfar_jump オプションに相当します。 指定した Far Jump ファイル名はサブプロパティとして表示します。	
	デフォルト	Far Jump ファイル名 [*設定数*]
	変更方法	[...] ボタンをクリックし，パス編集 ダイアログをオープン →［参照］ボタンをクリックし，Far Jump ファイルを指定 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	259 文字までの文字列 5000 個まで指定可能です。 ただし，最後に指定したファイル名のみが有効となります。

（8）［出力ファイル］

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

アセンブラ・ソース・ファイルを出力する	C ソースのコンパイル結果のアセンブラ・ソース・ファイルを出力するかどうかを選択します。 cx コマンドの -Xasm_path オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xasm_path)	C ソースのコンパイル結果のアセンブラ・ソース・ファイルを出力します。
		いいえ	C ソースのコンパイル結果のアセンブラ・ソース・ファイルを出力しません。

<table>
<tr><td rowspan="4">アセンブラ・ソース・ファイル出力フォルダ</td><td colspan="3">アセンブラ・ソース・ファイルの出力先フォルダを指定します。
相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。
絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。
次のプレースホルダに対応しています。
　%BuildModeName%　：ビルド・モード名に置換します。
アセンブラ・ソース・ファイルは，C ソース・ファイルの拡張子を .asm で置き換えたファイル名で出力します。
空欄の場合は，プロジェクト・フォルダを指定したものとみなします。
cx コマンドの -Xasm_path オプションに相当します。
なお，本プロパティは，［アセンブラ・ソース・ファイルを出力する］プロパティで［はい (-Xasm_path)］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">%BuildModeName%</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td colspan="2">247 文字までの文字列</td></tr>
<tr><td rowspan="4">プリプロセス処理したソースを出力する</td><td colspan="3">ソース・ファイルに対して，プリプロセス処理を実行した結果をファイルに出力するかどうかを選択します。
ファイルの出力先は，［中間ファイル出力フォルダ］プロパティで指定したフォルダです。
ファイルは，ソース・ファイルの拡張子を .i で置き換えた名前で出力します。
cx コマンドの -P オプションに相当します。
なお，本プロパティは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td>指定可能値</td><td>はい (-P)</td><td>ソース・ファイルに対して，プリプロセス処理を実行した結果をファイルに出力します。</td></tr>
<tr><td></td><td>いいえ</td><td>ソース・ファイルに対して，プリプロセス処理を実行した結果をファイルに出力しません。</td></tr>
</table>

(9)［シンボル情報］

シンボル情報に関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td rowspan="3">使用するシンボル情報ファイル</td><td colspan="2">ビルド時に使用するシンボル情報ファイルの名前を表示します。
プロジェクトに登録している有効なシンボル情報ファイルを検索して使用します。
cx コマンドの -Xsymbol_file オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td>プロジェクトに登録しているシンボル情報ファイル名</td></tr>
<tr><td>変更方法</td><td>変更不可</td></tr>
</table>

（10）［アセンブル・リスト］

アセンブル・リストに関する詳細情報の表示，および設定の変更を行います。

アセンブル・リスト・ファイルを出力する	アセンブル・リスト・ファイルを出力するかどうかを選択します。 cx コマンドの -Xprn_path オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xprn_path)	アセンブル・リスト・ファイルを出力します。
		いいえ	アセンブル・リスト・ファイルを出力しません。
アセンブル・リスト・ファイル出力フォルダ	アセンブル・リスト・ファイルの出力先フォルダを指定します。 アセンブル・リスト・ファイルは，ソース・ファイル名の拡張子を .prn で置き換えた名前で出力します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 　%BuildModeName%　：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xprn_path オプションに相当します。 なお，本プロパティは，［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択した場合のみ表示します。		
	デフォルト	%BuildModeName%	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集	
	指定可能値	247 文字までの文字列	

（11）［外部変数レジスタ］

外部変数レジスタに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［共通オプション］タブの［レジスタ・モード］カテゴリの［レジスタ・モード］プロパティで［32 レジスタ・モード (なし)］，または［汎用レジスタ・モード (-Xreg_mode=common)］を選択した場合，および［ビルド方法］カテゴリの［コアごとにビルド・オプションを指定する］プロパティで［はい］を選択した場合は表示しません。

r15 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr15 オプションに相当します。 なお，本プロパティは，［共通オプション］タブの［レジスタ・モード］カテゴリの［レジスタ・モード］プロパティで［26 レジスタ・モード (-Xreg_mode=26)］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列

r16 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr16 オプションに相当します。 なお，本プロパティは，［共通オプション］タブの［レジスタ・モード］カテゴリの［レジスタ・モード］プロパティで［26 レジスタ・モード (-Xreg_mode=26)］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r17 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr17 オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r18 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr18 オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r19 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr19 オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r20 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr20 オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r21 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr21 オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r22 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr22 オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列

r23 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr23 オプションに相当します。 なお，本プロパティは，［共通オプション］タブの［レジスタ・モード］カテゴリの［レジスタ・モード］プロパティで［26 レジスタ・モード (-Xreg_mode=26)］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列
r24 レジスタに割り当てる外部変数	レジスタに割り付ける外部変数（“_”を除くシンボル名）を指定します。 cx コマンドの -Xr24 オプションに相当します。 なお，本プロパティは，［共通オプション］タブの［レジスタ・モード］カテゴリの［レジスタ・モード］プロパティで［26 レジスタ・モード (-Xreg_mode=26)］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1022 文字までの文字列

（12）［その他］

コンパイルに関するその他の詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。

<table>
<tr><td>コンパイル前に実行するコマンド</td><td>コンパイル処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%CompiledFile% ：コンパイル時の出力ファイルの絶対パスに置換します。
%InputFile% ：コンパイル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，コンパイル処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
<table>
<tr><td>デフォルト</td><td>コンパイル前に実行するコマンド[定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td rowspan="4">コンパイル後に実行するコマンド</td><td colspan="2">コンパイル処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%CompiledFile% ：コンパイル時の出力ファイルの絶対パスに置換します。
%InputFile% ：コンパイル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，コンパイル処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>コンパイル後に実行するコマンド[定義数]</td></tr>
<tr><td>変更方法</td><td>[...]ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td rowspan="4">その他の追加オプション</td><td colspan="2">その他に追加するコンパイル・オプションを入力します。
ここで設定したオプションは，コンパイル・オプション群の最後に付加します。</td></tr>
<tr><td>デフォルト</td><td>空欄</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

コア *n* 用オプション	コア *n* 用のコンパイル・オプションを指定します（*n*：“共通”，またはコア番号）。 本プロパティは，コアの数＋１個表示されます。 なお，本プロパティは，マルチコア対応デバイスを使用するプロジェクトの場合，かつ［共通オプション］タブの［ビルド方法］カテゴリの［コアごとにビルド・オプションを指定する］プロパティで［はい］を選択した場合のみ表示します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	259 文字までの文字列

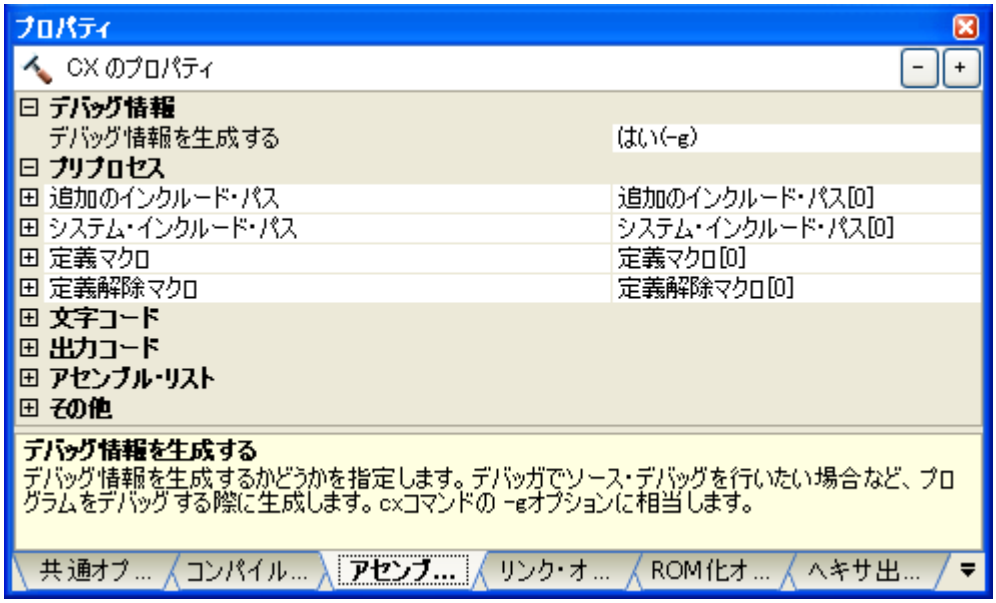
［アセンブル・オプション］タブ

本タブでは，アセンブル・フェーズに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

（1）［デバッグ情報］
（2）［プリプロセス］
（3）［文字コード］
（4）［出力コード］
（5）［アセンブル・リスト］
（6）［その他］

注意　本タブは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。

図 A―6　プロパティ パネル：［アセンブル・オプション］タブ



［各カテゴリの説明］

（1）［デバッグ情報］

デバッグ情報に関する詳細情報の表示，および設定の変更を行います。

デバッグ情報を生成する	デバッグ情報を生成するかどうかを選択します。 出力ファイル中にソース・デバッグ用の情報を出力することにより，デバッガでのソース・デバッグが可能となります。 cx コマンドの -g オプションに相当します。		
	デフォルト	はい (-g)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-g)	デバッグ情報を生成します。
		いいえ	デバッグ情報を生成しません。

(2) [プリプロセス]

プリプロセスに関する詳細情報の表示，および設定の変更を行います。

追加のインクルード・パス	アセンブル時の追加のインクルード・パスを指定します。 次のプレースホルダに対応しています。 %ActiveProjectDir% : アクティブ・プロジェクト・フォルダの絶対パスに置換します。 %ActiveProjectName% : アクティブ・プロジェクト名に置換します。 %BuildModeName% : ビルド・モード名に置換します。 %MainProjectDir% : メイン・プロジェクト・フォルダの絶対パスに置換します。 %MainProjectName% : メイン・プロジェクト名に置換します。 %MicomToolPath% : 本製品のインストール・フォルダの絶対パスに置換します。 %ProjectDir% : プロジェクト・フォルダの絶対パスに置換します。 %ProjectName% : プロジェクト名に置換します。 %TempDir% : テンポラリ・フォルダの絶対パスに置換します。 %WinDir% : Windows システム・フォルダの絶対パスに置換します。 指定したインクルード・パスは，CX の標準インクルード・ファイル・フォルダよりも優先して検索します。 パスはプロジェクト・フォルダを基点とします。 本プロパティを省略した場合は，CX の標準インクルード・ファイル・フォルダのみ検索します。 cx コマンドの -I オプションに相当します。 指定したインクルード・パスはサブプロパティとして表示します。 なお，プロジェクト・ツリーにインクルード・ファイルを追加すると，インクルード・パスをサブプロパティの一番最初に追加します。 インクルード・パスに大文字，小文字の区別はありません。		
	デフォルト	追加のインクルード・パス [*定義数*]	
	変更方法	[...] ボタンをクリックし，パス編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能	
	指定可能値	259 文字までの文字列 256 個まで指定可能です。	

<table>
<tr><td>システム・インクルード・パス</td><td colspan="2">アセンブル時にシステムが設定するインクルード・パスの指定順を変更します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
システム・インクルード・パスは，追加のインクルード・パスより低い優先度で検索します。
パスはプロジェクト・フォルダを基点とします。
cx コマンドの -I オプションに相当します。
インクルード・パスはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>システム・インクルード・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，システム・インクルード・パス順設定 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>変更不可（インクルード・パスの設定順の変更のみ可能）</td></tr>
<tr><td>定義マクロ</td><td colspan="2">定義したいマクロ名を指定します。
「マクロ名 = 定義値」の形式で 1 行に 1 つずつ指定します。
「= 定義値」の部分は省略可能で，省略した場合，定義値を 1 とします。
cx コマンドの -D オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>定義マクロ [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>256 文字までの文字列
256 個まで指定可能です。</td></tr>
<tr><td>定義解除マクロ</td><td colspan="2">定義解除したいマクロ名を指定します。
「マクロ名」の形式で 1 行に 1 つずつ指定します。
cx コマンドの -U オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>定義解除マクロ [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td>256 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

(3)［文字コード］

文字コードに関する詳細情報の表示，および設定の変更を行います。

文字コード	ソース・ファイル中の日本語のコメント，文字列に対して，使用する文字コードを選択します。 cx コマンドの -Xcharacter_set オプションに相当します。		
	デフォルト	SJIS(なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	SJIS(なし)	ソース・ファイル中の日本語の文字コードを SJIS と解釈します。
		EUC(-Xcharacter_set=euc_jp)	ソース・ファイル中の日本語の文字コードを EUC と解釈します。
		UFT-8(-Xcharacter_set=utf8)	ソース・ファイル中の日本語の文字コードを UFT-8 と解釈します。
		Big5(-Xcharacter_set=big5)	ソース・ファイル中の中国語の文字コードを繁体字中国語と解釈します。
		GB2312(-Xcharacter_set=gb2312)	ソース・ファイル中の中国語の文字コードを簡体字中国語と解釈します。
		処理しない (-Xcharacter_set=none)	ソース・ファイル中の日本語の文字コードを処理しません。

(4)［出力コード］

出力コードに関する詳細情報の表示，および設定の変更を行います。

sdata/sbss セクションに配置するデータ長の上限値 (バイト)	.sdata/.sbss セクションに配置するデータ長の上限サイズ（バイト）を指定します。 ただし，#pragma section 指令で .sdata/.sbss セクションを指定したデータは，そのサイズに関係なく .sdata/.sbss セクションに配置します。 cx コマンドの -Xsdata オプションに相当します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0 ～ 65535（10 進数）	
32 ビット分岐命令を使用する	jarl，および jr 命令に対して，far jump 機能を使用するかどうかを選択します。 far jump 機能を使用することにより，jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。 cx コマンドの -Xasm_far_jump【V850E2】【V850E2V3】オプションに相当します。 なお，本プロパティは，V850E2 コア，および V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xasm_far_jump)	jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。
		いいえ	jarl，および jr 命令としてアセンブルを行います。

(5)［アセンブル・リスト］

アセンブル・リストに関する詳細情報の表示，および設定の変更を行います。

アセンブル・リスト・ファイルを出力する	アセンブル・リスト・ファイルを出力するかどうかを選択します。 cx コマンドの -Xprn_path オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xprn_path)	アセンブル・リスト・ファイルを出力します。
		いいえ	アセンブル・リスト・ファイルを出力しません。
アセンブル・リスト・ファイル出力フォルダ	アセンブル・リスト・ファイルの出力先フォルダを指定します。 アセンブル・リスト・ファイルは，ソース・ファイル名の拡張子を .prn で置き換えた名前で出力します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 　%BuildModeName%　：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xprn_path オプションに相当します。 なお，本プロパティは，［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択した場合のみ表示します。		
	デフォルト	%BuildModeName%	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集	
	指定可能値	247 文字までの文字列	

(6)［その他］

アセンブルに関するその他の詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。

<table>
<tr><td>アセンブル前に実行するコマンド</td><td>
アセンブル処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%AssembledFile% ：アセンブル時の出力ファイルの絶対パスに置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%InputFile% ：アセンブル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，アセンブル処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
<table>
<tr><td>デフォルト</td><td>アセンブル前に実行するコマンド[定義数]</td></tr>
<tr><td>変更方法</td><td>[...]ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td rowspan="5">アセンブル後に実行するコマンド</td><td colspan="2">アセンブル処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%AssembledFile% ：アセンブル時の出力ファイルの絶対パスに置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%InputFile% ：アセンブル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，アセンブル処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>アセンブル後に実行するコマンド[定義数]</td></tr>
<tr><td>変更方法</td><td>[...]ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td rowspan="4">その他の追加オプション</td><td colspan="2">その他に追加するアセンブル・オプションを入力します。
ここで設定したオプションは，アセンブル・オプション群の最後に付加します。</td></tr>
<tr><td>デフォルト</td><td>空欄</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

コア *n* 用オプション	コア *n* 用のアセンブル・オプションを指定します（*n*：“共通”，またはコア番号）。 本プロパティは，コアの数＋１個表示されます。 なお，本プロパティは，マルチコア対応デバイスを使用するプロジェクトの場合，かつ［共通オプション］タブの［ビルド方法］カテゴリの［コアごとにビルド・オプションを指定する］プロパティで［はい］を選択した場合のみ表示します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	259 文字までの文字列

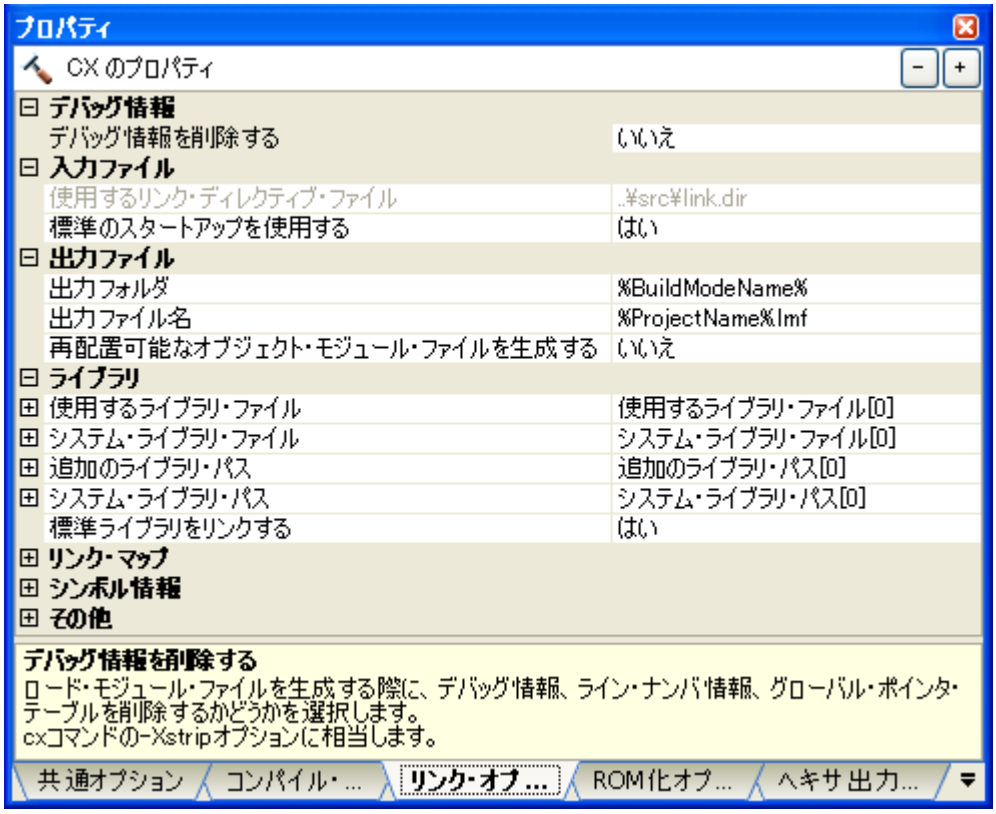
［リンク・オプション］タブ

本タブでは，リンク・フェーズに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

（1）［デバッグ情報］
（2）［入力ファイル］
（3）［出力ファイル］
（4）［ライブラリ］
（5）［リンク・マップ］
（6）［シンボル情報］
（7）［その他］

注意　本タブは，ライブラリ用のプロジェクトの場合は表示しません。

図 A—7　プロパティ パネル：［リンク・オプション］タブ



[各カテゴリの説明]

(1) [デバッグ情報]

デバッグ情報に関する詳細情報の表示，および設定の変更を行います。

デバッグ情報を削除する	ロード・モジュール・ファイルを生成する際に，デバッグ情報，ライン・ナンバ情報，グローバル・ポインタ・テーブルを削除するかどうかを選択します。 cx コマンドの -Xstrip オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xstrip)	ロード・モジュール・ファイルを生成する際に，デバッグ情報，ライン・ナンバ情報，グローバル・ポインタ・テーブルを削除します。
		いいえ	ロード・モジュール・ファイルを生成する際に，デバッグ情報，ライン・ナンバ情報，グローバル・ポインタ・テーブルを削除しません。

(2) [入力ファイル]

入力ファイルに関する詳細情報の表示，および設定の変更を行います。

使用するリンク・ディレクティブ・ファイル	リンクに使用するリンク・ディレクティブ・ファイルの名前を表示します。 プロジェクトに登録している有効なリンク・ディレクティブ・ファイルを検索して使用します。 cx コマンドの -Xlink_directive オプションに相当します。		
	デフォルト	*プロジェクトに登録しているリンク・ディレクティブ・ファイル名*	
	変更方法	変更不可	
標準のスタートアップを使用する	標準的なスタートアップ・ルーチンが書かれている CX 付属のオブジェクト・モジュール・ファイル（cstart.obj，または cstartN.obj）をリンクするかどうかを選択します。 [ROM 化オプション] タブの [出力ファイル] カテゴリの [ROM 化用ロード・モジュール・ファイルを出力する] プロパティで [はい] を選択した場合は cstart.obj，[いいえ (-Xno_romize)] を選択した場合は cstartN.obj をリンクします。 cx コマンドの -Xno_startup オプションに相当します。 なお，本プロパティは，プロジェクト・ツリーのスタートアップ・ノード直下にビルド対象ファイルを登録している場合，またはファイル・ノードに C ソース・ファイルを 1 つも登録していない場合は表示しません。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	標準スタートアップ・ルーチン（cstart.obj，または cstartN.obj）をリンクします。
		いいえ (-Xno_startup)	標準スタートアップ・ルーチンをリンクしません。

(3)［出力ファイル］

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td>出力フォルダ</td><td colspan="3">生成するロード・モジュールの出力先フォルダを指定します。
相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。
絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。
次のプレースホルダに対応しています。
　%BuildModeName%　：ビルド・モード名に置換します。
空欄の場合は，プロジェクト・フォルダを指定したものとみなします。
cx コマンドの -o オプションに相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">%BuildModeName%</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td colspan="2">247 文字までの文字列</td></tr>
<tr><td>出力ファイル名</td><td colspan="3">生成するロード・モジュールのファイル名を指定します。
.lmf 以外の拡張子を指定することはできません。
拡張子を省略した場合は，.lmf を自動的に付加します。
ターゲットがマルチコア CPU の場合は，共通部用ロード・モジュールとコア n 用ロード・モジュールを生成し，それらを元に最終ロード・モジュールを生成します（n：ターゲット CPU が持つコアの数）。
　共通部用ロード・モジュール：拡張子を除いた入力文字列 _cmn.lmf
　コア n 用ロード・モジュール：拡張子を除いた入力文字列 _pen.lmf
次のプレースホルダに対応しています。
　%ActiveProjectName%　：アクティブ・プロジェクト名に置換します。
　%MainProjectName%　：メイン・プロジェクト名に置換します。
　%ProjectName%　：プロジェクト名に置換します。
cx コマンドの -o オプションに相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">%ProjectName%.lmf</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力</td></tr>
<tr><td></td><td>指定可能値</td><td colspan="2">259 文字までの文字列</td></tr>
<tr><td>再配置可能なオブジェクト・モジュール・ファイルを生成する</td><td colspan="3">再配置可能なオブジェクト・モジュール・ファイルを生成するかどうかを選択します。
cx コマンドの -Xrelinkable_object オプションに相当します。
なお，本プロパティは，［共通オプション］タブの［フラッシュ対応］カテゴリの［ブート－フラッシュの再リンク機能を利用する］プロパティで［いいえ］を選択した場合のみ表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td></td><td rowspan="2">指定可能値</td><td>はい (-Xrelinkable_object)</td><td>再配置可能なオブジェクト・モジュール・ファイルを生成します。</td></tr>
<tr><td>いいえ</td><td>実行可能なオブジェクト・モジュール・ファイルを生成します。</td></tr>
</table>

(4) [ライブラリ]

ライブラリに関する詳細情報の表示，および設定の変更を行います。

使用するライブラリ・ファイル	標準ライブラリ以外の使用するライブラリ・ファイル（lib*xxx*.lib，または lib*xxx*.a）を指定します。 lib*xxx*.lib を優先して検索して，見つからなかった場合に lib*xxx*.a を検索します。 *xxx* のみを指定してください（例：“user”と指定すると，libuser.lib，または libuser.a を指定したものとみなします）。 1 行に 1 ファイルずつ指定します。 ライブラリ・ファイルはライブラリ・パスから検索します。 cx コマンドの -l オプションに相当します。 指定したライブラリ・ファイル名はサブプロパティとして表示します。	
	デフォルト	使用するライブラリ・ファイル [*定義数*]
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	249 文字までの文字列 256 個まで指定可能です。
システム・ライブラリ・ファイル	システムが使用するライブラリ・ファイルの名前を表示します。 システム・ライブラリ・ファイルは，使用するライブラリ・ファイルより低い優先度で検索します。 cx コマンドの -l オプションに相当します。 ライブラリ・ファイル名はサブプロパティとして表示します。	
	デフォルト	システム・ライブラリ・ファイル [*定義数*]
	変更方法	変更不可

<table>
<tr><td>追加のライブラリ・パス</td><td>標準ライブラリ以外の使用するライブラリ・ファイルの検索フォルダを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
相対パスを指定した場合は，プロジェクト・フォルダを基点とします。
cx コマンドの -L オプションに相当します。
指定したライブラリ・パス名はサブプロパティとして表示します。
<table><tr><td>デフォルト</td><td>追加のライブラリ・パス [定義数]</td></tr><tr><td>変更方法</td><td>[...] ボタンをクリックし，パス編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr><tr><td>指定可能値</td><td>259 文字までの文字列
256 個まで指定可能です。</td></tr></table></td></tr>
<tr><td>システム・ライブラリ・パス</td><td>システム・ライブラリ・ファイルの検索フォルダを表示します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
相対パスを表示している場合は，プロジェクト・フォルダを基点とします。
cx コマンドの -L オプションに相当します。
ライブラリ・パス名はサブプロパティとして表示します。
<table><tr><td>デフォルト</td><td>システム・ライブラリ・パス [定義数]</td></tr><tr><td>変更方法</td><td>変更不可</td></tr></table></td></tr>
</table>

標準ライブラリをリンクする	標準ライブラリをリンクするかどうかを選択します。 cx コマンドの -Xno_stdlib オプションに相当します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	標準ライブラリをリンクします。
		いいえ (-Xno_stdlib)	標準ライブラリをリンクしません。

(5)［リンク・マップ］

リンク・マップに関する詳細情報の表示，および設定の変更を行います。

リンク・マップ・ファイルを出力する	リンク・マップ・ファイルを出力するかどうかを選択します。 cx コマンドの -Xmap オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xmap)	リンク・マップ・ファイルを出力します。
		いいえ	リンク・マップ・ファイルを出力しません。
リンク・マップ・ファイル出力フォルダ	リンク・マップ・ファイルの出力先フォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 　%BuildModeName%　：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xmap オプションに相当します。 なお，本プロパティは，［リンク・マップ・ファイルを出力する］プロパティで［はい (-Xmap)］を選択した場合のみ表示します。		
	デフォルト	%BuildModeName%	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集	
	指定可能値	247 文字までの文字列	

リンク・マップ・ファイル名	リンク・マップ・ファイル名を指定します。 .map 以外の拡張子を指定することはできません。 拡張子を省略した場合は，.map を自動的に付加します。 次のプレースホルダに対応しています。 %ActiveProjectName% ：アクティブ・プロジェクト名に置換します。 %MainProjectName% ：メイン・プロジェクト名に置換します。 %ProjectName% ：プロジェクト名に置換します。 空欄の場合は，%ProjectName%.map を指定したものとみなします。 cx コマンドの -Xmap オプションに相当します。 なお，本プロパティは，［リンク・マップ・ファイルを出力する］プロパティで［はい (-Xmap)］を選択した場合のみ表示します。		
	デフォルト	%ProjectName%.map	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	259 文字までの文字列	
リンク・マップ・ファイルへシンボル情報を出力する	リンク・マップ・ファイルへシンボル情報を出力するかどうかを選択します。 cx コマンドの -Xsymbol_dump オプションに相当します。 なお，本プロパティは，［リンク・マップ・ファイルを出力する］プロパティで［はい (-Xmap)］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xsymbol_dump)	リンク・マップ・ファイルへシンボル情報を出力します。
		いいえ	リンク・マップ・ファイルへシンボル情報を出力しません。

(6)［シンボル情報］

シンボル情報に関する詳細情報の表示，および設定の変更を行います。

シンボル情報ファイルを出力する	ビルド時にシンボル情報ファイルを出力するかどうかを選択します。 マルチコア対応デバイスを使用している場合，本プロパティを設定することはできません。 cx コマンドの -Xsfg オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xsfg)	シンボル情報ファイルを出力します。
		いいえ	シンボル情報ファイルを出力しません。

<table>
<tr><td rowspan="4">シンボル情報ファイル出力フォルダ</td><td colspan="3">シンボル情報ファイルの出力先フォルダを指定します。
相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。
絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。
次のプレースホルダに対応しています。
%BuildModeName% ： ビルド・モード名に置換します。
空欄の場合は，プロジェクト・フォルダを指定したものとみなします。
cx コマンドの -Xsfg オプションに相当します。
なお，本プロパティは，［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">%BuildModeName%</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td colspan="2">247 文字までの文字列</td></tr>
<tr><td rowspan="4">シンボル情報ファイル名</td><td colspan="3">シンボル情報ファイル名を指定します。
.sfg 以外の拡張子を指定することはできません。
拡張子を省略した場合は，.sfg を自動的に付加します。
次のプレースホルダに対応しています。
%ActiveProjectName% ： アクティブ・プロジェクト名に置換します。
%MainProjectName% ： メイン・プロジェクト名に置換します。
%ProjectName% ： プロジェクト名に置換します。
空欄の場合は，プロジェクト・ファイル名の拡張子を .sfg で置き換えた名前で出力します。
cx コマンドの -Xsfg オプションに相当します。
なお，本プロパティは，［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">%ProjectName%.sfg</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td colspan="2">259 文字までの文字列</td></tr>
<tr><td rowspan="5">最適な配置情報を出力する</td><td colspan="3">変数をセクション単位で利用頻度の高い順にソートして，.tidata.byte，.tidata.word，.sidata，.sedata，.sdata セクションに配置可能な分を判断して，最適な配置情報をシンボル情報ファイルに出力するかどうかを選択します。
cx コマンドの -Xsfg_opt オプションに相当します。
なお，本プロパティは，［シンボル情報ファイルを出力する］プロパティで［はい (-Xsfg)］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい (-Xsfg_opt)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xsfg_opt)</td><td>変数をセクション単位で利用頻度の高い順にソートして，最適な配置情報をシンボル情報ファイルに出力します。</td></tr>
<tr><td>いいえ</td><td>変数を利用頻度の高い順にソートして，配置情報をシンボル情報ファイルに出力します。</td></tr>
</table>

.tidata セクションのサイズ	.tidata セクションに変数を配置するサイズの上限を指定します。 cx コマンドの -Xsfg_size_tidata オプションに相当します。 なお，本プロパティは，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択した場合のみ表示します。	
	デフォルト	256（10 進数）
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ 256（10 進数）
.tidata.byte セクションのサイズ	.tidata.byte セクションに変数を配置するサイズの上限を指定します。 cx コマンドの -Xsfg_size_tidata_byte オプションに相当します。 なお，本プロパティは，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択した場合のみ表示します。	
	デフォルト	128（10 進数）
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ 128（10 進数）
.sidata セクションのサイズ	.sidata セクションに変数を配置するサイズの上限を指定します。 cx コマンドの -Xsfg_size_sidata オプションに相当します。 なお，本プロパティは，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択した場合のみ表示します。	
	デフォルト	32512（10 進数）
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ 32512（10 進数）
.sedata セクションのサイズ	.sedata セクションに変数を配置するサイズの上限を指定します。 cx コマンドの -Xsfg_size_sedata オプションに相当します。 なお，本プロパティは，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択した場合のみ表示します。	
	デフォルト	32768（10 進数）
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ 32768（10 進数）
.sdata セクションのサイズ	.sdata セクションに変数を配置するサイズの上限を指定します。 cx コマンドの -Xsfg_size_sdata オプションに相当します。 なお，本プロパティは，［最適な配置情報を出力する］プロパティで［はい (-Xsfg_opt)］を選択した場合のみ表示します。	
	デフォルト	65536（10 進数）
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ 65536（10 進数）

(7) ［その他］

リンクに関するその他の詳細情報の表示，および設定の変更を行います。

ユーザ・オプション・バイトを設定する	ユーザ・オプション・バイトを設定するかどうかを選択します。 cx コマンドの -Xoption_byte オプションに相当します。 なお，本プロパティは，ユーザ・オプション・バイト機能を持たないデバイスを使用するプロジェクトの場合は表示しません。		
	デフォルト	はい (デバイスの初期値を使用)(なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (デバイスの初期値を使用)(なし)	デバイス・ファイルの初期値をユーザ・オプション・バイト領域に埋め込みます。
		いいえ (-Xoption_byte=none)	ユーザ・オプション・バイト領域の生成を抑止します。
エントリ・シンボル	生成するロード・モジュール・ファイルのエントリ・ポイント・アドレスとして設定するシンボルを指定します。 空欄の場合は，以下の順序でエントリ・ポイント・アドレスを決定します。 (1) シンボル __start のアドレス (2) __start が存在しない場合は，生成するロード・モジュール・ファイル内の最下位に割り付けられた text 属性のセクションの先頭アドレス (3) アドレス 0 cx コマンドの -Xentry_address オプションに相当します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集	
	指定可能値	1022 文字までの文字列	
レジスタ・モードをチェックする	すべての入力オブジェクト・モジュール・ファイルに対して，異なるレジスタ・モードの混在をチェックし，詳細情報を表示するかどうかを選択します。 cx コマンドの -Xregmode_info オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xregmode_info)	異なるレジスタ・モードの混在をチェックし，詳細情報を表示します。
		いいえ	異なるレジスタ・モードの混在をチェックし，詳細情報を表示しません。
強制リンクを行う	メモリのオーバフロー時に，リンク処理を続行するかどうかを選択します。 メモリの過不足情報は出力 パネルに表示します。 なお，強制リンクを行っても最終ファイルは生成しません。 cx コマンドの -Xforce_link オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xforce_link)	メモリのオーバフロー時に，リンク処理を続行します。
		いいえ	メモリのオーバフロー時に，リンク処理を終了します。

GP 情報を表示する	［コンパイル・オプション］タブの［出力コード］カテゴリの［sdata/sbss セクションに配置するデータ長の上限値 (バイト)］プロパティの数値設定において目安として用いるための情報を出力 パネルに表示するかどうかを選択します。 cx コマンドの -Xsdata_info オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xsdata_info)	［sdata/sbss セクションに配置するデータ長の上限値 (バイト)］プロパティの数値設定において目安として用いるための情報を表示します。
		いいえ	［sdata/sbss セクションに配置するデータ長の上限値 (バイト)］プロパティの数値設定において目安として用いるための情報を表示しません。
2 パス・モードでリンクする	2 パス・モードでリンクするかどうかを選択します。 2 パス・モードは 1 パス・モードよりも処理が低速ですが，より大きなサイズのファイルを処理することができます。 また，ホールの充てん値を指定することができます。 cx コマンドの -Xtwo_pass_link オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xtwo_pass_link)	2 パス・モードでリンクを行います。
		いいえ	1 パス・モードでリンクを行います。
ホールの充てん値	生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値を指定します。 空欄の場合は，0000（16 進数）を指定したものとみなします。 cx コマンドの -Xalign_fill オプションに相当します。 なお，本プロパティは，［2 パス・モードでリンクする］プロパティで［はい (-Xtwo_pass_link)］を選択した場合のみ表示します。		
	デフォルト	0000（16 進数）	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0000 ～ FFFF（4 桁以下の 16 進数）	

<table>
<tr><td rowspan="5">リロケーション不正を無視する</td><td colspan="3">リロケーション処理において，以下の不正箇所があった場合，エラーとはせず，警告メッセージを出力してリンクの処理を続行するかどうかを選択します。
- 未解決な外部参照のアドレス計算結果が不正
- 配置するセクションとの関係が不正
cx コマンドの -Xignore_address_error オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xignore_address_error)</td><td>リロケーション処理において，不正箇所があった場合，警告メッセージを出力してリンクの処理を続行します。</td></tr>
<tr><td>いいえ</td><td>リロケーション処理において，不正箇所があった場合，エラーとします。</td></tr>
<tr><td rowspan="5">すべての多重定義シンボルを検出する</td><td colspan="3">多重定義されたすべての外部シンボルに対してメッセージを出力して，リンク処理を中止するかどうかを選択します。
cx コマンドの -Xmultiple_symbol オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xmultiple_symbol)</td><td>多重定義されたすべての外部シンボルに対してメッセージを出力して，リンク処理を中止します。</td></tr>
<tr><td>いいえ</td><td>多重定義された最初の外部シンボルに対してメッセージを出力して，リンクの処理を中止します。</td></tr>
<tr><td rowspan="5">外部シンボル不正をチェックする</td><td colspan="3">外部シンボルをリンクする際，シンボルのサイズ，および整列条件の不正をチェックするかどうかを選択します。
cx コマンドの -Xlink_check_off オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい</td><td>外部シンボルをリンクする際，シンボルのサイズ，および整列条件の不正をチェックします。</td></tr>
<tr><td>いいえ (-Xlink_check_off=symbol)</td><td>外部シンボルをリンクする際，シンボルのサイズ，および整列条件の不正をチェックしません。</td></tr>
</table>

未定義外部シンボル不正をチェックする	未定義外部シンボルをリンクする際，シンボルのサイズ，および整列条件の不正をチェックするかどうかを選択します。 cx コマンドの -Xlink_check_off オプションに相当します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	未定義外部シンボルをリンクする際，シンボルのサイズ，および整列条件の不正をチェックします。
		いいえ (-Xlink_check_off=undefined)	未定義外部シンボルをリンクする際，シンボルのサイズ，および整列条件の不正をチェックしません。
内蔵 ROM 領域への配置をチェックする	内蔵 ROM 領域への配置に対して，チェックを行うかどうかを選択します。 ROM レス・モード使用時は，［いいえ (-Xlink_check_off=irom)］を選択してください。 cx コマンドの -Xlink_check_off オプションに相当します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	内蔵 ROM 領域への配置に対して，チェックを行います。
		いいえ (-Xlink_check_off=irom)	内蔵 ROM 領域への配置に対して，チェックを行いません。
ライブラリ・ファイルを再スキャンする	［ライブラリ］カテゴリの［使用するライブラリ・ファイル］，および［システム・ライブラリ・ファイル］プロパティで指定したライブラリ・ファイルの再スキャンを行うかどうかを選択します。 本プロパティを指定すると，ライブラリのリンク順によるシンボル未解決を防ぐことができます。 cx コマンドの -Xrescan オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xrescan)	使用するライブラリ・ファイルの再スキャンを行います。
		いいえ	使用するライブラリ・ファイルの再スキャンを行いません。

<table>
<tr><td>リンク前に実行するコマンド</td><td>
リンク処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%LinkedFile% ：リンク処理時の出力ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，リンク処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
なお，本プロパティは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。
<table>
<tr><td>デフォルト</td><td>リンク前に実行するコマンド [定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

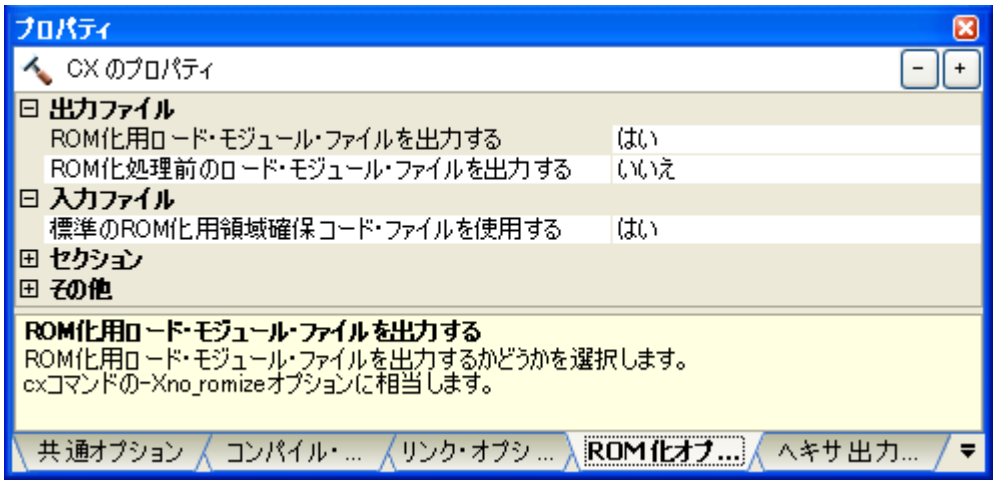
<table>
<tr><td>リンク後に実行するコマンド</td><td>リンク処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%LinkedFile% ：リンク処理時の出力ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，リンク処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
なお，本プロパティは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。
<table><tr><td>デフォルト</td><td>リンク後に実行するコマンド [定義数]</td></tr><tr><td>変更方法</td><td>［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr><tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr></table></td></tr>
<tr><td>その他の追加オプション</td><td>その他に追加するリンク・オプションを入力します。
ここで設定したオプションは，リンク・オプション群の最後に付加します。
なお，本プロパティは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。
<table><tr><td>デフォルト</td><td>空欄</td></tr><tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr><tr><td>指定可能値</td><td>259 文字までの文字列</td></tr></table></td></tr>
</table>

[ROM 化オプション] タブ

本タブでは，ROM 化フェーズに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

注意　本タブは，ライブラリ用のプロジェクトの場合は表示しません。

図 A―8　プロパティ パネル：[ROM 化オプション] タブ



[各カテゴリの説明]

（1）[出力ファイル]

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

ROM 化用ロード・モジュール・ファイルを出力する	ROM 化用ロード・モジュール・ファイルを出力するかどうかを選択します。 cx コマンドの -Xno_romize オプションに相当します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ROM 化用ロード・モジュール・ファイルを出力します。
		いいえ (-Xno_romize)	ROM 化用ロード・モジュール・ファイルを出力しません。

ROM 化処理前のロード・モジュール・ファイルを出力する	ROM 化処理前のロード・モジュール・ファイルを出力するかどうかを選択します。 cx コマンドの -Xlink_output オプションに相当します。 なお，本プロパティは，［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ROM 化処理前のロード・モジュール・ファイルを出力します。 ［リンク・オプション］タブの［出力ファイル］カテゴリの［出力ファイル名］プロパティで指定したファイル名に _NonROMize を付加した名前で出力します。
		いいえ	ROM 化処理前のロード・モジュール・ファイルを出力しません。

(2)［入力ファイル］

入力ファイルに関する詳細情報の表示，および設定の変更を行います。

標準の ROM 化用領域確保コード・ファイルを使用する	標準の ROM 化用領域確保コード・ファイル（rompcrt.obj）を使用するかどうかを選択します。 なお，本プロパティは，［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合のみ表示します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	標準の ROM 化用領域確保コード・ファイルを使用します。
		いいえ	標準の ROM 化用領域確保コード・ファイルを使用しません。 ROM 化用領域確保コード・ファイルを作成して，［ROM 化用領域確保コード・ファイル名］プロパティにファイル名を指定してください。
ROM 化用領域確保コード・ファイル名	ROM 化用領域確保コード・ファイル名を指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 空欄の場合は，リンク時にエラーとなるため，本プロパティは必ず指定してください。 cx コマンドの -Xrompcrt オプションに相当します。 なお，本プロパティは，［標準の ROM 化用領域確保コード・ファイルを使用する］プロパティで［いいえ］を選択した場合のみ表示します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，ROM 化用領域確保コード・ファイルを指定 ダイアログによる編集	
	指定可能値	259 文字までの文字列	

(3)［セクション］

セクションに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合のみ表示します。

rompsec セクションの開始シンボル	生成する rompsec セクションの先頭アドレスとして用いるシンボルを指定します。 空欄の場合は，__S_romp を指定したものとみなします。 cx コマンドの -Xrompsec_start オプションに相当します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集	
	指定可能値	1022 文字までの文字列	
rompsec セクションのみのロード・モジュール・ファイルを生成する	生成するロード・モジュール・ファイル中に text 属性を持つセクションを入れずに，rompsec セクションのみを持つロード・モジュール・ファイルを生成するかどうかを選択します。 cx コマンドの -Xrompsec_only オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xrompsec_only)	rompsec セクションのみを持つロード・モジュール・ファイルを生成します。
		いいえ	生成するロード・モジュール・ファイル中に text 属性を持つセクションも入れます。
rompsec セクションに含めるデータ・セクション	rompsec セクションに含めるデータ・セクションを指定します。 1 行に 1 つずつ指定します。 指定した各データ・セクションを上から順に rompsec セクションに含めます。 cx コマンドの -Xrompsec_data オプションに相当します。 指定したセクション名はサブプロパティとして表示します。		
	デフォルト	rompsec セクションに含めるデータ・セクション [*指定数*]	
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能	
	指定可能値	1022 文字までの文字列 1024 個まで指定可能です。	
rompsec セクションに含めるテキスト・セクション	rompsec セクションに含めるテキスト・セクションを指定します。 1 行に 1 つずつ指定します。 指定した各テキスト・セクションを上から順に rompsec セクションに含めます。 cx コマンドの -Xrompsec_text オプションに相当します。 指定したセクション名はサブプロパティとして表示します。		
	デフォルト	rompsec セクションに含めるテキスト・セクション [*指定数*]	
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能	
	指定可能値	1022 文字までの文字列 1024 個まで指定可能です。	

(4)［その他］

ROM 化に関するその他の詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］プロパティで［はい］を選択した場合のみ表示します。

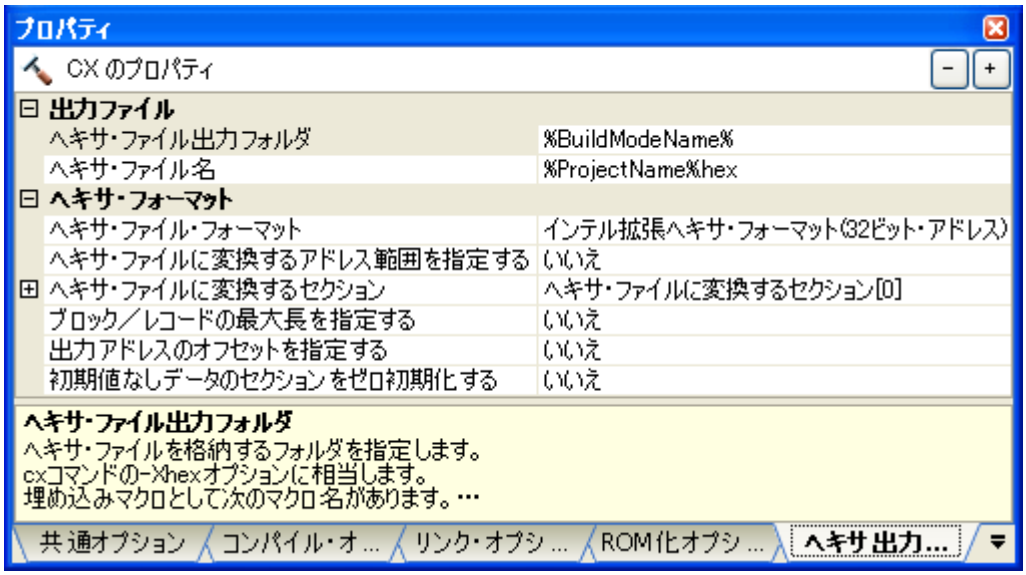
<table>
<tr><td rowspan="6">ROM 化時のエラーを無視する</td><td colspan="3">rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行うかどうか，入力ファイル，および出力ファイルのアドレスの重複チェックを行うかどうかを選択します。
cx コマンドの -Xromize_check_off オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ (なし)</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="4">指定可能値</td><td>はい (すべて無視)(-Xromize_check_off=rom_less,address)</td><td>内蔵 ROM 周辺の配置エラー・チェック，アドレスの重複チェックを行いません。</td></tr>
<tr><td>はい (内蔵 ROM 周辺の配置エラーを無視)(-Xromize_check_off=rom_less)</td><td>rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行いません。
本項目は，ROM レス・モード使用時に指定します。</td></tr>
<tr><td>はい (アドレスの重複を無視)(-Xromize_check_off=address)</td><td>入力ファイル，および出力ファイルのアドレスの重複チェックを行いません。</td></tr>
<tr><td>いいえ (なし)</td><td>内蔵 ROM 周辺の配置エラー・チェック，アドレスの重複チェックを行います。</td></tr>
</table>

［ヘキサ出力オプション］タブ

本タブでは，ヘキサ出力フェーズに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

注意　本タブは，ライブラリ用のプロジェクトの場合は表示しません。

図 A―9　プロパティ パネル：［ヘキサ出力オプション］タブ



［各カテゴリの説明］

（1）［出力ファイル］

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td rowspan="4">ヘキサ・ファイル出力フォルダ</td><td colspan="2">ヘキサ・ファイルの出力先フォルダを指定します。
相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。
絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。
次のプレースホルダに対応しています。
　%BuildModeName%　：ビルド・モード名に置換します。
空欄の場合は，プロジェクト・フォルダを指定したものとみなします。
cx コマンドの -Xhex オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td>%BuildModeName%</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>247 文字までの文字列</td></tr>
<tr><td rowspan="4">ヘキサ・ファイル名</td><td colspan="2">ヘキサ・ファイル名を指定します。
拡張子は自由に指定可能です。
次のプレースホルダに対応しています。
　%ActiveProjectName%　：アクティブ・プロジェクト名に置換します。
　%MainProjectName%　：メイン・プロジェクト名に置換します。
　%ProjectName%　：プロジェクト名に置換します。
空欄の場合は，%ProjectName%.hex を指定したものとみなします。
cx コマンドの -Xhex オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td>%ProjectName%.hex</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

(2)［ヘキサ・フォーマット］

ヘキサ・フォーマットに関する詳細情報の表示，および設定の変更を行います。

ヘキサ・ファイル・フォーマット	出力するヘキサ・ファイルのフォーマットを選択します。 cx コマンドの -Xhex_format オプションに相当します。		
	デフォルト	インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(なし)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	インテル拡張ヘキサ・フォーマット (-Xhex_format=I)	ヘキサ・ファイルをインテル拡張ヘキサ・フォーマットとします。
		インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(なし)	ヘキサ・ファイルをインテル拡張ヘキサ・フォーマット（32 ビット・アドレス）とします。
		モトローラ S タイプ・フォーマット (スタンダード・アドレス)(-Xhex_format=S)	ヘキサ・ファイルをモトローラ S タイプ・フォーマット（スタンダード・アドレス）とします。
		モトローラ S タイプ・フォーマット (32 ビット・アドレス)(-Xhex_format=s)	ヘキサ・ファイルをモトローラ S タイプ・フォーマット（32 ビット・アドレス）とします。
		拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)	ヘキサ・ファイルを拡張テクトロニクス・ヘキサ・フォーマットとします。
ヘキサ・ファイルに変換するアドレス範囲を指定する	ヘキサ・ファイルに変換するアドレス範囲を指定するかどうかを選択します。 cx コマンドの -Xhex_fill オプションに相当します。 なお，本プロパティは，［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合は表示しません。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xhex_fill)	ヘキサ・ファイルに変換するアドレス範囲を指定します。
		いいえ	ヘキサ・ファイルに変換するアドレス範囲を指定しません。
充てん値	ヘキサ・ファイルに変換する領域のうち，未使用領域の充てん値を指定します。 空欄の場合は，FF（16 進数）を指定したものとみなします。 cx コマンドの -Xhex_fill オプションに相当します。 なお，本プロパティは，［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合，および［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［いいえ］を選択した場合は表示しません。		
	デフォルト	FFFF（16 進数）	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	00 ～ FFFF（2 桁，または 4 桁の 16 進数）	

開始アドレス	ヘキサ・ファイルに変換する領域の開始アドレスを指定します。 本プロパティを指定する場合は，［サイズ］プロパティの指定も必要です。 どちらか一方が空欄の場合は，デバイス・ファイルで定義された内蔵 ROM 領域のすべてのコードをヘキサ・ファイルに変換します。 cx コマンドの -Xhex_fill オプションに相当します。 なお，本プロパティは，［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合，および［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［いいえ］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ FFFFFFFF（16 進数）
サイズ	ヘキサ・ファイルに変換する領域のサイズを指定します。 本プロパティを指定する場合は，［開始アドレス］プロパティの指定も必要です。 どちらか一方が空欄の場合は，デバイス・ファイルで定義された内蔵 ROM 領域のすべてのコードをヘキサ・ファイルに変換します。 cx コマンドの -Xhex_fill オプションに相当します。 なお，本プロパティは，［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合，および［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［いいえ］を選択した場合は表示しません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1 ～ 100000000（16 進数）
ヘキサ・ファイルに変換するセクション	ヘキサ・ファイルに変換するセクションを指定します。 1 行に 1 つずつ指定します。 本プロパティの設定を省略した場合は，NOBITS 以外のセクション・タイプとセクション属性 A を持つすべてのセクションをヘキサ・ファイルに変換します。 cx コマンドの -Xhex_section オプションに相当します。 指定したセクションはサブプロパティとして表示します。 なお，本プロパティは，［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合，または［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［いいえ］を選択した場合のみ表示します。	
	デフォルト	ヘキサ・ファイルに変換するセクション [*設定数*]
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	1022 文字までの文字列 1024 個まで指定可能です。

ブロック／レコードの最大長を指定する	ブロック／レコードの最大長を指定するかどうかを選択します。 cx コマンドの -Xhex_block_size オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xhex_block_size)	ブロック／レコードの最大長を指定します。
		いいえ	デフォルト値をブロック／レコードの最大長とします。

<table>
<tr><td rowspan="4">ブロック／レコードの最大長</td><td colspan="2">ブロック／レコードの最大長を指定します。
cx コマンドの -Xhex_block_size オプションに相当します。
なお，本プロパティは，［ブロック／レコードの最大長を指定する］プロパティで［はい (-Xhex_block_size)］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td>［ヘキサ・ファイル・フォーマット］プロパティで選択する項目によって異なります。
- ［インテル拡張ヘキサ・フォーマット (なし)］，または［インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(-Xhex_format=i)］を選択したのち，本プロパティのコンテキスト・メニューの［デフォルトに戻す］を選択した場合
32
- ［モトローラ S タイプ・フォーマット (スタンダード・アドレス)(-Xhex_format=S)］を選択したのち，本プロパティのコンテキスト・メニューの［デフォルトに戻す］を選択した場合
80
- ［モトローラ S タイプ・フォーマット (32 ビット・アドレス)(-Xhex_format=s)］を選択したのち，本プロパティのコンテキスト・メニューの［デフォルトに戻す］を選択した場合
80
- ［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択したのち，本プロパティのコンテキスト・メニューの［デフォルトに戻す］を選択した場合
255
なお，［ヘキサ・ファイル・フォーマット］プロパティの変更時に，本プロパティの設定値が指定可能値の範囲外となる場合は，ただちに上記のデフォルト値に設定します。</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td>［ヘキサ・ファイル・フォーマット］プロパティで選択する項目によって異なります。
- ［インテル拡張ヘキサ・フォーマット (なし)］，または［インテル拡張ヘキサ・フォーマット (32 ビット・アドレス)(-Xhex_format=i)］を選択した場合
1 ～ 255（10 進数），または 01 ～ FF（16 進数）
- ［モトローラ S タイプ・フォーマット (スタンダード・アドレス)(-Xhex_format=S)］を選択した場合
1 ～ 251（10 進数），または 01 ～ FB（16 進数）
- ［モトローラ S タイプ・フォーマット (32 ビット・アドレス)(-Xhex_format=s)］を選択した場合
1 ～ 250（10 進数），または 01 ～ FA（16 進数）
- ［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合
16 ～ 255（10 進数），または 10 ～ FF（16 進数）</td></tr>
</table>

出力アドレスのオフセットを指定する	出力するアドレスのオフセットを指定するかどうかを選択します。 cx コマンドの -Xhex_offset オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xhex_offset)	出力するアドレスのオフセットを指定します。
		いいえ	出力するアドレスのオフセットを指定しません。
出力アドレスのオフセット	出力するアドレスのオフセットを指定します。 cx コマンドの -Xhex_offset オプションに相当します。 なお，本プロパティは，［出力アドレスのオフセットを指定する］プロパティで［はい (-Xhex_offset)］を選択した場合のみ表示します。		
	デフォルト	0（16 進数）	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0 ～ FFFFFFFE（16 進数）	
初期値なしデータのセクションをゼロ初期化する	ヘキサ・ファイルに変換する際に，初期値なしデータのセクション（セクション・タイプ NOBITS とセクション属性 A を持つセクション）をゼロ初期化するかどうかを選択します。 cx コマンドの -Xhex_null オプションに相当します。 なお，本プロパティは，［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合，または［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［いいえ］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xhex_null)	初期値なしデータのセクションをゼロ初期化します。
		いいえ	初期値なしデータのセクションをゼロ初期化しません。

(3)［シンボル・テーブル］

シンボル・テーブルに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイル・フォーマット］プロパティで［拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)］を選択した場合のみ表示します。

シンボル・テーブルを変換する	ヘキサ・ファイルに変換する際に，シンボル・テーブルを変換するかどうかを選択します。 cx コマンドの -Xhex_symtab オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (グローバル・シンボルとローカル・シンボルを変換する)(-Xhex_symtab=all)	グローバル・シンボルとローカル・シンボルを変換します。
		はい (グローバル・シンボルのみ変換する)(-Xhex_symtab=global)	グローバル・シンボルのみ変換します。
		いいえ	シンボル・テーブルを変換しません。

(4) [CRC 演算]

CRC 演算に関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，[ヘキサ・フォーマット] カテゴリの [ヘキサ・ファイル・フォーマット] プロパティで [拡張テクトロニクス・ヘキサ・フォーマット (-Xhex_format=T)] 以外，および [ヘキサ・ファイルに変換するアドレス範囲を指定する] プロパティで [はい (-Xhex_fill)] を選択した場合のみ表示します。

CRC 演算を行う	CRC（Cyclic Redundancy Check）演算を行うかどうかを選択します。 cx コマンドの -Xcrc オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xcrc)	指定した範囲のヘキサ形式オブジェクトを，下位アドレスから上位アドレスの順で CRC 演算を行い，演算結果を指定したアドレスへ出力します。
		いいえ	CRC 演算，および演算結果の出力を行いません。
CRC 結果出力アドレス	CRC 演算の結果を出力するアドレスを指定します。 本プロパティは必ず指定してください。 cx コマンドの -Xcrc オプションに相当します。 なお，本プロパティは，[CRC 演算を行う] プロパティで [はい (-Xcrc)] を選択した場合のみ表示されます。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0 ～ FFFFFFFF（16 進数）	

CRC 演算の範囲	CRC 演算の対象となる範囲を，開始アドレス，終了アドレスの順で指定します（例：0x0,0xFF）。 カンマで区切ることにより，複数指定することができます（例：0x0,0xFF,0x400,0x4FF）。 本プロパティは必ず指定してください。 cx コマンドの -Xcrc オプションに相当します。 なお，本プロパティは，［CRC 演算を行う］プロパティで［はい (-Xcrc)］を選択した場合のみ表示されます。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数） ［CRC 演算方法］プロパティで［高速 CRC］を選択した場合については，デバイスのユーザーズ・マニュアルを参照してください。	
CRC 演算方法	CRC 演算方法を選択します。 各演算方法の詳細については，デバイスのユーザーズ・マニュアルを参照してください。 cx コマンドの -Xcrc_method オプションに相当します。 なお，本プロパティは，［CRC 演算を行う］プロパティで［はい (-Xcrc)］を選択した場合のみ表示されます。		
	デフォルト	高速 CRC(CRC-16-CCITT)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	高速 CRC(CRC-16-CCITT)	高速 CRC（high-speed CRC）用の CRC-16-CCITT による演算結果を出力します。 CRC 演算の初期値は 0 となります。
		汎用 CRC	汎用 CRC（general-purpose CRC）用の演算結果を出力します。
CRC 演算の初期値	CRC 演算の初期値を指定します。 空欄の場合，0 を指定したものとします。 cx コマンドの -Xcrc_method オプションに相当します。 なお，本プロパティは，［CRC 演算方法］プロパティで［汎用 CRC］を選択した場合のみ表示されます。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0 ～ FFFF（16 進数）	

(5)［その他］

ヘキサ出力に関するその他の詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイルに変換するアドレス範囲を指定する］プロパティで［はい］を選択した場合のみ表示します。

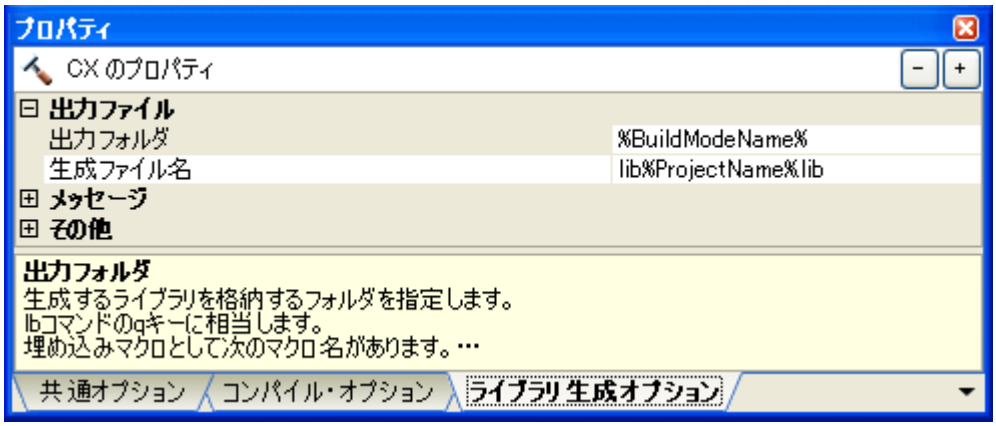
<table>
<tr><td rowspan="5">内蔵 ROM 領域オーバフロー時に警告を表示する</td><td colspan="3">ヘキサ・ファイルに変換する領域が内蔵 ROM 領域をオーバフローした場合，警告メッセージを表示するかどうかを選択します。
cx コマンドの -Xhex_rom_less オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい</td><td>ヘキサ・ファイルに変換する領域が内蔵 ROM 領域をオーバフローした場合，警告メッセージを表示します。</td></tr>
<tr><td>いいえ (-Xhex_rom_less)</td><td>ヘキサ・ファイルに変換する領域が内蔵 ROM 領域をオーバフローした場合，警告メッセージを表示しません。
本項目を選択する場合は，［開始アドレス］プロパティ，および［サイズ］プロパティの指定も必要です。</td></tr>
</table>

［ライブラリ生成オプション］タブ

本タブでは，ライブラリ生成フェーズに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

注意　本タブは，ライブラリ用のプロジェクトの場合のみ表示します。

図 A—10　プロパティ パネル：［ライブラリ生成オプション］タブ



［各カテゴリの説明］

（1）［出力ファイル］

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

出力フォルダ	生成するライブラリ・ファイルの出力先フォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 ライブラリアンの r キーに相当します。	
	デフォルト	%BuildModeName%
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集
	指定可能値	247 文字までの文字列
生成ファイル名	生成するライブラリ・ファイルの名前を指定します。 .lib 以外の拡張子を指定することはできません。 拡張子を省略した場合は，.lib を自動的に付加します。 次のプレースホルダに対応しています。 %ActiveProjectName% ：アクティブ・プロジェクト名に置換します。 %MainProjectName% ：メイン・プロジェクト名に置換します。 %ProjectName% ：プロジェクト名に置換します。 空欄の場合は，lib%ProjectName%.lib を指定したものとみなします。 ライブラリアンの r キーに相当します。	
	デフォルト	lib%ProjectName%.lib
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	259 文字までの文字列

(2) ［メッセージ］

メッセージに関する詳細情報の表示，および設定の変更を行います。

実行状況を表示する	ライブラリアンの実行状況[注]を出力 パネルに表示するかどうかを選択します。 ライブラリアンの v オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (v)	ライブラリアンの実行状況を表示します。
		いいえ	ライブラリアンの実行状況を表示しません。

注 実行状況の出力形式の意味を以下に示します。

出力形式	意味
q - *ファイル名*	ライブラリ・ファイルの新規作成，またはメンバの追加

(3)［その他］

ライブラリ生成に関するその他の詳細情報の表示，および設定の変更を行います。

<table>
<tr><td>ライブラリ生成前に実行するコマンド</td><td>
ライブラリ生成処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%LibraryFile% ：ライブラリ生成処理時の出力ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，ライブラリ生成処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
<table>
<tr><td>デフォルト</td><td>ライブラリ生成前に実行するコマンド [定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td rowspan="4">ライブラリ生成後に実行するコマンド</td><td colspan="2">ライブラリ生成処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%LibraryFile% ：ライブラリ生成処理時の出力ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，ライブラリ生成処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>ライブラリ生成後に実行するコマンド [定義数]</td></tr>
<tr><td>変更方法</td><td>[...] ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td rowspan="4">その他の追加オプション</td><td colspan="2">その他に追加するライブラリ生成オプションを入力します。
ここで設定したオプションは，ライブラリ生成オプション群の最後に付加します。</td></tr>
<tr><td>デフォルト</td><td>空欄</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

［ビルド設定］タブ

本タブでは，各 C ソース・ファイル，アセンブラ・ソース・ファイル，オブジェクト・モジュール・ファイル，リンク・ディレクティブ・ファイル，シンボル情報ファイル，ライブラリ・ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

図 A―11　プロパティ パネル：［ビルド設定］タブ（C ソース・ファイルを選択した場合）

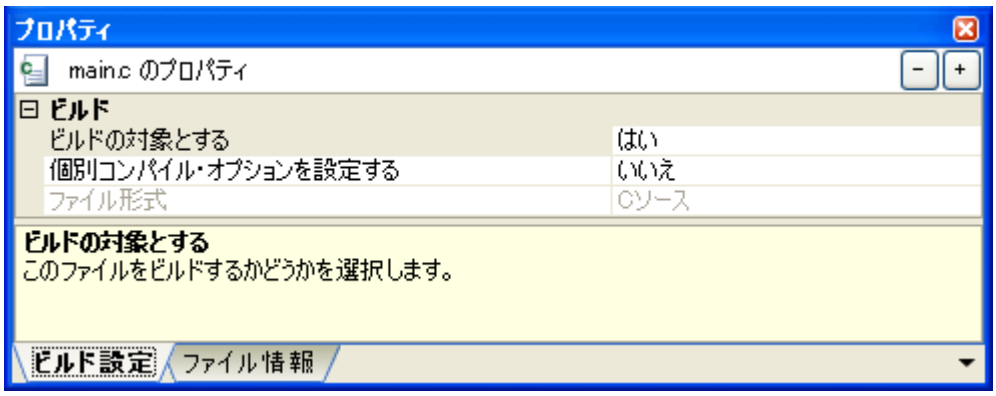


図 A―12　プロパティ パネル：［ビルド設定］タブ（アセンブラ・ソース・ファイルを選択した場合）

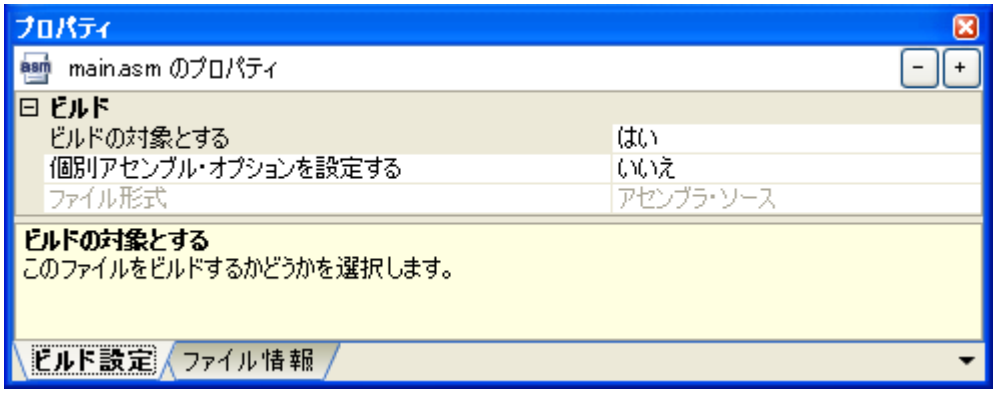


図 A—13 プロパティ パネル：［ビルド設定］タブ（オブジェクト・モジュール・ファイルを選択した場合）

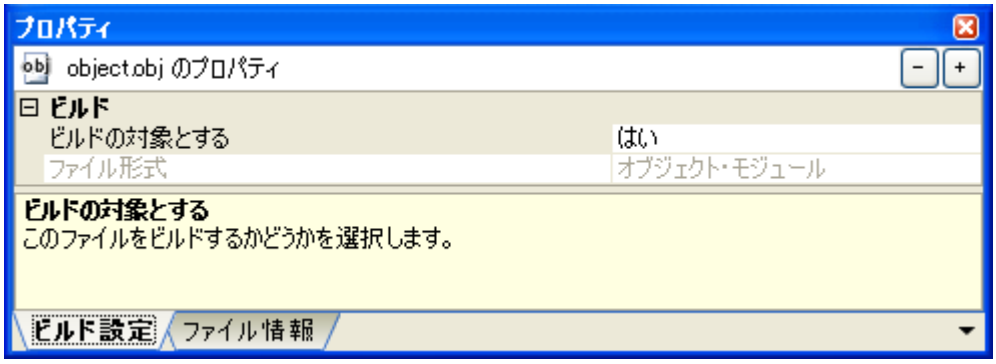


図 A—14 プロパティ パネル：［ビルド設定］タブ（リンク・ディレクティブ・ファイルを選択した場合）

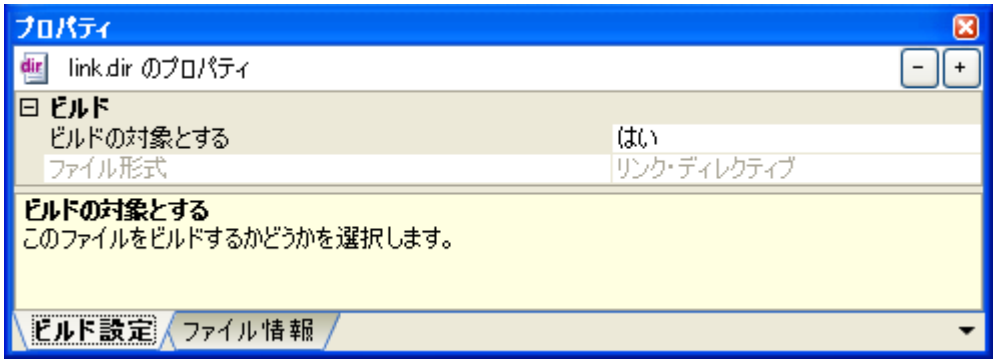


図 A—15 プロパティ パネル：［ビルド設定］タブ（シンボル情報ファイルを選択した場合）

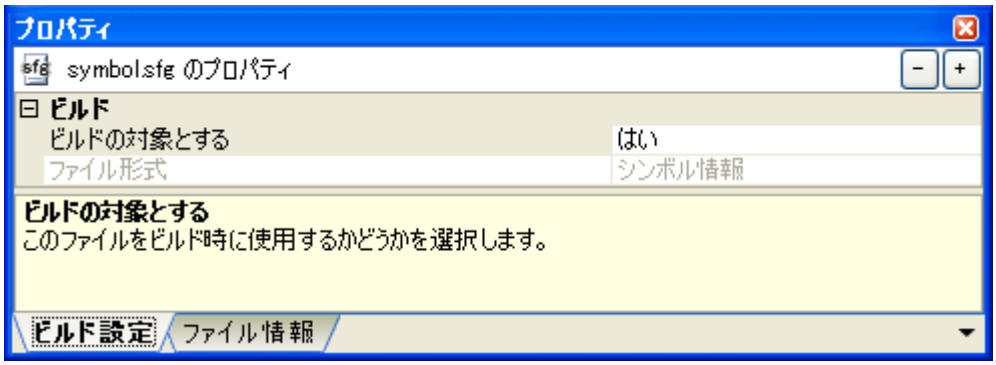
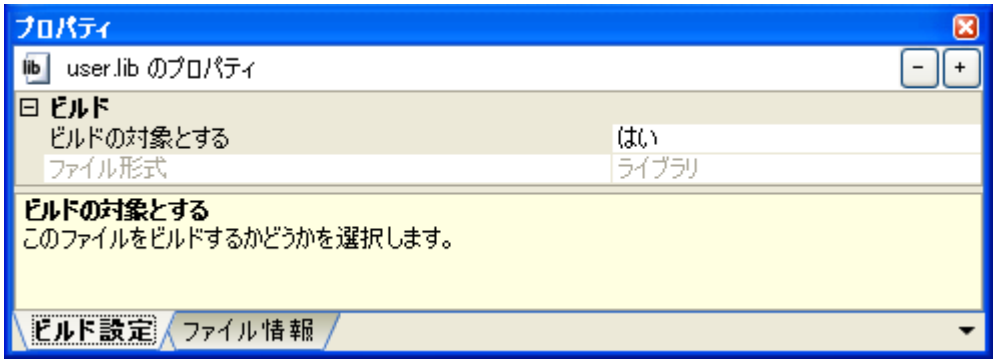


図 A―16　プロパティ パネル：［ビルド設定］タブ（ライブラリ・ファイルを選択した場合）



［各カテゴリの説明］

(1) ［ビルド］

ビルドに関する詳細情報の表示，および設定の変更を行います。

ビルドの対象とする	選択しているファイルをビルド対象とするかどうかを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	選択しているファイルをビルド対象とします。
		いいえ	選択しているファイルをビルド対象としません。
個別コンパイル・オプションを設定する	選択しているＣソース・ファイルにプロジェクトの設定とは異なるコンパイル・オプションを設定するかどうかを選択します。 なお，本プロパティは，プロジェクト・ツリーでＣソース・ファイルを選択し，本タブの［ビルドの対象とする］プロパティで［はい］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	選択しているＣソース・ファイルにプロジェクトの設定とは異なるオプションを設定します。
		いいえ	選択しているＣソース・ファイルにプロジェクトの設定とは異なるオプションを設定しません。

個別アセンブル・オプションを設定する	選択しているアセンブラ・ソース・ファイルにプロジェクトの設定とは異なるアセンブル・オプションを設定するかどうかを選択します。 なお，本プロパティは，プロジェクト・ツリーでアセンブラ・ソース・ファイルを選択し，本タブの［ビルドの対象とする］プロパティで［はい］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	選択しているアセンブラ・ソース・ファイルにプロジェクトの設定とは異なるオプションを設定します。
		いいえ	選択しているアセンブラ・ソース・ファイルにプロジェクトの設定とは異なるオプションを設定しません。
ファイル形式	選択しているファイルの形式を表示します。		
	デフォルト	C ソース（C ソース・ファイルを選択している場合） アセンブラ・ソース（アセンブラ・ソース・ファイルを選択している場合） オブジェクト・モジュール（オブジェクト・モジュール・ファイルを選択している場合） リンク・ディレクティブ（リンク・ディレクティブ・ファイルを選択している場合） シンボル情報ファイル（シンボル情報ファイルを選択している場合） ライブラリ（ライブラリ・ファイルを選択している場合）	
	変更方法	変更不可	

(2)［マルチコア］

マルチコアに関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，マルチコア対応デバイスを使用するプロジェクトにおいて，C ソース・ファイル，アセンブラ・ソース・ファイル，オブジェクト・モジュール・ファイル，ライブラリ・ファイルを選択している場合のみ表示します。

対象コア番号	対象コア番号を指定します。 ［コア *n*(-Xmulti=pe*n*)］は，コアの個数分表示されます（*n*：ターゲット CPU が持つコアの数）。 cx コマンドの -Xmulti オプションに相当します。		
	デフォルト	共通 (-Xmulti=cmn)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	共通 (-Xmulti=cmn)	選択しているファイルに含まれるコードとデータは，すべてのコアで利用可能となります。
		コア *n*(-Xmulti=pe*n*)	選択しているファイルに含まれるコードとデータは，指定したコアのみで利用可能となります。

［個別コンパイル・オプション］タブ

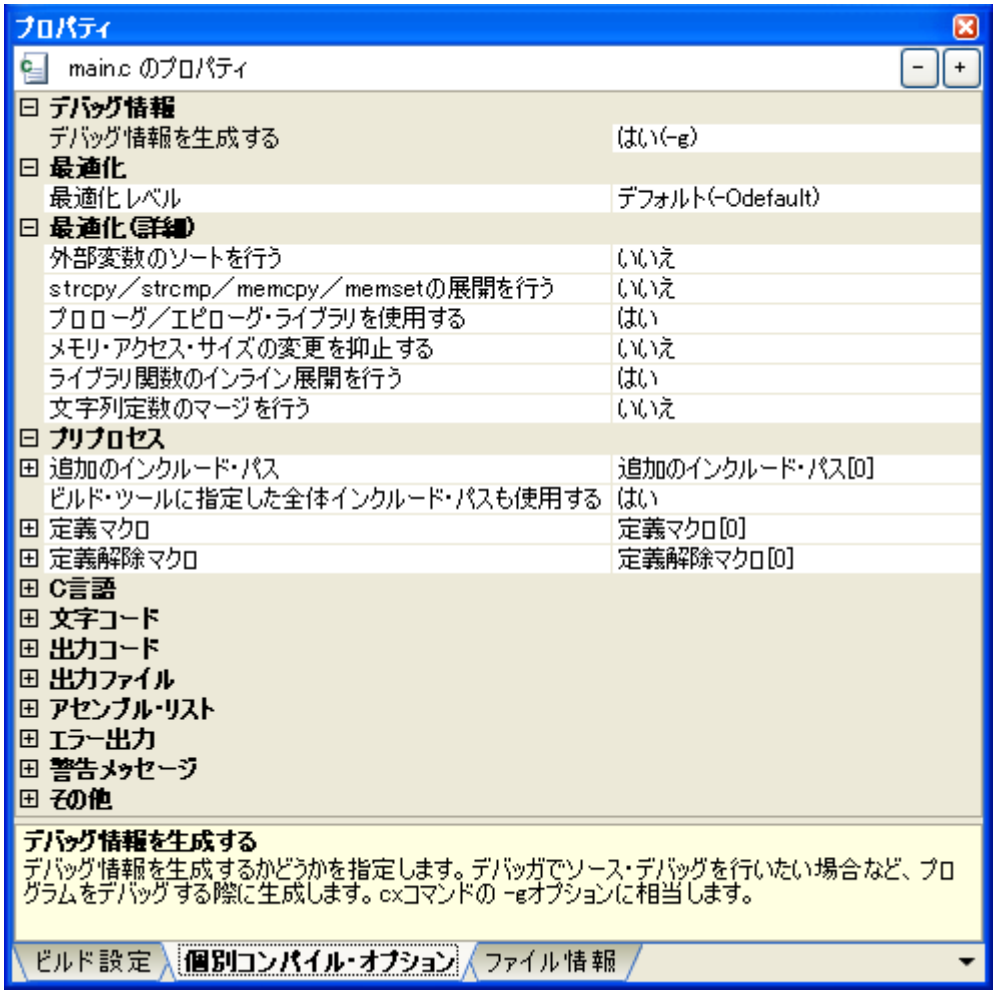
本タブでは，１つのＣソース・ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

なお，本タブは，［共通オプション］タブ，および［コンパイル・オプション］タブの設定内容を継承します。

これらのタブと異なる値を設定した場合は，プロパティが太字表示となります。

備考　本タブは，［ビルド設定］タブの［ビルド］カテゴリの［個別コンパイル・オプションを設定する］プロパティで［はい］を選択した場合のみ表示します。

図 A―17 プロパティ パネル：［個別コンパイル・オプション］タブ



［各カテゴリの説明］

(1) ［デバッグ情報］

デバッグ情報に関する詳細情報の表示，および設定の変更を行います。

デバッグ情報を生成する	デバッグ情報を生成するかどうかを選択します。 出力ファイル中にソース・デバッグ用の情報を出力することにより，デバッガでのソース・デバッグが可能となります。 cx コマンドの -g オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-g)	デバッグ情報を生成します。
		いいえ	デバッグ情報を生成しません。

(2)［最適化］

最適化に関する詳細情報の表示，および設定の変更を行います。

最適化レベル	コンパイルの最適化レベルを選択します。 cx コマンドの -O オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	デフォルト (-Odefault)	デバッグに影響しない範囲の最適化（式の最適化，およびレジスタ割り付けなど）を行います。
		サイズ優先 (-Osize)	オブジェクト・サイズ優先の最適化を行います。 ROM/RAM 容量の削減を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
		実行速度優先 (-Ospeed)	実行速度優先の最適化を行います。 実行速度の短縮を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
		デバッグ優先 (-Onothing)	デバッグを優先して最適化を行います。 デバッグのしやすさを重視し，デフォルトで実行する最適化を含むすべての最適化を抑止します。
		詳細指定	［最適化 (詳細)］カテゴリにプロパティを追加表示し，そこで選択した最適化項目を優先して最適化を行います。

(3)［最適化 (詳細)］

最適化に関する詳細情報の表示，および設定の変更を行います。

ループ展開最大数	for，while などのループを展開する最大数を指定します。 0，または 1 を指定した場合は，展開を抑止します。 空欄の場合は，4 を指定したものとみなします。 cx コマンドの -Ounroll オプションに相当します。 なお，本プロパティは，［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。	
	デフォルト	*コンパイル・オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0 ～ 999（10 進数），または空欄

未使用 static 関数の削除を行う	呼び出されない static 関数の削除を行うかどうかを選択します。 cx コマンドの -Odelete_static_func オプションに相当します。 なお，本プロパティは，［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Odelete_static_func=on)	呼び出されない static 関数の削除を行います。
		いいえ	呼び出されない static 関数の削除を指定しません。
関数のインライン展開を行う	関数を呼び出し箇所にインライン展開するかどうかを選択します。 cx コマンドの -Oinline オプションに相当します。 なお，本プロパティは，［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (指定関数のみ)(なし)	#pragma inline 指定した関数を呼び出し箇所にインライン展開します。
		はい (自動判別)(-Oinline=2)	自動的にインライン展開対象の関数を判別して展開します。
		はい (コード・サイズが増加しないよう自動判別)(-Oinline=3)	コード・サイズがなるべく増加しない範囲で，自動的にインライン展開対象の関数を判別して展開します。
		いいえ (-Oinline=0)	関数のインライン展開を指定しません。
パイプライン最適化を行う	機械語レベルで命令の並べ替えを行い，プログラムの実行性能を引き出すかどうかを選択します。 cx コマンドの -Opipeline オプションに相当します。 なお，本プロパティは，V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合，かつ［最適化］カテゴリの［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Opipeline=on)	パイプライン最適化を行います。
		いいえ	パイプライン最適化を指定しません。

共通コードのサブルーチン化を行う	共通コードのサブルーチン化を行うかどうかを選択します。 cx コマンドの -Osubroutine オプションに相当します。 なお，本プロパティは，［最適化］カテゴリの［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Osubroutine=1)	共通コードのサブルーチン化を行います。
		いいえ	共通コードのサブルーチン化を行いません。
関数末尾の関数呼び出しに jr 命令を使用する	関数の末尾が関数呼び出しの場合に，jarl 命令の代わりに jr 命令を優先的に使用するかどうかを選択します。 cx コマンドの -Otail_call オプションに相当します。 なお，本プロパティは，［最適化］カテゴリの［最適化レベル］プロパティで［詳細指定］を選択した場合のみ表示します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Otail_call=on)	関数の末尾が関数呼び出しの場合に，jarl 命令の代わりに jr 命令を優先的に使用します。 lp の退避／復帰コードを削除し，コード・サイズを小さくすることができます。 ただし，一部のデバッグ機能を使用することができなくなります。
		いいえ	関数の末尾が関数呼び出しの場合に，jarl 命令を使用します。
未使用関数の削除を行う	未使用関数の削除を行うかどうかを選択します。 本プロパティは，インライン展開後に不要になった関数を削除するのに有効です。 cx コマンドの -Xdelete_func オプションに相当します。 なお，本プロパティは，［コンパイル・オプション］タブの［シンボル情報］カテゴリの［使用するシンボル情報ファイル］プロパティに値が表示されている場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xdelete_func)	未使用関数の削除を行います。
		いいえ	未使用関数の削除を指定しません。

<table>
<tr><td rowspan="4">未使用関数の検索開始関数</td><td colspan="3">未使用関数の削除を行う際，未使用関数の検索を開始する関数の名前を指定します。
複数指定する場合は，関数名をカンマで区切って指定します（例：main,init）。
空欄の場合は，main を指定したものとみなします。
cx コマンドの -Xdelete_func オプションに相当します。
なお，本プロパティは，［未使用関数の削除を行う］プロパティで［はい］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td colspan="2">テキスト・ボックスによる直接入力</td></tr>
<tr><td>指定可能値</td><td colspan="2">1022 文字までの文字列</td></tr>
<tr><td rowspan="4">外部変数のソートを行う</td><td colspan="3">外部変数のソートを行うかどうかを選択します。
本プロパティは，RAM 容量を削減するためのものです。
cx コマンドの -Xsort_var オプションに相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td>指定可能値</td><td>はい (-Xsort_var)</td><td>const/sconst セクション以外のセクションに配置している外部変数をアライメントが大きい順に並び替えます。</td></tr>
<tr><td></td><td>いいえ</td><td>外部変数のソートを指定しません。</td></tr>
<tr><td rowspan="5">strcpy ／ strcmp ／ memcpy ／ memset の展開を行う</td><td colspan="3">配列（文字列を含む），および構造体の整列条件を 4 バイトとし，関数 strcpy()，strcmp()，memcpy()，memset() の呼び出しをインライン展開するかどうかを選択します。
生成するプログラムの実行速度は高速になりますが，コード・サイズは増大します。
cx コマンドの -Xinline_strcpy オプションに相当します。
なお，本プロパティは，［コンパイル・オプション］タブの［出力コード］カテゴリの［構造体パッキングを行う］プロパティで［いいえ］を選択した場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xinline_strcpy)</td><td>関数 strcpy()，strcmp()，memcpy()，memset() の呼び出しをインライン展開します。</td></tr>
<tr><td>いいえ</td><td>関数 strcpy()，strcmp()，memcpy()，memset() のインライン展開を指定しません。</td></tr>
</table>

プロローグ／エピローグ・ライブラリを使用する	関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にするかどうかを選択します。 cx コマンドの -Xpro_epi_runtime オプションに相当します。 なお，本プロパティは，［最適化］カテゴリの［最適化レベル］プロパティで［実行速度優先 (-Ospeed)］を選択した場合は表示しません。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にします。
		いいえ (-Xpro_epi_runtime=off)	関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理に指定しません。
メモリ・アクセス・サイズの変更を抑止する	ロード／ストア命令を 1 ビット操作命令（set1，clr1，tst1，not1）へ置き換える動作を禁止するかどうかを選択します。 cx コマンドの -Xkeep_access_size オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xkeep_access_size)	ロード／ストア命令を 1 ビット操作命令へ置き換える動作を禁止します。
		いいえ	ロード／ストア命令の代わり 1 ビット操作命令を生成することがあります。
ライブラリ関数のインライン展開を行う	ライブラリ関数の呼び出し箇所において，インライン展開を行うかどうかを選択します。 cx コマンドの -Xcall_lib オプションに相当します。 なお，本プロパティは，［最適化 (詳細)］カテゴリの［strcpy ／ strcmp ／ memcpy ／ memset の展開を行う］プロパティで［いいえ］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	sqrtf（FPU を持つデバイスのみ），memcmp，memcpy，memset 関数を呼び出す代わりに，適切な命令（命令列）を生成します。
		いいえ (-Xcall_lib)	常にライブラリ関数を呼び出す命令を生成します。

文字列定数のマージを行う	ソース・ファイル内で同じ文字列定数が複数存在する場合，これらをまとめて 1 つの領域に割り付けるかどうかを選択します。 cx コマンドの -Xmerge_string オプションに相当します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xmerge_string)	ソース・ファイル内で複数存在する同じ文字列定数をまとめて 1 つの領域に割り付けます。
		いいえ	ソース・ファイル内で複数存在する同じ文字列定数をそれぞれを別々の領域に割り付けます。

(4) [プリプロセス]

プリプロセスに関する詳細情報の表示，および設定の変更を行います。

追加のインクルード・パス	コンパイル時の追加のインクルード・パスを指定します。 次のプレースホルダに対応しています。 %ActiveProjectDir% : アクティブ・プロジェクト・フォルダの絶対パスに置換します。 %ActiveProjectName% : アクティブ・プロジェクト名に置換します。 %BuildModeName% : ビルド・モード名に置換します。 %MainProjectDir% : メイン・プロジェクト・フォルダの絶対パスに置換します。 %MainProjectName% : メイン・プロジェクト名に置換します。 %MicomToolPath% : 本製品のインストール・フォルダの絶対パスに置換します。 %ProjectDir% : プロジェクト・フォルダの絶対パスに置換します。 %ProjectName% : プロジェクト名に置換します。 %TempDir% : テンポラリ・フォルダの絶対パスに置換します。 %WinDir% : Windows システム・フォルダの絶対パスに置換します。 指定したインクルード・パスは，CX の標準インクルード・ファイル・フォルダよりも優先して検索します。 パスはプロジェクト・フォルダを基点とします。 本プロパティを省略した場合は，CX の標準インクルード・ファイル・フォルダのみ検索します。 cx コマンドの -I オプションに相当します。 指定したインクルード・パスはサブプロパティとして表示します。 インクルード・パスに大文字，小文字の区別はありません。	
	デフォルト	追加のインクルード・パス [*定義数*]
	変更方法	[...] ボタンをクリックし，パス編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	259 文字までの文字列 256 個まで指定可能です。

<table>
<tr><td>ビルド・ツールに指定した全体インクルード・パスも使用する</td><td colspan="3">使用するビルド・ツールの［コンパイル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］プロパティで指定したインクルード・パスも使用してコンパイルするかどうかを選択します。
インクルード・パスは，以下の順で追加します。
- 本タブの［追加のインクルード・パス］プロパティで指定したパス
- ［コンパイル・オプション］タブの［追加のインクルード・パス］プロパティで指定したパス
- ［コンパイル・オプション］タブの［システム・インクルード・パス］プロパティに表示しているパス
cx コマンドの -I オプションに相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">はい</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td></td><td>指定可能値</td><td>はい</td><td>使用するビルド・ツールのプロパティで指定したインクルード・パスも使用してコンパイルします。</td></tr>
<tr><td></td><td></td><td>いいえ</td><td>使用するビルド・ツールのプロパティで指定したインクルード・パスを使用しません。</td></tr>
<tr><td>定義マクロ</td><td colspan="3">定義したいマクロ名を指定します。
「マクロ名 = 定義値」の形式で 1 行に 1 つずつ指定します。
「= 定義値」の部分は省略可能で，省略した場合，定義値を 1 とします。
cx コマンドの -D オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td colspan="2">256 文字までの文字列
256 個まで指定可能です。</td></tr>
<tr><td>定義解除マクロ</td><td colspan="3">定義解除したいマクロ名を指定します。
「マクロ名」の形式で 1 行に 1 つずつ指定します。
cx コマンドの -U オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td colspan="2">256 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

プリプロセス結果にCソース・コメントを出力する	プリプロセス結果のファイルに，C ソースのコメントを出力するかどうかを選択します。 cx コマンドの -Xpreprocess オプションに相当します。 なお，本プロパティは，［出力ファイル］カテゴリの［プリプロセス処理したソースを出力する］プロパティで［はい (-P)］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xpreprocess=comment)	プリプロセス結果のファイルに，C ソースのコメントを出力します。
		いいえ	プリプロセス結果のファイルに，C ソースのコメントを出力しません。
プリプロセス結果に行番号情報を出力する	プリプロセス結果のファイルに，C ソースの行番号情報を出力するかどうかを選択します。 cx コマンドの -Xpreprocess オプションに相当します。 なお，本プロパティは，［出力ファイル］カテゴリの［プリプロセス処理したソースを出力する］プロパティで［はい (-P)］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xpreprocess=line)	プリプロセス結果のファイルに，C ソースの行番号情報を出力します。
		いいえ	プリプロセス結果のファイルに，C ソースの行番号情報を出力しません。

(5)［C 言語］

C 言語に関する詳細情報の表示，および設定の変更を行います。

ANSI 規格に厳密に合わせてコンパイルする	C ソース・プログラムを ANSI 規格に厳密にあわせて処理し，規格に反する記述に対してエラーや警告を出力するかどうかを選択します。 cx コマンドの -Xansi オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xansi)	C ソース・プログラムを ANSI 規格に厳密にあわせて処理し，規格に反する記述に対してエラーや警告を出力します。
		いいえ	従来の C 言語の仕様との両立性を持たせ，警告を出力して処理を続行します。

(6)［文字コード］

文字コードに関する詳細情報の表示，および設定の変更を行います。

文字コード	ソース・ファイル中の日本語のコメント，文字列に対して，使用する文字コードを選択します。 cx コマンドの -Xcharacter_set オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	SJIS(なし)	ソース・ファイル中の日本語の文字コードを SJIS と解釈します。
		EUC(-Xcharacter_set=euc_jp)	ソース・ファイル中の日本語の文字コードを EUC と解釈します。
		Big5(-Xcharacter_set=big5)	ソース・ファイル中の中国語の文字コードを繁体字中国語と解釈します。
		GB2312(-Xcharacter_set=gb2312)	ソース・ファイル中の中国語の文字コードを簡体字中国語と解釈します。
		UFT-8(-Xcharacter_set=utf8)	ソース・ファイル中の日本語の文字コードを UFT-8 と解釈します。
		処理しない (-Xcharacter_set=none)	ソース・ファイル中の日本語の文字コードを処理しません。

(7)［出力コード］

出力コードに関する詳細情報の表示，および設定の変更を行います。

アセンブラ・ソースにコメントを出力する	出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力するかどうかを選択します。 cx コマンドの -Xpass_source オプションに相当します。 なお，本プロパティは，［出力ファイル］カテゴリの［アセンブラ・ソース・ファイルを出力する］プロパティで［はい (-Xasm_path)］を選択した場合，または［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xpass_source)	出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力します。
		いいえ	出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力しません。

switch 文の出力コードの選択	プログラム中の switch 文のコード出力形式を選択します。 cx コマンドの -Xswitch オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	自動選択 (なし)	cx コマンドが最適な出力形式を自動的に選択します。
		if-else(-Xswitch=ifelse)	プログラム中の switch 文を case 文の並びに沿った if-else 文と同じ形式で出力します。 頻度が多い順に case 文を書いているときやラベル数が少ないときに，本項目を選択します。 上から順に比較するので，頻繁に合致する case 文を先に記述すると余計な比較が減り，実行速度向上につながります。
		バイナリ・サーチ (-Xswitch=binary)	プログラム中の switch 文をバイナリ・サーチ形式で出力します。 バイナリ・サーチ・アルゴリズムに用いて合致する case 文を探します。 ラベル数が多いときに本項目を選択すると，どの case 文も同じくらいの速さで見つけることができます。
		テーブル分岐 (-Xswitch=table)	プログラム中の switch 文をテーブル・ジャンプ形式で出力します。 case 文の値を基にインデックス化したテーブルを参照し，switch 文の値により case ラベルを選択して処理を行います。 どの case 文にも同じくらい速く分岐します。 ただし，case 値が連続していないときは無駄な領域ができます。
switch テーブルのラベル・サイズ (バイト)	switch 文の case ラベルに対する分岐テーブルの 1 ラベルあたりのサイズを選択します。 cx コマンドの -Xword_case オプションに相当します。 なお，本プロパティは，［switch 文の出力コードの選択］プロパティで［自動選択 (なし)］，または［テーブル分岐 (-Xswitch=table)］を選択した場合のみ表示します。 ［switch 文の出力コードの選択］プロパティで［自動選択 (なし)］を選択したときに cx コマンドが［if-else(-Xswitch=ifelse)］，または［バイナリ・サーチ (-Xswitch=binary)］を自動選択した場合，本プロパティの設定は無効（2 バイト固定）となります。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	2 バイト (なし)	switch 文の case ラベルに対する分岐テーブルを 1 ラベルあたり 2 バイトで生成します。
		4 バイト (-Xword_case)	switch 文の case ラベルに対する分岐テーブルを 1 ラベルあたり 4 バイトで生成します。 長い switch 文のためにコンパイル・エラーとなる場合に選択します。

<table>
<tr><td rowspan="5">浮動小数点演算方法</td><td colspan="3">浮動小数点演算に対して，ランタイム・ライブラリの呼び出し命令を生成するか，FPU（浮動小数点ユニット）の浮動小数点演算命令を生成するかを選択します。
cx コマンドの -Xfloat【V850E2V3】オプションに相当します。
なお，本プロパティは，FPU を持つ V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="3">指定可能値</td><td>自動選択 (なし)</td><td>ターゲット・デバイスが FPU を持たない場合は，浮動小数点演算命令を生成せず，ランタイム・ライブラリの呼び出し命令を生成します。
ターゲット・デバイスが FPU を持つ場合は，浮動小数点演算命令を生成します。</td></tr>
<tr><td>ソフトウェアで行う (-Xfloat=soft)</td><td>浮動小数点演算に対して，ランタイム・ライブラリの呼び出し命令を生成します。</td></tr>
<tr><td>FPU で行う (-Xfloat=fpu)</td><td>浮動小数点演算に対して，FPU（浮動小数点ユニット）の浮動小数点演算命令を生成します。</td></tr>
<tr><td rowspan="5">div ／ divu 除算命令を生成する</td><td colspan="3">除算に対して，divq，および divqu 命令を生成する代わりに，div，および divu 命令を生成するかどうかを選択します。
divq，および divqu 命令は高速ですが，実行サイクル数がオペランドの値により変わります。
cx コマンドの -Xdiv【V850E2V3】オプションに相当します。
なお，本プロパティは，FPU を持つ V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td rowspan="2">指定可能値</td><td>はい (-Xdiv)</td><td>除算に対して，div，および divu 命令を生成します。</td></tr>
<tr><td>いいえ</td><td>除算に対して，divq，および divqu 命令を生成します。</td></tr>
</table>

Far Jump ファイル名	Far Jump ファイル名を指定します。 Far Jump ファイルには，ファイルに記述した関数への分岐命令に対して，jmp 命令（V850E/ES コアの場合），または jarl32，および jr32 命令（V850E2 コア，または V850E2V3 アーキテクチャの場合）を使用したコードを出力します。 関数本体が jarl，および jr 命令では分岐できない範囲（± 2M バイト以上）にあり，cx コマンドがエラーを出力する場合，Far Jump ファイルを用いてコンパイルし直します。 なお，拡張子は .fjp としてください。 cx コマンドの -Xfar_jump オプションに相当します。 指定した Far Jump ファイル名はサブプロパティとして表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	[...] ボタンをクリックし，パス編集 ダイアログをオープン →［参照］ボタンをクリックし，Far Jump ファイルを指定 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能	
	指定可能値	259 文字までの文字列 5000 個まで指定可能です。 ただし，最後に指定したファイル名のみが有効となります。	

(8)［出力ファイル］

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

オブジェクト・モジュール・ファイル名	コンパイル後に生成するオブジェクト・モジュール・ファイルの名前を指定します。 .obj 以外の拡張子を指定することはできません。 拡張子を省略した場合は，.obj を自動的に付加します。 空欄の場合は，ソース・ファイル名の拡張子を .obj に置き換えたものとなります。 cx コマンドの -o オプションに相当します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	259 文字までの文字列	
アセンブラ・ソース・ファイルを出力する	C ソースのコンパイル結果のアセンブラ・ソース・ファイルを出力するかどうかを選択します。 cx コマンドの -Xasm_path オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xasm_path)	C ソースのコンパイル結果のアセンブラ・ソース・ファイルを出力します。
		いいえ	C ソースのコンパイル結果のアセンブラ・ソース・ファイルを出力しません。

アセンブラ・ソース・ファイル出力フォルダ	アセンブラ・ソース・ファイルの出力先フォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ：ビルド・モード名に置換します。 アセンブラ・ソース・ファイルは，C ソース・ファイルの拡張子を .asm で置き換えたファイル名で出力します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xasm_path オプションに相当します。 なお，本プロパティは，［アセンブラ・ソース・ファイルを出力する］プロパティで［はい (-Xasm_path)］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集	
	指定可能値	247 文字までの文字列	
プリプロセス処理したソースを出力する	ソース・ファイルに対して，プリプロセス処理を実行した結果をファイルに出力するかどうかを選択します。 ファイルの出力先は，［中間ファイル出力フォルダ］プロパティで指定したフォルダです。 ファイルは，ソース・ファイルの拡張子を .i で置き換えた名前で出力します。 cx コマンドの -P オプションに相当します。 なお，本プロパティは，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［いいえ］を選択した場合のみ表示します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-P)	ソース・ファイルに対して，プリプロセス処理を実行した結果をファイルに出力します。
		いいえ	ソース・ファイルに対して，プリプロセス処理を実行した結果をファイルに出力しません。

(9)［アセンブル・リスト］

アセンブル・リストに関する詳細情報の表示，および設定の変更を行います。

アセンブル・リスト・ファイルを出力する	アセンブル・リスト・ファイルを出力するかどうかを選択します。 cx コマンドの -Xprn_path オプションに相当します。		
	デフォルト	*コンパイル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xprn_path)	アセンブル・リスト・ファイルを出力します。
		いいえ	アセンブル・リスト・ファイルを出力しません。

アセンブル・リスト・ファイル出力フォルダ	アセンブル・リスト・ファイルの出力先フォルダを指定します。 アセンブル・リスト・ファイルは，ソース・ファイル名の拡張子を .prn で置き換えた名前で出力します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ： ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xprn_path オプションに相当します。 なお，本プロパティは，［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択した場合のみ表示します。	
	デフォルト	*コンパイル・オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集
	指定可能値	247 文字までの文字列

（10）［エラー出力］

エラー出力に関する詳細情報の表示，および設定の変更を行います。

エラー・メッセージ・ファイルを出力する	エラー・メッセージ・ファイルを出力 パネルに出力するかどうかを選択します。 cx コマンドの -Xerror_file オプションに相当します。		
	デフォルト	*共通オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xerror_file)	エラー・メッセージ・ファイルを出力します。
		いいえ	エラー・メッセージ・ファイルを出力しません。
エラー・メッセージ・ファイル出力フォルダ	エラー・メッセージ・ファイルの出力先フォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ： ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティは，［エラー・メッセージ・ファイルを出力する］プロパティで［はい (-Xerror_file)］を選択した場合のみ表示します。		
	デフォルト	*共通オプションの設定値*	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集	
	指定可能値	247 文字までの文字列	

エラー・メッセージ・ファイル名	エラー・メッセージ・ファイルの名前を指定します。 拡張子は自由に指定可能です。 次のプレースホルダに対応しています。 %ActiveProjectName% ：アクティブ・プロジェクト名に置換します。 %MainProjectName% ：メイン・プロジェクト名に置換します。 %ProjectName% ：プロジェクト名に置換します。 空欄の場合は，%ProjectName%.err を指定したものとみなします。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティは，［エラー・メッセージ・ファイルを出力する］プロパティで［はい (-Xerror_file)］を選択した場合のみ表示します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	259 文字までの文字列

（11）［警告メッセージ］

警告メッセージに関する詳細情報の表示，および設定の変更を行います。

必ず表示させる警告メッセージ	警告レベルに関わらず，表示させたい警告メッセージの番号を指定します。 複数指定する場合は，メッセージ番号をカンマで区切って指定します（例：02042,02107）。 また，ハイフンを使用して，区間設定を行うこともできます（例：02222-02554,02699-02782）。 ［表示させない警告メッセージ］プロパティと同一の番号を指定した場合は，本プロパティを優先します。 cx コマンドの -Xwarning オプションに相当します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	2048 文字までの文字列
表示させない警告メッセージ	警告レベルに関わらず，表示させない警告メッセージの番号を指定します。 複数指定する場合は，メッセージ番号をカンマで区切って指定します（例：02042,02107）。 また，ハイフンを使用して，区間設定を行うこともできます（例：02222-02554,02699-02782）。 ［必ず表示させる警告メッセージ］プロパティと同一の番号を指定した場合は，［必ず表示させる警告メッセージ］プロパティを優先します。 cx コマンドの -Xno_warning オプションに相当します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	2048 文字までの文字列

（12）［その他］

コンパイルに関するその他の詳細情報の表示，および設定の変更を行います。

<table>
<tr><td>コンパイル前に実行するコマンド</td><td colspan="2">コンパイル処理前に実行するコマンドを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%CompiledFile% ：コンパイル時の出力ファイルの絶対パスに置換します。
%InputFile% ：コンパイル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，コンパイル処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
空欄
- 上記以外の場合
コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>

<table>
<tr><td rowspan="4">コンパイル後に実行するコマンド</td><td colspan="2">コンパイル処理後に実行するコマンドを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%CompiledFile% ：コンパイル時の出力ファイルの絶対パスに置換します。
%InputFile% ：コンパイル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，コンパイル処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
空欄
- 上記以外の場合
コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td rowspan="4">その他の追加オプション</td><td colspan="2">その他に追加するコンパイル・オプションを入力します。
ここで設定したオプションは，コンパイル・オプション群の最後に付加します。</td></tr>
<tr><td>デフォルト</td><td>- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
空欄
- 上記以外の場合
コンパイル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

［個別アセンブル・オプション］タブ

本タブでは，１つのアセンブラ・ソース・ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

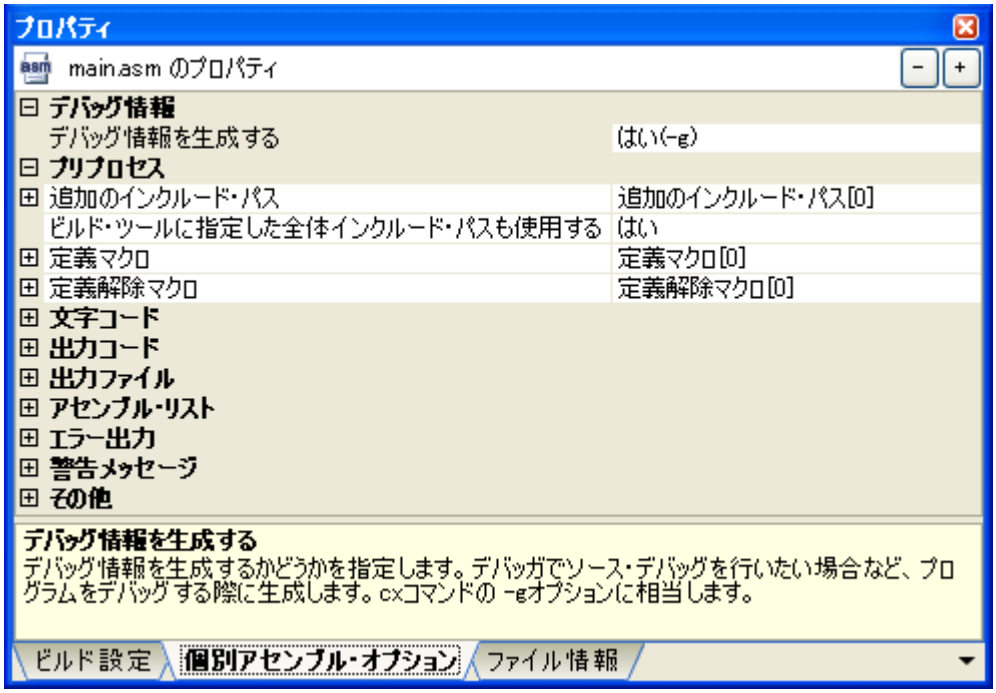
なお，本タブは，［共通オプション］タブ，［コンパイル・オプション］タブ，および［アセンブル・オプション］タブの設定内容を継承します。

これらのタブと異なる値を設定した場合は，プロパティが太字表示となります。

（1）［デバッグ情報］
（2）［プリプロセス］
（3）［文字コード］
（4）［出力コード］
（5）［出力ファイル］
（6）［アセンブル・リスト］
（7）［エラー出力］
（8）［警告メッセージ］
（9）［その他］

備考 本タブは，［ビルド設定］タブの［ビルド］カテゴリの［個別アセンブル・オプションを設定する］プロパティで［はい］を選択した場合のみ表示します。

図 A—18 プロパティ パネル：［個別アセンブル・オプション］タブ



［各カテゴリの説明］

(1)［デバッグ情報］

デバッグ情報に関する詳細情報の表示，および設定の変更を行います。

デバッグ情報を生成する	デバッグ情報を生成するかどうかを選択します。 出力ファイル中にソース・デバッグ用の情報を出力することにより，デバッガでのソース・デバッグが可能となります。 cx コマンドの -g オプションに相当します。		
	デフォルト	- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合 *コンパイル・オプションの設定値* - 上記以外の場合 *アセンブル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-g)	デバッグ情報を生成します。
		いいえ	デバッグ情報を生成しません。

(2)［プリプロセス］

プリプロセスに関する詳細情報の表示，および設定の変更を行います。

<table>
<tr><td>追加のインクルード・パス</td><td>アセンブル時の追加のインクルード・パスを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
指定したインクルード・パスは，CX の標準インクルード・ファイル・フォルダよりも優先して検索します。
パスはプロジェクト・フォルダを基点とします。
本プロパティを省略した場合は，CX の標準インクルード・ファイル・フォルダのみ検索します。
cx コマンドの -I オプションに相当します。
指定したインクルード・パスはサブプロパティとして表示します。
インクルード・パスに大文字，小文字の区別はありません。
<table><tr><td>デフォルト</td><td>追加のインクルード・パス [定義数]</td></tr><tr><td>変更方法</td><td>[...] ボタンをクリックし，パス編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr><tr><td>指定可能値</td><td>259 文字までの文字列
256 個まで指定可能です。</td></tr></table></td></tr>
</table>

<table>
<tr><td>ビルド・ツールに指定した全体インクルード・パスも使用する</td><td colspan="3">使用するビルド・ツールの［アセンブル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］プロパティで指定したインクルード・パスも使用してアセンブルするかどうかを選択します。
なお，［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合は，［コンパイル・オプション］タブの設定を使用します。
［一括ビルドを行う］プロパティが［はい］の場合，インクルード・パスは，以下の順で追加します。
- 本タブの［追加のインクルード・パス］プロパティで指定したパス
- ［コンパイル・オプション］タブの［追加のインクルード・パス］プロパティで指定したパス
- ［コンパイル・オプション］タブの［システム・インクルード・パス］プロパティに表示しているパス
［一括ビルドを行う］プロパティが［いいえ］の場合，インクルード・パスは，以下の順で追加します。
- 本タブの［追加のインクルード・パス］プロパティで指定したパス
- ［アセンブル・オプション］タブの［追加のインクルード・パス］プロパティで指定したパス
- ［アセンブル・オプション］タブの［システム・インクルード・パス］プロパティに表示しているパス
cx コマンドの -I オプションに相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">はい</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">ドロップダウン・リストによる選択</td></tr>
<tr><td></td><td>指定可能値</td><td>はい</td><td>使用するビルド・ツールのプロパティで指定したインクルード・パスも使用してアセンブルします。</td></tr>
<tr><td></td><td></td><td>いいえ</td><td>使用するビルド・ツールのプロパティで指定したインクルード・パスを使用しません。</td></tr>
<tr><td>定義マクロ</td><td colspan="3">定義したいマクロ名を指定します。
「マクロ名 = 定義値」の形式で 1 行に 1 つずつ指定します。
「= 定義値」の部分は省略可能で，省略した場合，定義値を 1 とします。
cx コマンドの -D オプションに相当します。
指定したマクロはサブプロパティとして表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
コンパイル・オプションの設定値
- 上記以外の場合
アセンブル・オプションの設定値</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td></td><td>指定可能値</td><td colspan="2">256 文字までの文字列
256 個まで指定可能です。</td></tr>
</table>

定義解除マクロ	定義解除したいマクロ名を指定します。 「*マクロ名*」の形式で 1 行に 1 つずつ指定します。 cx コマンドの -U オプションに相当します。 指定したマクロはサブプロパティとして表示します。	
	デフォルト	- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合 *コンパイル・オプションの設定値* - 上記以外の場合 *アセンブル・オプションの設定値*
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	256 文字までの文字列 256 個まで指定可能です。

(3) ［文字コード］

文字コードに関する詳細情報の表示，および設定の変更を行います。

文字コード	ソース・ファイル中の日本語のコメント，文字列に対して，使用する文字コードを選択します。 cx コマンドの -Xcharacter_set オプションに相当します。		
	デフォルト	- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合 *コンパイル・オプションの設定値* - 上記以外の場合 *アセンブル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	SJIS(なし)	ソース・ファイル中の日本語の文字コードを SJIS と解釈します。
		EUC(-Xcharacter_set=euc_jp)	ソース・ファイル中の日本語の文字コードを EUC と解釈します。
		UFT-8(-Xcharacter_set=utf8)	ソース・ファイル中の日本語の文字コードを UFT-8 と解釈します。
		Big5(-Xcharacter_set=big5)	ソース・ファイル中の中国語の文字コードを繁体字中国語と解釈します。
		GB2312(-Xcharacter_set=gb2312)	ソース・ファイル中の中国語の文字コードを簡体字中国語と解釈します。
		処理しない (-Xcharacter_set=none)	ソース・ファイル中の日本語の文字コードを処理しません。

(4)［出力コード］

出力コードに関する詳細情報の表示，および設定の変更を行います。

32 ビット分岐命令を使用する	jarl，および jr 命令に対して，far jump 機能を使用するかどうかを選択します。 far jump 機能を使用することにより，jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。 cx コマンドの -Xasm_far_jump【V850E2】【V850E2V3】オプションに相当します。 なお，本プロパティは，V850E2 コア，および V850E2V3 アーキテクチャのデバイスを使用するプロジェクトの場合のみ表示します。		
	デフォルト	- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合 *コンパイル・オプションの設定値* - 上記以外の場合 *アセンブル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xasm_far_jump)	jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。
		いいえ	jarl，および jr 命令としてアセンブルを行います。

(5)［出力ファイル］

出力ファイルに関する詳細情報の表示，および設定の変更を行います。

オブジェクト・モジュール・ファイル名	アセンブル後に生成するオブジェクト・モジュール・ファイルの名前を指定します。 .obj 以外の拡張子を指定することはできません。 拡張子を省略した場合は，.obj を自動的に付加します。 空欄の場合は，ソース・ファイル名の拡張子を .obj に置き換えたものとなります。 cx コマンドの -o オプションに相当します。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	259 文字までの文字列

(6)［アセンブル・リスト］

アセンブル・リストに関する詳細情報の表示，および設定の変更を行います。

アセンブル・リスト・ファイルを出力する	アセンブル・リスト・ファイルを出力するかどうかを選択します。 cx コマンドの -Xprn_path オプションに相当します。		
	デフォルト	- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合 *コンパイル・オプションの設定値* - 上記以外の場合 *アセンブル・オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xprn_path)	アセンブル・リスト・ファイルを出力します。
		いいえ	アセンブル・リスト・ファイルを出力しません。
アセンブル・リスト・ファイル出力フォルダ	アセンブル・リスト・ファイルの出力先フォルダを指定します。 アセンブル・リスト・ファイルは，ソース・ファイル名の拡張子を .prn で置き換えた名前で出力します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ： ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xprn_path オプションに相当します。 なお，本プロパティは，［アセンブル・リスト・ファイルを出力する］プロパティで［はい (-Xprn_path)］を選択した場合のみ表示します。		
	デフォルト	- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合 *コンパイル・オプションの設定値* - 上記以外の場合 *アセンブル・オプションの設定値*	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集	
	指定可能値	247 文字までの文字列	

(7) ［エラー出力］

エラー出力に関する詳細情報の表示，および設定の変更を行います。

エラー・メッセージ・ファイルを出力する	エラー・メッセージ・ファイルを出力 パネルに出力するかどうかを選択します。 cx コマンドの -Xerror_file オプションに相当します。		
	デフォルト	*共通オプションの設定値*	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい (-Xerror_file)	エラー・メッセージ・ファイルを出力します。
		いいえ	エラー・メッセージ・ファイルを出力しません。

エラー・メッセージ・ファイル出力フォルダ	エラー・メッセージ・ファイルの出力先フォルダを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 絶対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とした相対パスに変換します（ドライブが異なる場合を除く）。 次のプレースホルダに対応しています。 %BuildModeName% ：ビルド・モード名に置換します。 空欄の場合は，プロジェクト・フォルダを指定したものとみなします。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティは，［エラー・メッセージ・ファイルを出力する］プロパティで［はい (-Xerror_file)］を選択した場合のみ表示します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集
	指定可能値	247 文字までの文字列
エラー・メッセージ・ファイル名	エラー・メッセージ・ファイルの名前を指定します。 拡張子は自由に指定可能です。 次のプレースホルダに対応しています。 %ActiveProjectName% ：アクティブ・プロジェクト名に置換します。 %MainProjectName% ：メイン・プロジェクト名に置換します。 %ProjectName% ：プロジェクト名に置換します。 空欄の場合は，%ProjectName%.err を指定したものとみなします。 cx コマンドの -Xerror_file オプションに相当します。 なお，本プロパティは，［エラー・メッセージ・ファイルを出力する］プロパティで［はい (-Xerror_file)］を選択した場合のみ表示します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	259 文字までの文字列

(8)［警告メッセージ］

警告メッセージに関する詳細情報の表示，および設定の変更を行います。

必ず表示させる警告メッセージ	警告レベルに関わらず，表示させたい警告メッセージの番号を指定します。 複数指定する場合は，メッセージ番号をカンマで区切って指定します（例：02042,02107）。 また，ハイフンを使用して，区間設定を行うこともできます（例：02222-02554,02699-02782）。 ［表示させない警告メッセージ］プロパティと同一の番号を指定した場合は，本プロパティを優先します。 cx コマンドの -Xwarning オプションに相当します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	2048 文字までの文字列

表示させない警告メッセージ	警告レベルに関わらず，表示させない警告メッセージの番号を指定します。 複数指定する場合は，メッセージ番号をカンマで区切って指定します（例：02042,02107）。 また，ハイフンを使用して，区間設定を行うこともできます（例：02222-02554,02699-02782）。 ［必ず表示させる警告メッセージ］プロパティと同一の番号を指定した場合は，［必ず表示させる警告メッセージ］プロパティを優先します。 cx コマンドの -Xno_warning オプションに相当します。	
	デフォルト	*共通オプションの設定値*
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	2048 文字までの文字列

（9）［その他］

アセンブルに関するその他の詳細情報の表示，および設定の変更を行います。

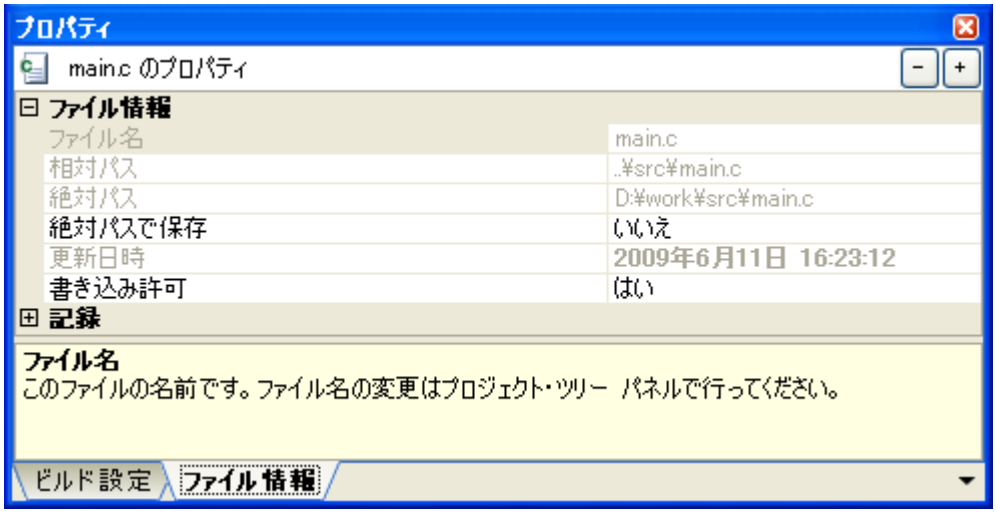
<table>
<tr><td>アセンブル前に実行するコマンド</td><td>
アセンブル処理前に実行するコマンドを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%AssembledFile% ：アセンブル時の出力ファイルの絶対パスに置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%InputFile% ：アセンブル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，アセンブル処理前に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。
<table>
<tr><td>デフォルト</td><td>- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
空欄
- 上記以外の場合
アセンブル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td rowspan="5">アセンブル後に実行するコマンド</td><td colspan="2">アセンブル処理後に実行するコマンドを指定します。
次のプレースホルダに対応しています。
%ActiveProjectDir% ：アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName% ：アクティブ・プロジェクト名に置換します。
%AssembledFile% ：アセンブル時の出力ファイルの絶対パスに置換します。
%BuildModeName% ：ビルド・モード名に置換します。
%InputFile% ：アセンブル対象ファイルの絶対パスに置換します。
%MainProjectDir% ：メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName% ：メイン・プロジェクト名に置換します。
%MicomToolPath% ：本製品のインストール・フォルダの絶対パスに置換します。
%Options% ：ビルド実行時のコマンド・ライン・オプションに置換します。
%OutputDir% ：出力フォルダの絶対パスに置換します。
%OutputFile% ：出力ファイルの絶対パスに置換します。
%Program% ：実行中のプログラム名に置換します。
%ProjectDir% ：プロジェクト・フォルダの絶対パスに置換します。
%ProjectName% ：プロジェクト名に置換します。
%TempDir% ：テンポラリ・フォルダの絶対パスに置換します。
%WinDir% ：Windows システム・フォルダの絶対パスに置換します。
先頭行に“#!python”と記述すると，2 行目から最終行までの内容を Python コンソールのスクリプトと判断し，アセンブル処理後に Python コンソールで実行します。
なお，スクリプト中にはプレースホルダの記述も可能です。
指定したコマンドはサブプロパティとして表示します。</td></tr>
<tr><td>デフォルト</td><td>- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
空欄
- 上記以外の場合
アセンブル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>［...］ボタンをクリックし，テキスト編集 ダイアログによる編集
サブプロパティはテキスト・ボックスによる直接入力も可能</td></tr>
<tr><td>指定可能値</td><td>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td rowspan="4">その他の追加オプション</td><td colspan="2">その他に追加するアセンブル・オプションを入力します。
ここで設定したオプションは，アセンブル・オプション群の最後に付加します。</td></tr>
<tr><td>デフォルト</td><td>- ［共通オプション］タブの［ビルド方法］カテゴリの［一括ビルドを行う］プロパティで［はい］を選択した場合
空欄
- 上記以外の場合
アセンブル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td>259 文字までの文字列</td></tr>
</table>

［ファイル情報］タブ

本タブでは，各ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

図 A―19　プロパティ パネル：［ファイル情報］タブ



［各カテゴリの説明］

（1）［ファイル情報］

ファイルに関する詳細情報の表示，および設定の変更を行います。

ファイル名	ファイル名を表示します。 ファイル名の変更は，プロジェクト・ツリーで行ってください。	
	デフォルト	*ファイル名*
	変更方法	変更不可
相対パス	プロジェクト・フォルダからの相対パス名を表示します。	
	デフォルト	*プロジェクト・フォルダからの相対パス名*
	変更方法	変更不可
絶対パス	ファイルの絶対パス名を表示します。	
	デフォルト	*ファイルの絶対パス名*
	変更方法	変更不可

絶対パスで保存	ファイルの場所を絶対パスで保存するかどうかを選択します。 なお，本プロパティは，依存関係ファイルについては表示しません。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ファイルの場所を絶対パスで保存します。
		いいえ	ファイルの場所を相対パスで保存します。
更新日時	ファイルを最後に変更した日時を表示します。		
	デフォルト	*ファイルの更新日時*	
	変更方法	変更不可	
書き込み許可	ファイルに書き込みを許可するかどうかを選択します。 なお，本プロパティは，依存関係ファイルについては表示しません。		
	デフォルト	はい（ファイルに書き込みを許可している場合） いいえ（ファイルに書き込みを許可していない場合）	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ファイルに書き込みを許可します。
		いいえ	ファイルに書き込みを許可しません。

(2)［記録］

記録に関する詳細情報の表示，および設定の変更を行います。

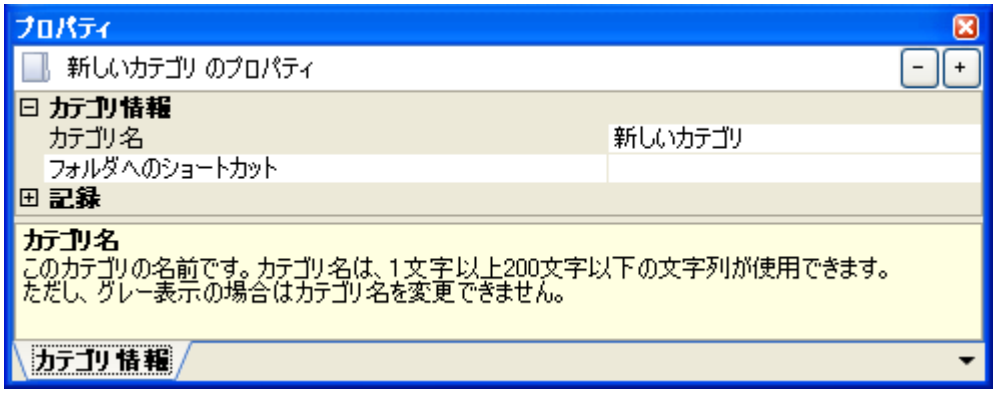
なお，本カテゴリは，依存関係ファイルについては表示しません。

メモ	ファイルにメモを追加します。 1 行に 1 項目ずつ指定します。 追加したメモはサブプロパティとして表示します。	
	デフォルト	メモ [*項目数*]
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	256 文字までの文字列 256 個まで指定可能です。

［カテゴリ情報］タブ

本タブでは，カテゴリ・ノード（ユーザが追加したファイルのカテゴリ），ファイル・ノード，ビルド・ツール生成ファイル・ノード，スタートアップ・ノードに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

図 A―20　プロパティ パネル：［カテゴリ情報］タブ



［各カテゴリの説明］

（1）［カテゴリ情報］

カテゴリに関する詳細情報の表示，および設定の変更を行います。

カテゴリ名	ファイルを分類するためのカテゴリ名を指定します。 なお，本プロパティは，ファイル・ノード，ビルド・ツール生成ファイル・ノード，スタートアップ・ノードについてはグレー表示となり，変更することはできません。	
	デフォルト	*ファイルのカテゴリ名*
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	1 ～ 200 文字までの文字列
フォルダへのショートカット	フォルダへのショートカットを指定します。 相対パスで指定した場合は，メイン・プロジェクト，またはサブプロジェクトのフォルダを基点とします。 なお，本プロパティは，ファイル・ノード，ビルド・ツール生成ファイル・ノード，スタートアップ・ノードについては表示されません。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，フォルダの参照 ダイアログによる編集
	指定可能値	247 文字までの文字列

(2)［記録］

記録に関する詳細情報の表示，および設定の変更を行います。

なお，本カテゴリは，ファイル・ノード，ビルド・ツール生成ファイル・ノード，スタートアップ・ノードについては表示しません。

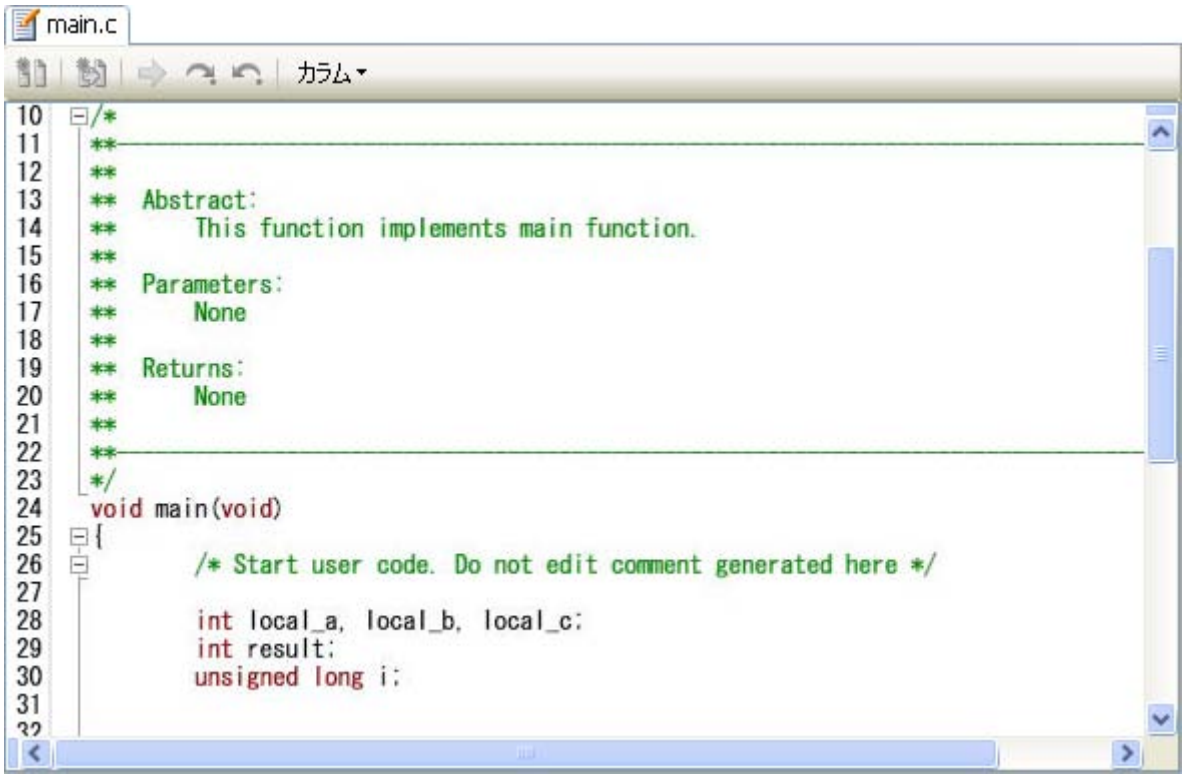
メモ	ファイルのカテゴリにメモを追加します。 1 行に 1 項目ずつ指定します。 追加したメモはサブプロパティとして表示します。	
	デフォルト	メモ [*項目数*]
	変更方法	[...] ボタンをクリックし，テキスト編集 ダイアログによる編集 サブプロパティはテキスト・ボックスによる直接入力も可能
	指定可能値	256 文字までの文字列 256 個まで指定可能です。

エディタ パネル

テキスト・ファイル／ソース・ファイルの表示／編集を行います。

本パネルについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

図 A—21　エディタ パネル



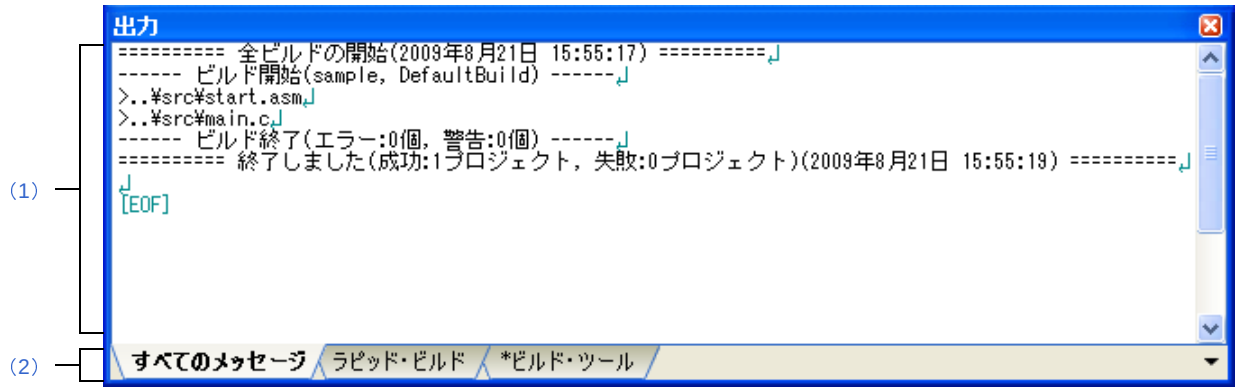
出力 パネル

ビルド・ツールから出力するメッセージの表示を行います。

メッセージは，出力元のツールごとに分類したタブ上でそれぞれ個別に表示します。

備考　ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

図 A―22　出力 パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［［ファイル］メニュー（出力 パネル専用部分）］
- ［［編集］メニュー（出力 パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［出力］の選択

［各エリアの説明］

（1）メッセージ・エリア

各ツールから出力したメッセージを表示します。

ビルド結果の表示では，ビルドを行うごとに，以前のメッセージをクリアしたのち新しいメッセージを表示します（［すべてのメッセージ］タブを除く）。

備考　メッセージの最大表示行数は 500000 行です。

500001 行以上のメッセージを出力した場合は，古い行から削除します。

なお，メッセージの表示色は，出力メッセージの種別により，次のように異なります（表示の際の文字色／背景色は，オプション ダイアログにおける［全般 - フォントと色］カテゴリ項目の設定に依存します）。

メッセージ種別	表示例（デフォルト）			説明
通常メッセージ	AaBbCc	文字色	黒	何らかの情報を通知する際に表示します。
		背景色	白	
警告メッセージ	AaBbCc	文字色	青	操作に対して，何らかの警告を通知する際に表示します。
		背景色	標準色	
エラー・メッセージ	AaBbCc	文字色	赤	致命的なエラー，または操作ミスにより実行が不可能な場合に表示します。
		背景色	薄グレー	

本エリアは，次の機能を備えています。

（a）タグ・ジャンプ

出力したメッセージをダブルクリック，またはメッセージにキャレットを合わせて［Enter］キーを押下することにより，エディタ パネルをオープンして該当ファイルの該当行番号を表示します。

これにより，ビルド時に出力したエラー・メッセージなどから，ソース・ファイルの該当するエラー行へジャンプすることができます。

（b）ヘルプの表示

警告メッセージ，またはエラー・メッセージを表示している行にキャレットがある状態で，コンテキスト・メニューの［メッセージに関するヘルプ］を選択するか，または［F1］キーを押下することにより，その行のメッセージに関するヘルプを表示します。

（c）ログの保存

［ファイル］メニュー→［名前を付けて出力 - *タブ名*を保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，現在選択しているタブ上に表示している内容をテキスト・ファイル（*.txt）に保存することができます（非選択状態のタブ上のメッセージは保存の対象となりません）。

（2）タブ選択エリア

メッセージの出力元を示すタブを選択します。

表示するタブは次のとおりです。

タブ名	説明
すべてのメッセージ	すべてのメッセージを出力順に一括して表示します（ラピッド・ビルド実行時を除く）。
ラピッド・ビルド	ラピッド・ビルドの実行により，ビルド・ツールから出力したメッセージを表示します。
ビルド・ツール	ビルド，リビルド，バッチ・ビルドの実行により，ビルド・ツールから出力したメッセージを表示します。

注意　新たなメッセージを非選択状態のタブ上に出力しても，自動的なタブの表示切り替えは行いません。
この場合，タブ名の先頭に  マークが付加し，新たなメッセージを出力していることを示します。

[[ファイル] メニュー（出力 パネル専用部分）]

出力 パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通です）。

出力 - *タブ名*を保存	現在選択しているタブ上に表示している内容を，前回保存したテキスト・ファイル（*.txt）に保存します（「(c) ログの保存」参照）。 なお，起動後に初めて本項目を選択した場合は，［名前を付けて*タブ名*を保存 ...］の選択と同等の動作となります。
名前を付けて出力 - *タブ名*を保存 ...	現在選択しているタブ上に表示している内容を，指定したファイル（*.txt）に保存するために，名前を付けて保存 ダイアログをオープンします（「(c) ログの保存」参照）。

[[編集] メニュー（出力 パネル専用部分）]

出力 パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効となります）。

コピー	選択している文字列をクリップ・ボードにコピーします。
すべて選択	本パネルに表示しているすべてのメッセージを選択状態にします。
検索 ...	検索・置換 ダイアログを［クイック検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを［一括置換］タブが選択状態でオープンします。

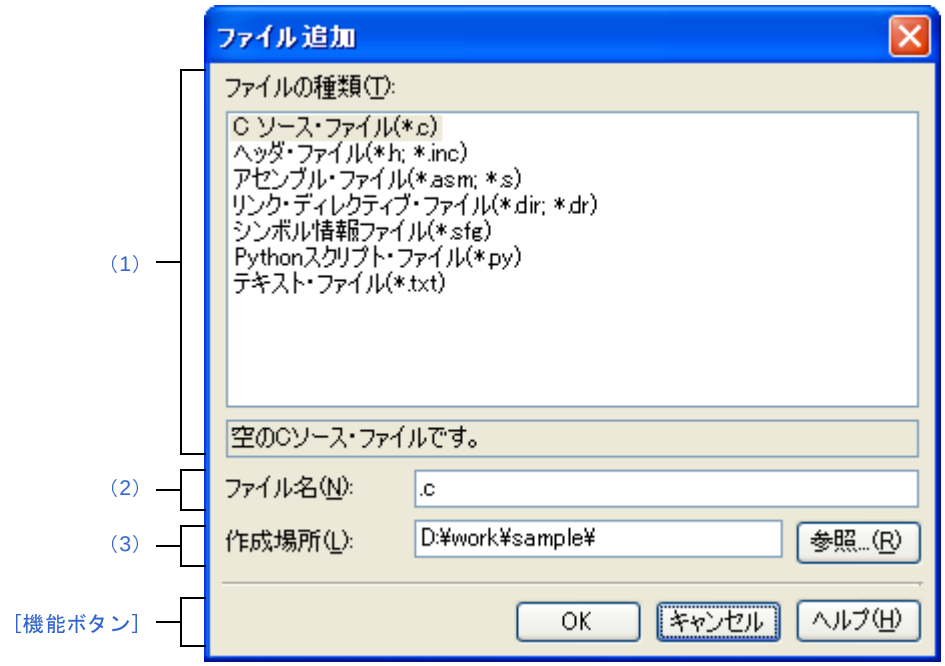
[コンテキスト・メニュー]

コピー	選択している文字列をクリップ・ボードにコピーします。
すべて選択	本パネルに表示しているすべてのメッセージを選択状態にします。
クリア	本パネルに表示しているすべてのメッセージを消去します。
タグ・ジャンプ	キャレット行にファイル名，行，桁の情報があれば，その位置へジャンプします。
メッセージに関するヘルプ	現在のキャレット位置のメッセージに関するヘルプを表示します。 ただし，警告メッセージ／エラー・メッセージのみが対象となります。

ファイル追加 ダイアログ

新規にファイルを作成し，プロジェクトへの追加を行います。

図 A—23　ファイル追加 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ［ファイル］メニュー→［追加］→［新しいファイルを追加 ...］を選択
- プロジェクト・ツリー パネルにおいて，プロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノード，カテゴリ・ノードのいずれかを選択したのち，コンテキスト・メニュー→［追加］→［新しいファイルを追加 ...］を選択

[各エリアの説明]

(1) [ファイルの種類] エリア

作成するファイルの種類を選択します。

ファイルの種類を選択すると，下部のボックスにその説明を表示します。

表示するファイルの種類を以下に示します。

- C ソース・ファイル（*.c）
- ヘッダ・ファイル（*.h; *.inc）
- アセンブル・ファイル（*.asm; *.s）
- リンク・ディレクティブ・ファイル（*.dir; *.dr）
- シンボル情報ファイル（*.sfg）
- Python スクリプト・ファイル（*.py）
- テキスト・ファイル（*.txt）

(2) [ファイル名] エリア

作成するファイルの名前を直接入力します。

デフォルトでは，“.txt” を表示します。

備考 拡張子を指定しなかった場合は，[ファイルの種類] エリアで選択した拡張子を付加します。
また，[ファイルの種類] エリアと異なる拡張子を指定した場合も，[ファイルの種類] エリアで選択した拡張子を付加します（例えば，ファイル名に “aaa.txt”，ファイルの種類に “C ソース・ファイル (*.c)” を指定した場合，ファイル名は “aaa.txt.c” となります）。

(3) [作成場所] エリア

ファイルの作成場所のパスをテキスト・ボックスに直接入力，または [参照 ...] ボタンから選択します。

デフォルトでは，プロジェクト・フォルダのパスを表示します。

ただし，カテゴリ・ノード（フォルダへのショートカットを設定し，そのフォルダが存在する場合のみ）のコンテキスト・メニューから本ダイアログをオープンした場合は，カテゴリに設定したフォルダのパスを表示します。

(a) ボタン

参照 ...	フォルダの参照 ダイアログをオープンします。 フォルダを選択すると，テキスト・ボックスにパスを追加します。

備考 1. テキスト・ボックスが空欄の場合は，プロジェクト・フォルダを指定したものとみなします。

2. 相対パスで指定した場合は，プロジェクト・フォルダからの相対パスとみなします。

備考 [ファイル名] エリア，[作成場所] エリアで指定可能な文字数は，パス名とファイル名をあわせて 259 文字までです。入力内容が正しくない場合，以下のメッセージを [ファイル名] エリアにツールチップ表示します。

メッセージ	説明
パスを含むファイル名が長すぎます。259 文字以内にしてください。	パスを含むファイル名が 259 文字を越えています。
指定したパスに存在しないフォルダが含まれています。	パスに存在しないフォルダが含まれています。
ファイル名，もしくは，パス名が不正です。文字 (¥ , /, :, *, ?, ", <, >, |) は使用できません。	不正なパスを含むファイル名を指定しました。 ファイル名，およびフォルダ名に文字 (¥ , /, :, *, ?, ", <, >, |) は使用できません。

[機能ボタン]

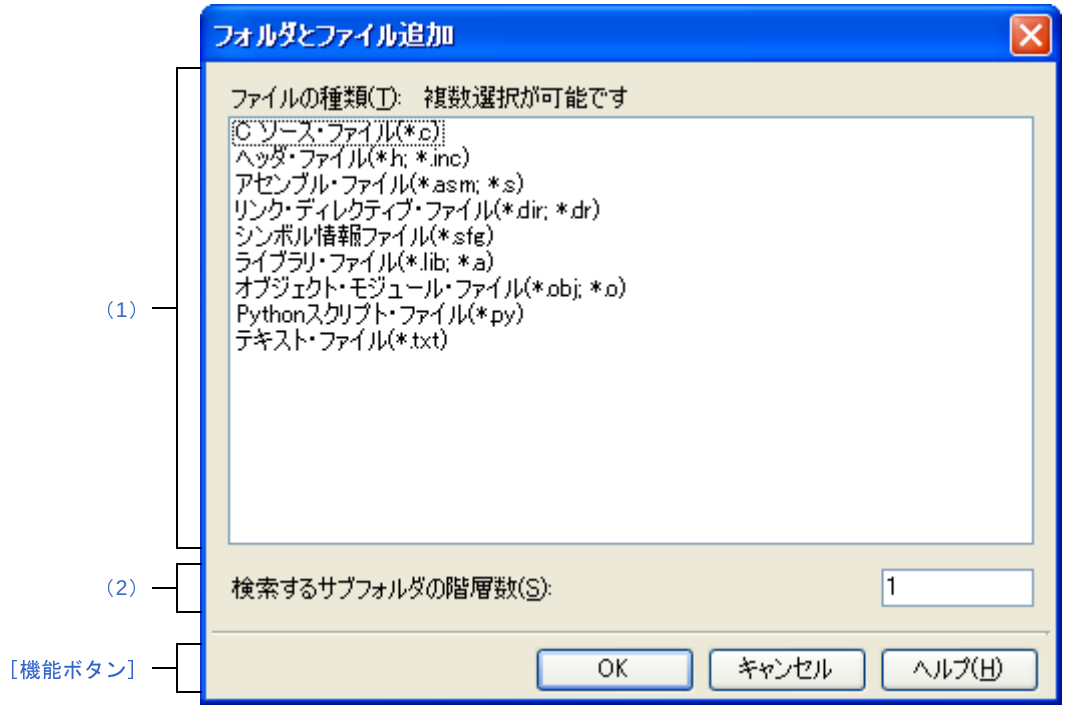
ボタン	機能
OK	入力したファイル名でファイルを作成して，プロジェクトに追加し，エディタ パネルでオープンします。そののち，本ダイアログをクローズします。
キャンセル	ファイルの生成を行わずに，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

フォルダとファイル追加 ダイアログ

既存のファイルとフォルダ構成のプロジェクトへの追加を行います。

フォルダはカテゴリとして追加します。

図 A―24　フォルダとファイル追加 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- エクスプローラなどからフォルダをドラッグし，プロジェクト・ツリー パネル上でドロップ

[各エリアの説明]

(1)［ファイルの種類］エリア

プロジェクトに追加するファイルの種類を選択します。

［Ctrl］キー＋左クリック，または［Shift］キー＋左クリックにより，複数選択することができます。

何も選択しない場合は，すべての種類を選択したものとみなします。

表示するファイルの種類を以下に示します。

- C ソース・ファイル（*.c）
- ヘッダ・ファイル（*.h; *.inc）
- アセンブル・ファイル（*.asm; *.s）
- リンク・ディレクティブ・ファイル（*.dir; *.dr）
- シンボル情報ファイル（*.sfg）
- ライブラリ・ファイル（*.lib; *.a）
- オブジェクト・モジュール・ファイル（*.obj; *.o）
- Python スクリプト・ファイル（*.py）
- テキスト・ファイル（*.txt）

(2)［検索するサブフォルダの階層数］エリア

プロジェクトに追加するサブフォルダの階層数を直接入力します。

デフォルトでは，“1”を表示します。

備考　入力可能な値は 10 までの 10 進数です。

入力内容が正しくない場合，以下のメッセージをツールチップ表示します。

メッセージ	説明
0 未満もしくは 10 を越える値を指定できません。	サブフォルダの指定階層数が 0 未満，または 10 を越えています。
10 進数で指定してください。	10 進数以外の数値や文字列を指定しました。

[機能ボタン]

ボタン	機能
OK	ドラッグ・アンド・ドロップしたフォルダとそのフォルダ下に存在するファイルをプロジェクトに追加します。そののち，本ダイアログをクローズします。
キャンセル	フォルダとファイルの追加を行わずに，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

文字列入力 ダイアログ

1 行分の文字列の入力，編集を行います。

図 A—25　文字列入力 ダイアログ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［共通オプション］タブの［警告メッセージ］カテゴリの［必ず表示させる警告メッセージ］，［表示させない警告メッセージ］，［その他］カテゴリの［ビルド・オプション一覧表示フォーマット］，［その他の追加オプション］
  - ［コンパイル・オプション］タブの［その他］カテゴリの［その他の追加オプション］
  - ［アセンブル・オプション］タブの［その他］カテゴリの［その他の追加オプション］
  - ［リンク・オプション］タブの［その他］カテゴリの［エントリ・シンボル］，［その他の追加オプション］
  - ［ROM 化オプション］タブの［セクション］カテゴリの［rompsec セクションの開始シンボル］
  - ［ライブラリ生成オプション］タブの［その他］カテゴリの［その他の追加オプション］
  - ［個別コンパイル・オプション］タブの［その他］カテゴリの［その他の追加オプション］
  - ［個別アセンブル・オプション］タブの［その他］カテゴリの［その他の追加オプション］
- リンク・ディレクティブ生成 ダイアログの［セグメント／セクション一覧］エリアでセグメント，またはセクションを選択したのち，［セグメント／セクションの詳細］エリアの［名前］において，［...］ボタンをクリック
- リンク・ディレクティブ生成 ダイアログの［セグメント／セクション一覧］エリアでセクションを選択したのち，［セグメント／セクションの詳細］エリアの［入力セクション名］において，［...］ボタンをクリック
- リンク・ディレクティブ生成 ダイアログの［シンボル一覧］エリアでシンボルを選択したのち，［シンボルの詳細］エリアの［名前］において，［...］ボタンをクリック
- リンク・ディレクティブ生成 ダイアログの［シンボル一覧］エリアでシンボルを選択したのち，［シンボルの詳細］エリアの［ベース・シンボル名］において，［...］ボタンをクリック

［各エリアの説明］

（1）文字列入力エリア

文字列の入力を行います。

（a）［文字列］

1 行分の文字列の入力を行います。

デフォルトでは，本ダイアログの呼び出し元の内容を反映します。

なお，改行することはできません。

備考　入力可能な文字数は，32767 文字までです。

入力内容が正しくない場合，以下のメッセージをツールチップ表示します。

メッセージ	説明
*呼び出し元で指定している最大文字数*文字を越える文字を指定できません。	入力した文字列の文字数が，呼び出し元で指定している最大文字数を越えています。

（b）ボタン

	本ダイアログの呼び出し元に指定可能なプレースホルダをポップアップ表示します（昇順）。 プレースホルダを選択すると，前後に“%”を付加して［文字列］に表示します。

注意　本ボタンは，本ダイアログの呼び出し元がプレースホルダに対応している場合のみ表示されます。

備考　指定可能なプレースホルダは，本ダイアログの呼び出し元によって異なります。

具体的なプレースホルダについては，呼び出し元の説明を参照してください。

［機能ボタン］

ボタン	機能
OK	入力した文字列を本ダイアログの呼び出し元に反映し，本ダイアログをクローズします。
キャンセル	入力した文字列を本ダイアログの呼び出し元に反映せずに，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

テキスト編集 ダイアログ

複数行のテキストの入力，編集を行います。

図 A―26　テキスト編集 ダイアログ（呼び出し元がプレースホルダに対応している場合）

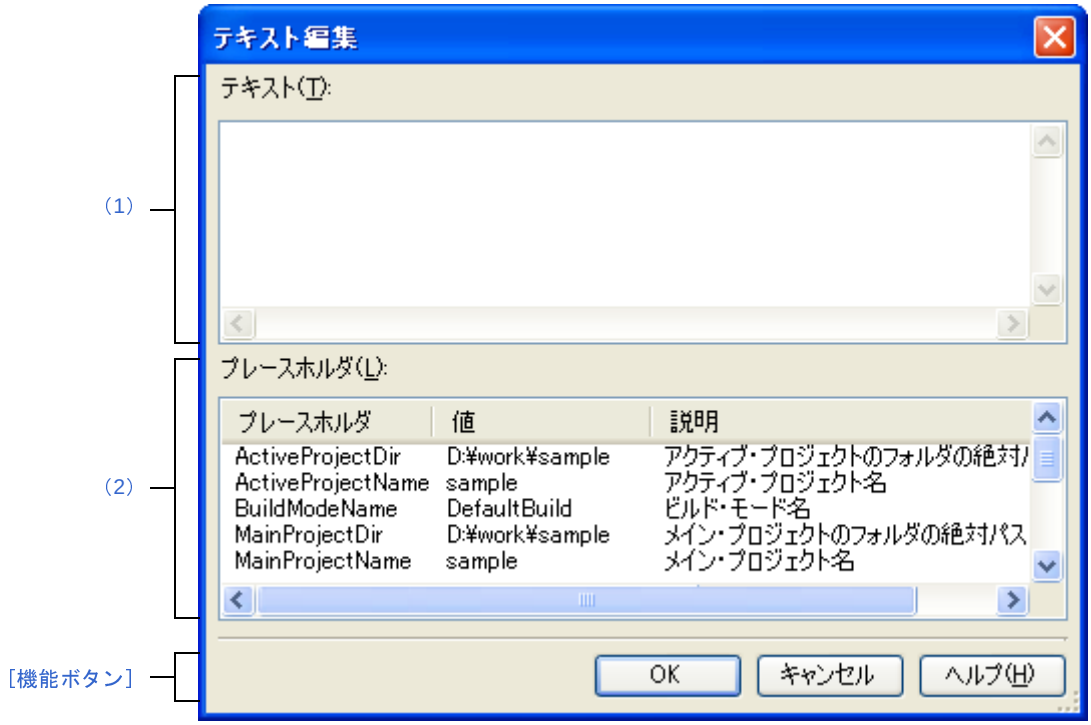
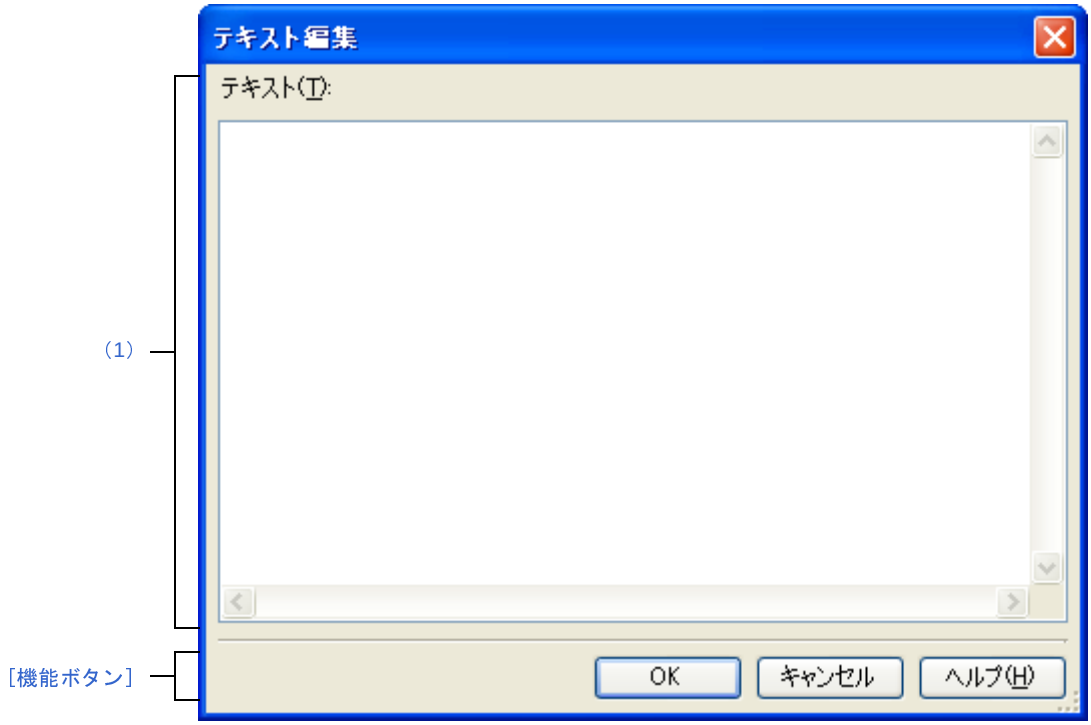


図 A―27　テキスト編集 ダイアログ（呼び出し元がプレースホルダに対応していない場合）



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［定義マクロ］，［よく使うオプション（アセンブル）］カテゴリの［定義マクロ］，［よく使うオプション（リンク）］カテゴリの［使用するライブラリ・ファイル］，［記録］カテゴリの［メモ］，［その他］カテゴリの［ビルド前に実行するコマンド］，［ビルド後に実行するコマンド］
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］，［定義解除マクロ］，［その他］カテゴリの［コンパイル前に実行するコマンド］，［コンパイル後に実行するコマンド］
  - ［アセンブル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］，［その他］カテゴリの［アセンブル前に実行するコマンド］，［アセンブル後に実行するコマンド］
  - ［リンク・オプション］タブの［ライブラリ］カテゴリの［使用するライブラリ・ファイル］，［その他］カテゴリの［リンク前に実行するコマンド］，［リンク後に実行するコマンド］
  - ［ROM 化オプション］タブの［セクション］カテゴリの［rompsec セクションに含めるデータ・セクション］，［rompsec セクションに含めるテキスト・セクション］
  - ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイルに変換するセクション］
  - ［ライブラリ生成オプション］タブの［その他］カテゴリの［ライブラリ生成前に実行するコマンド］，［ライブラリ生成後に実行するコマンド］
  - ［個別コンパイル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］，［定義解除マクロ］，［その他］カテゴリの［コンパイル前に実行するコマンド］，［コンパイル後に実行するコマンド］
  - ［個別アセンブル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］，［その他］カテゴリの［アセンブル前に実行するコマンド］，［アセンブル後に実行するコマンド］
  - ［ファイル情報］タブの［記録］カテゴリの［メモ］
  - ［カテゴリ情報］タブの［記録］カテゴリの［メモ］

［各エリアの説明］

(1) ［テキスト］

複数行のテキストの編集を行います。

デフォルトでは，本ダイアログの呼び出し元の内容を反映します。

備考　入力可能な行数は 65535 行まで，文字数は 65535 文字までです。

入力内容が正しくない場合，以下のメッセージをツールチップ表示します。

メッセージ	説明
*呼び出し元で指定している最大文字数*文字を越える文字を指定できません。制限を越えた行頭のかっこ内に今の文字数を表示しました。	入力した文字列の文字数が，呼び出し元で指定している最大文字数を越えています。

(2)［プレースホルダ］

本ダイアログの呼び出し元に指定可能なプレースホルダの一覧を表示します（昇順）。

行をダブルクリックすると，プレースホルダの前後に“%”を付加して［テキスト］に表示します。

(a)［プレースホルダ］

プレースホルダを表示します。

(b)［値］

プレースホルダの置換後の文字列を表示します。

(c)［説明］

プレースホルダの説明を表示します。

注意　本エリアは，本ダイアログの呼び出し元がプレースホルダに対応している場合のみ表示されます。

備考　指定可能なプレースホルダは，本ダイアログの呼び出し元によって異なります。
具体的なプレースホルダについては，呼び出し元の説明を参照してください。

［機能ボタン］

ボタン	機能
OK	入力したテキストを本ダイアログをオープンしたテキスト・ボックスに反映し，本ダイアログをクローズします。
キャンセル	入力したテキストを本ダイアログをオープンしたテキスト・ボックスに反映せずに，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

パス編集 ダイアログ

パス，またはパスを含むファイル名の編集，追加を行います。

図 A—28　パス編集 ダイアログ（パスを編集する場合）

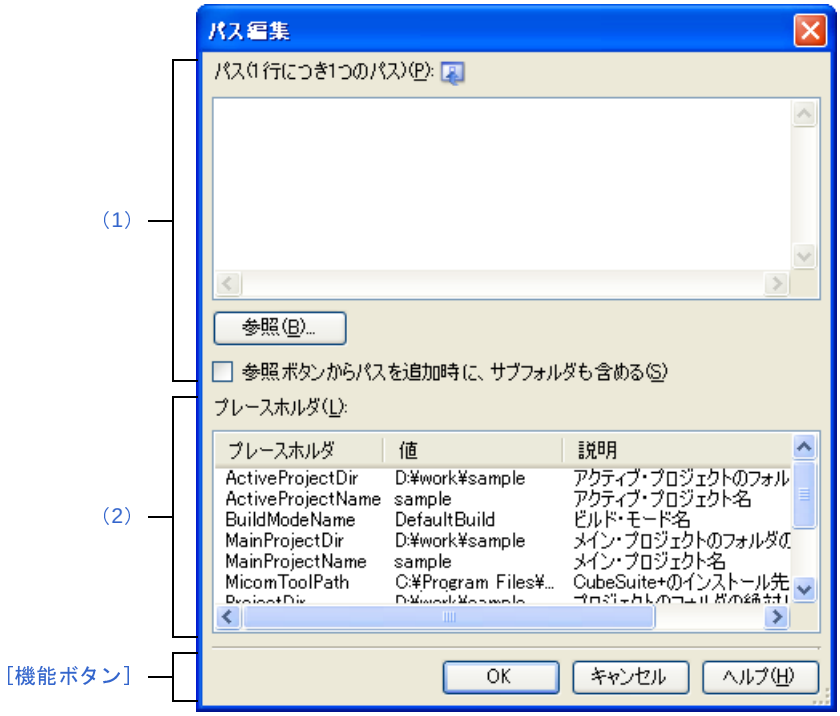
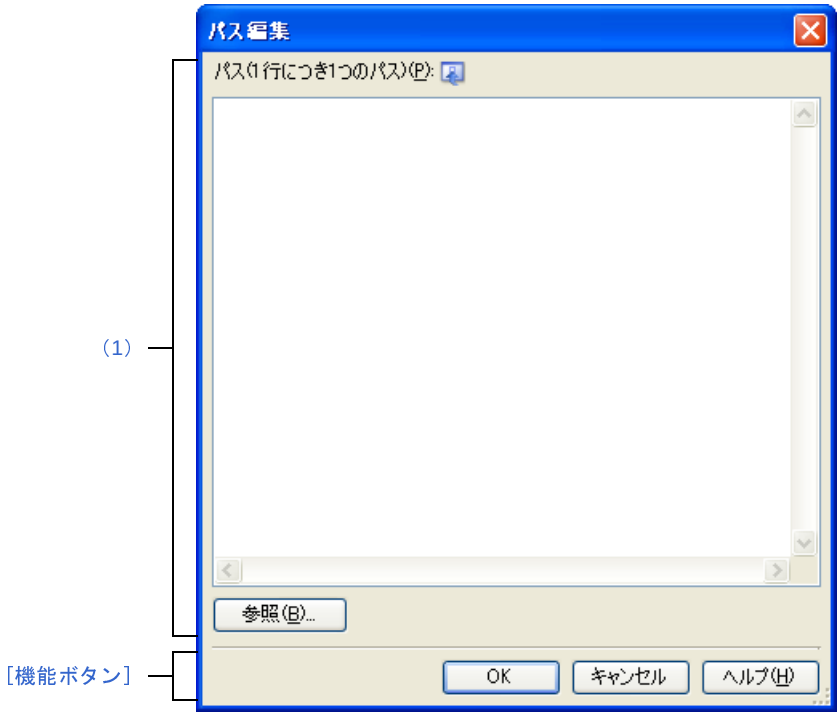


図 A—29　パス編集 ダイアログ（パスを含むファイル名を編集する場合）



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［追加のインクルード・パス］，［よく使うオプション（アセンブル）］カテゴリの［追加のインクルード・パス］，［よく使うオプション（リンク）］カテゴリの［追加のライブラリ・パス］
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］，［出力コード］カテゴリの［Far Jump ファイル名］
  - ［アセンブル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］
  - ［リンク・オプション］タブの［ライブラリ］カテゴリの［追加のライブラリ・パス］
  - ［個別コンパイル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］
  - ［個別アセンブル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］

［各エリアの説明］

（1）パス編集エリア

パス，またはパスを含むファイル名の編集，追加を行います。

（a）［パス (1 行につき 1 つのパス)］

直接入力により，パス，またはパスを含むファイル名の編集，追加を行います。

パス，またはパスを含むファイル名は複数行指定可能です。1 行につき 1 つのパス，またはパスを含むファイル名を指定してください。

デフォルトで，本ダイアログをオープンしたテキスト・ボックスの内容を反映します。

パスの追加は，以下の方法でも行うことができます。

- ［参照 ...］ボタンをクリックし，フォルダの参照 ダイアログによるフォルダの選択
- エクスプローラなどからフォルダをドラッグ・アンド・ドロップ

パスを含むファイル名の追加は，以下の方法でも行うことができます。

- ［参照 ...］ボタンをクリックし，Far Jump ファイルを指定 ダイアログによるファイルの選択
- エクスプローラなどからファイルをドラッグ・アンド・ドロップ

注意　絶対パスで非常に長いパスを相対パスで指定すると，［OK］ボタンのクリック時にエラーになる場合があります。その場合は，絶対パスで指定してください。

備考　入力可能な行数は 10000 行まで，文字数は Windows のパスの最大文字数までです。
入力内容が正しくない場合，以下のメッセージをツールチップ表示します。

メッセージ	説明
パスを指定してください。	空欄になっています。
パスが長すぎます。*呼び出し元で指定している最大文字数*文字以下のパスを指定してください。	パスを含むファイル名が呼び出し元で指定している最大文字数を越えています。
指定したパスに存在しないフォルダが含まれています。	パスに存在しないフォルダが含まれています。
ファイル名，もしくは，パス名が不正です。文字 (¥ , /, :, *, ?, ", <, >, |) は使用できません。	不正なパスを含むファイル名を指定しました。 ファイル名，およびフォルダ名に文字 (¥ , /, :, *, ?, ", <, >, |) は使用できません。
*呼び出し元で指定している最大パス数，またはファイル数*行を越える行を指定できません。	入力したパス，またはファイルの総数が呼び出し元で指定している最大パス数，またはファイル数を越えています。

(b) ボタン

参照 ...	- パスを追加する場合 フォルダの参照 ダイアログをオープンします。 フォルダを選択すると，［パス (1 行につき 1 つのパス)］にパスを追加します。 - パスを含むファイル名を追加する場合 Far Jump ファイルを指定 ダイアログをオープンします。 ファイルを選択すると，［パス (1 行につき 1 つのパス)］にファイル名を追加します。

(c)［参照ボタンからパスを追加時に、サブフォルダも含める］

このチェック・ボックスをチェックしたのち，［参照 ...］ボタンからパスの指定を行うと，サブフォルダも含めて［パス (1 行につき 1 つのパス)］にパスを追加します（5 階層まで）。

(2)［プレースホルダ］

本ダイアログの呼び出し元に指定可能なプレースホルダの一覧を表示します（昇順）。
行をダブルクリックすると，プレースホルダの前後に“%”を付加してパス編集エリアに表示します。

(a)［プレースホルダ］

プレースホルダを表示します。

(b)［値］

プレースホルダの置換後の文字列を表示します。

(c)［説明］

プレースホルダの説明を表示します。

注意　本エリアは，本ダイアログの呼び出し元がプレースホルダに対応している場合のみ表示されます。

備考　指定可能なプレースホルダは，本ダイアログの呼び出し元によって異なります。
具体的なプレースホルダについては，呼び出し元の説明を参照してください。

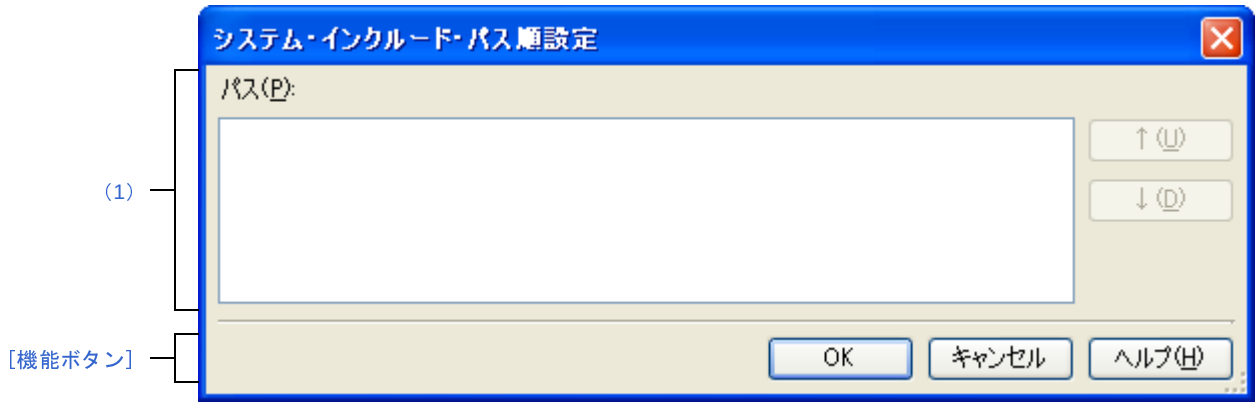
[機能ボタン]

ボタン	機能
OK	入力したパスを本ダイアログをオープンしたテキスト・ボックスに反映し，本ダイアログをクローズします。
キャンセル	入力したパスを本ダイアログをオープンしたテキスト・ボックスに反映せずに，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

システム・インクルード・パス順設定 ダイアログ

コンパイラに対して指定するシステム・インクルード・パスの参照，および指定順の設定を行います。

図 A―30　システム・インクルード・パス順設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［システム・インクルード・パス］，［よく使うオプション（アセンブル）］カテゴリの［システム・インクルード・パス］
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［システム・インクルード・パス］
  - ［アセンブル・オプション］タブの［プリプロセス］カテゴリの［システム・インクルード・パス］

［各エリアの説明］

(1) パス一覧表示エリア

コンパイラに対して指定するシステム・インクルード・パスの一覧を表示します。

(a) ［パス］

システム・インクルード・パス名の一覧を，コンパイラへの指定順に表示します。

デフォルトでは，プロジェクトに登録している順番となります。

パスの表示順を変更することにより，コンパイラへの指定順を設定することができます。

表示順の変更は，［↑］，および［↓］ボタン，またはパス名のドラッグ・アンド・ドロップにより行います。

備考 1. パス名にマウス・カーソルをあわせると，そのパスを絶対パスでポップアップ表示します。

2. 新規に追加したシステム・インクルード・パスは，一覧の最後のパスの次に追加します。

3. パス名をドラッグ・アンド・ドロップする際，連続して並んでいるパス名のみ複数選択することができます。

（b）ボタン

↑	選択しているパスを上へ移動します。
↓	選択しているパスを下へ移動します。

備考　上記のボタンは，パスを選択していない場合は無効となります。

［機能ボタン］

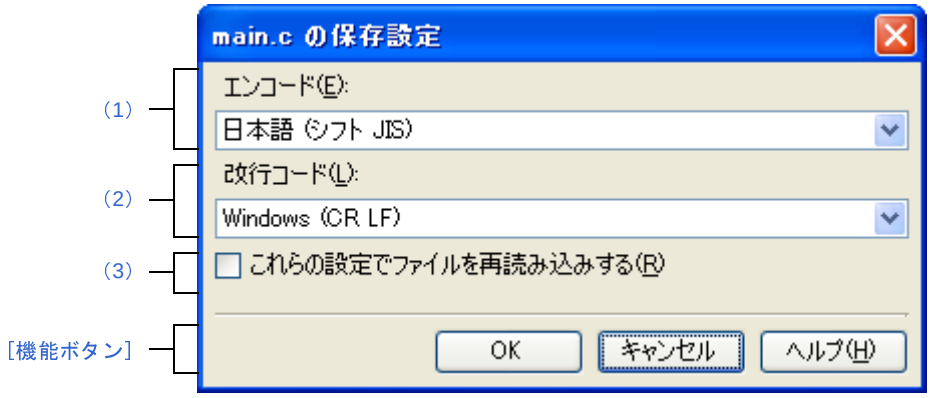
ボタン	機能
OK	コンパイラへのパスの指定順をパス一覧表示エリアの表示順に設定し，本ダイアログをクローズします。
キャンセル	パスの指定順の設定をキャンセルし，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

ファイルの保存設定 ダイアログ

エディタ パネルで編集中のファイルのエンコードと改行コードの設定を行います。

備考　タイトルバーには，設定対象ファイルの名前を表示します。

図 A—31　ファイルの保存設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- エディタ パネルにフォーカスがある状態で，［ファイル］メニュー→［*ファイル名*の保存設定 ...］を選択

［各エリアの説明］

(1)［エンコード］

設定するエンコードをドロップダウン・リストにより選択します。

ドロップダウン・リストの項目は，以下の順番で表示します。

ただし，同じエンコード名，および現在の OS が対応していないエンコード名は表示しません。

- *現在のファイルのエンコード名*（デフォルト）
- *現在の OS の既定のエンコード名*
- *最近使用した エンコード名（最大4 件）*
- *現在のロケールでよく使用されているエンコード名*

（例：ロケールが日本の場合）

- 日本語 (シフト JIS)
- 日本語 (JIS 1 バイト カタカナ可 - SO/SI)
- 日本語 (EUC)

- Unicode (UTF-8)

- 現在の OS が対応する上記以外のエンコード名（アルファベット順）

(2)［改行コード］

設定する改行コードをドロップダウン・リストにより選択します。

以下の項目を選択することができます。

- 現在の改行コードを維持
- Windows (CR LF)
- Macintosh (CR)
- Unix (LF)

デフォルトでは，現在の改行コードを選択します。

(3)［これらの設定でファイルを再読み込みする］

☑	［OK］ボタンをクリックした際に，指定したエンコード，および改行コードでファイルの再読み込みを行います。
☐	［OK］ボタンをクリックした際に，ファイルの再読み込みを行いません（デフォルト）。

［機能ボタン］

ボタン	機能
OK	指定したエンコード，および改行コードを対象ファイルに設定し，本ダイアログをクローズします。 ［これらの設定でファイルを再読み込みする］をチェックした場合，指定したエンコード，および改行コードを対象ファイルに設定し，ファイルを読み込み直したのち，このダイアログをクローズします。
キャンセル	設定を無効とし，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

リンク・ディレクティブ生成 ダイアログ

指定したメモリ，セグメント，セクション，シンボルの配置情報から，リンク・ディレクティブ・ファイルを生成します。

図 A―32 リンク・ディレクティブ生成 ダイアログ

ここでは，次の項目について説明します。

- [オープン方法]
- [各エリアの説明]
- [機能ボタン]

[オープン方法]

- プロジェクト・ツリー パネルにおいて，ビルド・ツール・ノードを選択したのち，コンテキスト・メニュー→［リンク・ディレクティブ・ファイルを生成する ...］を選択

[各エリアの説明]

(1)［セグメント／セクション一覧］エリア

デバイスのメモリ配置情報と，現在設定しているセグメントとセクションの一覧を表示します。

(a)［メモリ／名前］

メモリ領域，セグメント，およびセクションの名前を表示します。

メモリ領域は，以下のうち，該当するメモリ領域の名前を表示します。

- 内蔵 ROM 領域
- ノン・マップ領域
- 内蔵 RAM 領域
- データフラッシュ領域

セグメント，およびセクションについては，この項目を直接編集することができます。

セグメント名，およびセクション名を変更すると，［セグメント／セクションの詳細］エリアの［名前］も変更します。

注意　セグメント名，およびセクション名は，予約セクションの扱いによって，編集できない場合があります。詳細については，［セグメント／セクションの詳細］エリアの備考を参照してください。

(b)［開始アドレス］

メモリ領域，セグメント，セクションの開始アドレスを表示します。

セグメント，およびセクションについては，この項目を直接編集することができます。

開始アドレスを変更すると，［セグメント／セクションの詳細］エリアの［開始アドレス］も変更します。

(c)［終了アドレス］

メモリ領域の終了アドレスを表示します。

セグメント，およびセクションの行については，“－”を表示します。

（d）ボタン

セグメント追加	一覧で選択している行の直下に，新しいセグメントを追加します。 セグメント名は，デフォルトで“NewSegment_*XXX*”となります（*XXX*：0 ～ 255 の 10 進数）。 セグメントの詳細設定は，［セグメント／セクションの詳細］エリアで行います。 なお，このボタンは，セクションの行を選択している場合，および一覧に 256 個のセグメントを登録している場合は無効となります。
セクション追加	一覧で選択している行の直下に，新しいセクションを追加します。 セクション名は，デフォルトで“NewSection_*XXX*”となります（*XXX*：0 ～ 255 の 10 進数）。 セクションの詳細設定は，［セグメント／セクションの詳細］エリアで行います。 なお，このボタンは，一覧に 256 個のセクションを登録している場合は無効となります。
削除	一覧で選択しているセグメント，またはセクションを削除します。 セグメントを削除する場合は，セグメントに含まれているセクションも削除します。

また，このエリアは，次の機能を備えています。

- 行の展開／折りたたみ表示の切り替え

  行をダブルクリック，または行の先頭にある⊞マーク／⊟マークをクリックすることにより，各行の展開／折りたたみ表示の切り替えを行うことができます。

- セグメント，およびセクションの行の移動

  ドラッグ・アンド・ドロップにより，セグメント，およびセクションの行を移動することができます。

  備考　セグメントを移動する場合は，セグメントに含まれるセクションも移動します。

- セグメント，およびセクションのコピー

  セグメント，またはセクションを選択したのち，［Ctrl］＋［C］キーの押下によりコピー，［Ctrl］＋［V］キーの押下により貼り付けを行うことができます。
  貼り付け位置は，［Ctrl］＋［V］キーの押下時に選択している行の直下となります。
  コピー後のセグメント，およびセクションの名前には，先頭に“Copy_”を付加します。

  備考 1.　セグメントをコピーする場合，セグメントに含まれるセクションはコピーしません。
  2.　コピー後のセグメント，およびセクションの開始アドレスは，空欄となります。
  3.　コピー先のセグメントの属性により，コピーできない場合は，エラーとなります。

（2）［セグメント／セクションの詳細］エリア

［セグメント／セクション一覧］エリアで選択したセグメント／セクションの詳細情報の表示，および編集を行います。

(a) セグメントの詳細情報

名前	セグメント名を指定します。 使用可能な文字は，数字（0 ～ 9），英大文字（A ～ Z），英小文字（a ～ z），アンダスコア（_），ドット（.），スラッシュ（/），円マーク，バックスラッシュ（\）です。		
	デフォルト	NewSegment_*XXX*（*XXX*：0 ～ 255 の 10 進数）	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集	
	指定可能値	1022 文字までの文字列	
属性	セグメントの属性を選択します。 セグメントが予約セクションを含んでいる場合は，そのセクションの属性により，セグメントの属性も設定できるものが限られる場合があります。その場合，設定できない属性については，ドロップダウン・リストに項目を表示しません。		
	デフォルト	- 内蔵 ROM 領域，またはノン・マップ領域に追加した場合 実行可能 (RX) - 内蔵 RAM 領域に追加した場合 リード／ライト可能 (RW) - データフラッシュ領域に追加した場合 リード・オンリー (R)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	実行可能 (RX)	読み出し，実行が可能なセグメントに設定します。
		リード・オンリー (R)	読み出しが可能なセグメントに設定します。
		リード／ライト可能 (RW)	読み出し，書き込みが可能なセグメントに設定します。
		すべて可能 (RWX)	読み出し，書き込み，実行が可能なセグメントに設定します。
開始アドレス	セグメントを配置する開始アドレスを指定します。 空欄の場合は，コンパイラのリンク機能により直前のセグメントの後に配置します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数）	
最大メモリ・サイズ	セグメントの最大メモリ・サイズを指定します。 空欄の場合は，コンパイラのリンク機能により 0x100000（バイト）として扱われます。 リンク時に，指定した最大メモリ・サイズを越えた場合は，エラーとなります。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数）	

ホール・サイズ	セグメント間のホール・サイズを指定します。 空欄の場合は，コンパイラのリンク機能により 0x0（バイト）として扱われます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数）
フィリング値	セグメント間のホールを埋める値（フィリング値）を指定します。 空欄の場合は，コンパイラのリンク機能により 0x0000 として扱われます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0000 ～ 0xFFFF（16 進数）
整列条件	セグメントの整列条件（アライメント値）を指定します。 奇数の値を指定した場合は，自動的に 1 を足して偶数の値に変更します。 空欄の場合は，コンパイラのリンク機能により 0x8 として扱われます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0 ～ 0xFF（16 進数）

（b）セクションの詳細情報

名前	セクション名を指定します。 使用可能な文字は，数字（0 ～ 9），英大文字（A ～ Z），英小文字（a ～ z），アンダスコア（_），ドット（.），スラッシュ（/），円マーク，バックスラッシュ（\）です。	
	デフォルト	NewSection_*XXX*（*XXX*：0 ～ 255 の 10 進数）
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	1022 文字までの文字列
種類	セクションの種類を選択します。 オブジェクト・ファイル内に実際の値を持っているセクション（.text，.data 等）の場合は［初期値あり (PROGBITS)］，実際の値を持っていないセクション（.bss，.sbss 等）の場合は［初期値なし (NOBITS)］を選択します。	
	デフォルト	初期値あり (PROGBITS)
	変更方法	ドロップダウン・リストによる選択
	指定可能値	初期値あり (PROGBITS)：初期値ありセクションに設定します。
		初期値なし (NOBITS)：初期値なしセクショ 3 ンに設定します。

属性	セクションの属性を選択します。		
	デフォルト	- 親セグメントの属性が［実行可能 AX)］の場合 実行可能 (AX) - 親セグメントの属性が［リード・オンリー (A)］の場合 リード・オンリー (A) - 親セグメントの属性が［リード／ライト可能 (AW)］の場合 リード／ライト可能 (AW) - 親セグメントの属性が［すべて可能 (AWX)］の場合 すべて可能 (AWX)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	実行可能 (AX)	メモリを占有し，実行が可能なセクションに設定します。 なお，この項目は，親セグメントの属性が［リード・オンリー (R)］の場合は表示しません。
		リード・オンリー (A)	メモリを占有するセクションに設定します。
		リード／ライト可能 (AW)	メモリを占有し，書き込みが可能なセクションに設定します。 なお，この項目は，親セグメントの属性が［リード／ライト可能 (RW)］，または［すべて可能 (RWX)］の場合のみ表示します。
		GP 相対 1 命令アクセス (AWG)	メモリを占有し，書き込みが可能，およびグローバル・ポインタ（gp）と 16 ビットのディスプレースメントを用いて参照が可能なメモリ範囲内に割り付けるセクションに設定します。 なお，この項目は，親セグメントの属性が［リード／ライト可能 (RW)］，または［すべて可能 (RWX)］の場合のみ表示します。
		すべて可能 (AWX)	メモリを占有し，書き込み，実行が可能なセクションに設定します。 なお，この項目は，親セグメントの属性が［すべて可能 (RWX)］の場合のみ表示します。
開始アドレス	セクションを配置する開始アドレスを指定します。 空欄の場合は，コンパイラのリンク機能により直前のセクションの後に配置します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数）	

ホール・サイズ	セクション間のホール・サイズを指定します。 空欄の場合は，コンパイラのリンク機能により 0x0（バイト）として扱われます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数）
整列条件	セクションの整列条件を指定します。 奇数の値を指定した場合は，自動的に 1 を足して偶数の値に変更します。 空欄の場合は，コンパイラのリンク機能により 0x4 として扱われます。 ただし，セクション名が“.tidata.byte”，または“.tibss.byte”の場合は，奇数の値を指定することができます。また，空欄の場合は，コンパイラのリンク機能により 0x1 として扱われます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0 ～ 0xFF（16 進数）
入力セクション名	入力セクション名を指定します。 使用可能な文字は，数字（0 ～ 9），英大文字（A ～ Z），英小文字（a ～ z），アンダスコア（_），ドット（.），スラッシュ（/），円マーク，バックスラッシュ（\）です。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	1022 文字までの文字列
オブジェクト・ファイル名	入力セクションを含んでいるオブジェクト・ファイル名を指定します。 指定したオブジェクト・ファイル名はサブプロパティとして表示します。	
	デフォルト	オブジェクト・ファイル名 [*設定数*]
	変更方法	［...］ボタンをクリックし，オブジェクト・ファイル指定 ダイアログによる編集

備考 予約セクションについては，以下のように扱います。

- コンパイラで予約セクションとして定義しているセクションを［名前］，または［入力セクション名］に指定した場合，［種類］，および［属性］は編集不可となり，自動で値を設定します。
  予約セクション名と自動で設定する値の組み合わせを，以下に示します。

予約セクション名	種類	属性
.pro_epi_runtime	初期値あり (PROGBITS)	実行可能 (AX)
.text	初期値あり (PROGBITS)	実行可能 (AX)
.data	初期値あり (PROGBITS)	リード／ライト可能 (AW)
.sedata	初期値あり (PROGBITS)	リード／ライト可能 (AW)
.sidata	初期値あり (PROGBITS)	リード／ライト可能 (AW)

予約セクション名	種類	属性
.tidata	初期値あり (PROGBITS)	リード／ライト可能 (AW)
.tidata.byte	初期値あり (PROGBITS)	リード／ライト可能 (AW)
.tidata.word	初期値あり (PROGBITS)	リード／ライト可能 (AW)
.bss	初期値なし (NOBITS)	リード／ライト可能 (AW)
.sebss	初期値なし (NOBITS)	リード／ライト可能 (AW)
.sibss	初期値なし (NOBITS)	リード／ライト可能 (AW)
.tibss	初期値なし (NOBITS)	リード／ライト可能 (AW)
.tibss.byte	初期値なし (NOBITS)	リード／ライト可能 (AW)
.tibss.word	初期値なし (NOBITS)	リード／ライト可能 (AW)
.sdata	初期値あり (PROGBITS)	GP 相対 1 命令アクセス (AWG)
.sbss	初期値あり (PROGBITS)	GP 相対 1 命令アクセス (AWG)
.const	初期値あり (PROGBITS)	リード・オンリー (A)
.sconst	初期値あり (PROGBITS)	リード・オンリー (A)

- 以下の予約セクションは，リンカにより，割り当て可能なセグメント名が固定されています。

セクション名	セグメント名
.sidata，.sibss，.tidata，.tibss，.tidata byte，.tibss.byte，.tidata.word，.tibss.word	SIDATA
.sedata，.sebss	SEDATA
.sconst	SCONST

これらのセクション名を［名前］に指定した場合，親セグメントの名前を参照します。
なお，これらのセクションは，セグメント内で移動することはできますが，ほかのセグメントへ移動することはできません。

- 以下の予約セクションは，リンカにより，出力セクション名と入力セクション名の対応が固定されているため，入力セクション名を省略してもリンカによって自動で割り付けられます。
  .pro_epi_runtime，.tidata，.tibss，.tidata.byte，.tibss.byte，.tidata.word，.sidata，.sibss，.sedata，.sebss

(3)［シンボル一覧］エリア

現在設定しているシンボルの一覧を表示します。

(a)［名前］

シンボルの名前を表示します。
この項目は直接編集することもできます。
シンボル名を変更すると，［シンボルの詳細］エリアの［名前］も変更します。

(b)［種別］

シンボルの種別を表示します。

この項目は直接編集することもできます。

種別を変更すると，［シンボルの詳細］エリアの［種別］も変更します。

(c)［アドレス］

シンボルを配置する開始アドレスを表示します。

この項目は直接編集することもできます。

アドレスを変更すると，［シンボルの詳細］エリアの［アドレス］も変更します。

(d) ボタン

シンボル追加	一覧で選択している行の直下に，新しいシンボルを追加します。 シンボル名は，デフォルトで“NewSymbol_*XXX*”となります（*XXX*：0 ～ 255 の 10 進数）。 シンボルの詳細設定は，［シンボルの詳細］エリアで行います。 なお，このボタンは，一覧に 256 個のシンボルを登録している場合は無効となります。
シンボル削除	一覧で選択しているシンボルを削除します。

また，このエリアは，次の機能を備えています。

- シンボルの行の移動

ドラッグ・アンド・ドロップにより，シンボルの行を移動することができます。

(4)［シンボルの詳細］エリア

［シンボル一覧］エリアで選択したシンボルの詳細情報の表示，および編集を行います。

名前	シンボル名を指定します。 使用可能な文字は，数字（0 ～ 9），英大文字（A ～ Z），英小文字（a ～ z），アンダスコア（_），ドット（.），スラッシュ（/），円マーク，バックスラッシュ（\）です。	
	デフォルト	NewSymbol_*XXX*（*XXX*：0 ～ 255 の 10 進数）
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	1022 文字までの文字列

種別	シンボルの種別を選択します。		
	デフォルト	TP シンボル (%TP_SYMBOL)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	TP シンボル (%TP_SYMBOL)	TP シンボルに設定します。
		GP シンボル (%GP_SYMBOL)	GP シンボルに設定します。
		EP シンボル (%EP_SYMBOL)	EP シンボルに設定します。
ベース・シンボル名	ベース・シンボル（GP シンボル値を定める際に用いる TP シンボル）として，すでに存在する TP シンボルを指定します。 ベース・シンボル名を指定すると，TP シンボル値からのオフセット値が GP シンボル値となります。 使用可能な文字は，数字（0 ～ 9），英大文字（A ～ Z），英小文字（a ～ z），アンダスコア（_），ドット（.），スラッシュ（/），円マーク，バックスラッシュ（\）です。 なお，このプロパティは，［種別］プロパティで［GP シンボル (%GP_SYMBOL)］を選択した場合のみ表示します。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集	
	指定可能値	1022 文字までの文字列	
アドレス	シンボルを配置する開始アドレスを指定します。 空欄の場合は，コンパイラのリンク機能のアドレス決定規則により，自動的にアドレスが割り振られます。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF（16 進数）	
整列条件	シンボルの整列条件（アライメント値）を指定します。 奇数の値を指定した場合は，自動的に 1 を足して偶数の値に変更します。 空欄の場合は，コンパイラのリンク機能により 0x4 として扱われます。		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0 ～ 0xFF（16 進数）	
セグメント名	TP シンボル値，GP シンボル値の参照対象とするセグメント名を指定します。 指定したセグメント名はサブプロパティとして表示します。 なお，このプロパティは，［種別］プロパティで［EP シンボル (%EP_SYMBOL)］を選択した場合は表示しません。		
	デフォルト	セグメント名 [*設定数*]	
	変更方法	[...] ボタンをクリックし，セグメント指定 ダイアログによる編集	

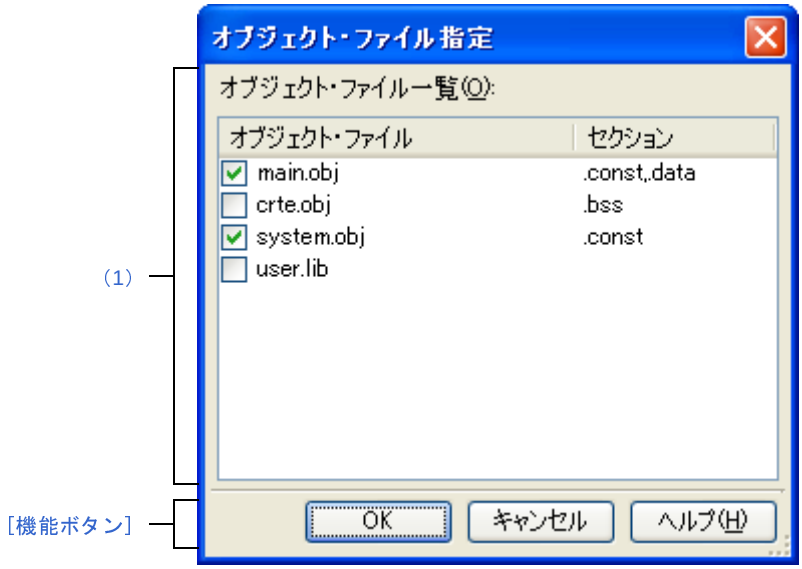
［機能ボタン］

ボタン	機能
シンボル	［シンボル一覧］エリア，および［シンボルの詳細］エリアの表示／非表示を切り替えます。
生成	指定したメモリ，セグメント／セクション，シンボルの配置情報を元に，リンク・ディレクティブ・ファイル（ファイル名：*プロジェクト名*.dir）を生成し，プロジェクトに登録します。 リンク・ディレクティブ・ファイルの生成先は，プロジェクト・フォルダとなります。 生成したリンク・ディレクティブ・ファイルは，プロジェクト・ツリーのファイル・ノードにも表示します。 生成したリンク・ディレクティブ・ファイルはビルド対象となります。すでにリンク・ディレクティブ・ファイルをプロジェクトに登録していた場合，登録済みのリンク・ディレクティブ・ファイルはビルド対象外となります。
閉じる	本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

オブジェクト・ファイル指定 ダイアログ

本ダイアログの呼び出し元に設定するオブジェクト・ファイルを，プロジェクトに追加しているオブジェクト・ファイル，およびライブラリ・ファイルの中から選択します。

図 A―33　オブジェクト・ファイル指定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- リンク・ディレクティブ生成 ダイアログの［セグメント／セクション一覧］エリアでセクションを選択したのち，［セグメント／セクションの詳細］エリアの［オブジェクト・ファイル名］において，［...］ボタンをクリック

［各エリアの説明］

(1)［オブジェクト・ファイル一覧］エリア

リンク・ディレクティブ生成 ダイアログをオープンしたプロジェクトに追加しているオブジェクト・ファイル，およびライブラリ・ファイルと，リンク・ディレクティブ生成 ダイアログでそれらを指定しているセクションの一覧を表示します。

(a)［オブジェクト・ファイル］

以下のファイル名一覧を表示します。

本ダイアログをオープンしたリンク・ディレクティブ生成 ダイアログの［セグメント／セクションの詳細］エリアの［オブジェクト・ファイル名］に設定するファイルをチェック・ボックスにより選択します。

- プロジェクトに追加しているソース・ファイルから生成するオブジェクト・モジュール・ファイル
- プロジェクト・ツリーに直接追加したオブジェクト・モジュール・ファイル
- プロジェクト・ツリーに直接追加したライブラリ・ファイル

備考 1. ファイル名にマウス・カーソルをあわせると，そのファイルの絶対パスをポップアップ表示します。

2. 本ダイアログをオープンしたリンク・ディレクティブ生成 ダイアログの［セグメント／セクションの詳細］エリアの［オブジェクト・ファイル名］において，すでにオブジェクト・ファイルを設定していた場合は，該当するオブジェクト・ファイルのチェック・ボックスはデフォルトでチェック状態となります。

(b)［セクション］

リンク・ディレクティブ生成 ダイアログで該当オブジェクト・ファイルを指定しているセクションを表示します。

オブジェクト・ファイルを複数のセクションから指定している場合は，カンマで区切って表示します。

オブジェクト・ファイルを指定しているセクションが存在しない場合は，空欄となります。

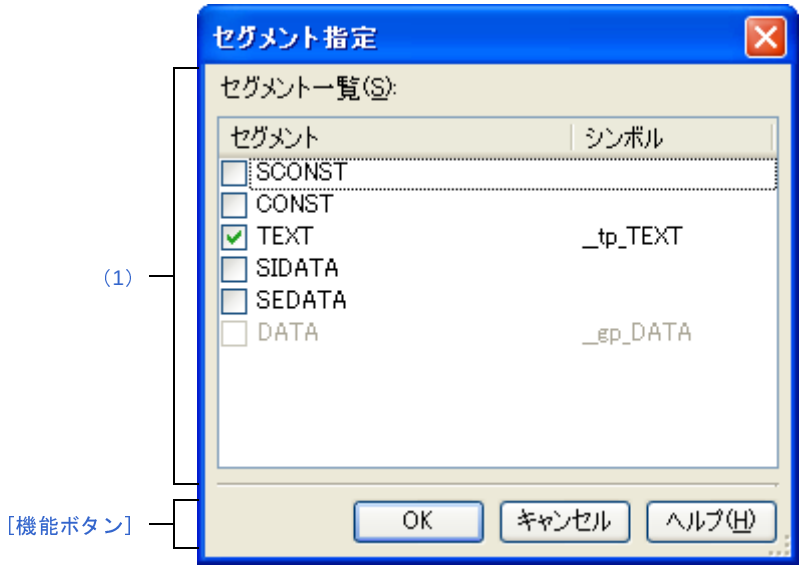
［機能ボタン］

ボタン	機能
OK	本ダイアログをクローズし，選択したファイルをリンク・ディレクティブ生成 ダイアログの［セグメント／セクションの詳細］エリアの［オブジェクト・ファイル名］に設定します。
キャンセル	ファイルの選択をキャンセルし，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

セグメント指定 ダイアログ

本ダイアログの呼び出し元に設定するセグメントを，リンク・ディレクティブ生成 ダイアログで現在設定しているセグメントの中から選択します。

図 A—34 セグメント指定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- リンク・ディレクティブ生成 ダイアログの［シンボル一覧］エリアでシンボルを選択したのち，［シンボルの詳細］エリアの［セグメント名］において，［...］ボタンをクリック

［各エリアの説明］

(1) ［セグメント一覧］エリア

リンク・ディレクティブ生成 ダイアログで現在設定しているセグメントと，それらを指定しているシンボルの一覧を表示します。

(a) ［セグメント］

リンク・ディレクティブ生成 ダイアログで現在設定しているセグメント名一覧を表示します。

本ダイアログをオープンしたリンク・ディレクティブ生成 ダイアログの［シンボルの詳細］エリアの［セグメント名］に設定するセグメントをチェック・ボックスにより選択します。

備考 1. ファイル名にマウス・カーソルをあわせると，そのファイルの絶対パスをポップアップ表示します。

2. 本ダイアログをオープンしたリンク・ディレクティブ生成 ダイアログの［シンボルの詳細］エリアの［セグメント名］において，すでにセグメントを設定していた場合は，該当するセグメントのチェック・ボックスはデフォルトでチェック状態となります。

3. 本ダイアログをオープンしたシンボル以外のシンボルを指定しているセグメントの場合は，該当するセグメントのチェック・ボックスはチェック不可状態となります。

（b）［シンボル］

表示しているセグメントを指定しているシンボルを表示します。

セグメントを指定しているシンボルが存在しない場合は，空欄となります。

［機能ボタン］

ボタン	機能
OK	本ダイアログをクローズし，選択したセグメントをリンク・ディレクティブ生成 ダイアログの［シンボルの詳細］エリアの［セグメント名］に設定します。
キャンセル	ファイルの選択をキャンセルし，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

リンク順設定 ダイアログ

リンカに入力するオブジェクト・モジュール・ファイル，およびライブラリ・ファイルの参照，およびリンク順の設定を行います。

図 A―35　リンク順設定 ダイアログ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロジェクト・ツリー パネルにおいて，ビルド・ツール・ノードを選択したのち，コンテキスト・メニュー→［リンク順を設定する ...］を選択

［各エリアの説明］

(1) ファイル一覧表示エリア

リンカに入力するファイルの一覧を表示します。

(a)［ファイル］

以下のファイルのファイル名一覧を，リンカへの入力順に表示します。

- 選択しているメイン・プロジェクト，またはサブプロジェクトに追加しているソース・ファイルから生成するオブジェクト・モジュール・ファイル

- 選択しているメイン・プロジェクト，またはサブプロジェクトのプロジェクト・ツリーに直接追加したオブジェクト・モジュール・ファイル
- 選択しているメイン・プロジェクト，またはサブプロジェクトのプロジェクト・ツリーに直接追加したライブラリ・ファイル

デフォルトでは，プロジェクトに追加している順番となります。
ファイルの表示順を変更することにより，リンカへのファイルの入力順を設定することができます。
表示順の変更は，［↑］，および［↓］ボタン，またはファイル名のドラッグ・アンド・ドロップにより行います。

備考 1. ファイル名にマウス・カーソルをあわせると，そのファイルの存在する場所がプロジェクト・ファイルと同一のドライブの場合は相対パスで，異なるドライブの場合は絶対パスでポップアップ表示します。

2. 新規に追加したソース・ファイルから生成するオブジェクト・モジュール・ファイル，および新規に追加したオブジェクト・モジュール・ファイルは，一覧の最後のオブジェクト・モジュール・ファイルの次に追加します。
新規に追加したライブラリ・ファイルは，一覧の最後に追加します。

3. ファイルをドラッグ・アンド・ドロップする際，連続して並んでいるファイル名のみ複数選択することができます。

（b）ボタン

↑	選択しているファイルを上へ移動します。 なお，本ボタンは，ファイルを選択していない場合は無効となります。
↓	選択しているファイルを下へ移動します。 なお，本ボタンは，ファイルを選択していない場合は無効となります。
インポート	インポートするファイルを選択 ダイアログをオープンします。 選択したリンク順指定ファイルからファイル名の記述順を取得し，［ファイル］に反映します。 なお，本ボタンは，［ファイル］に何も表示されていない場合は無効となります。
エクスポート	エクスポートするファイルを選択 ダイアログをオープンします。 ［ファイル］に表示しているファイル名一覧を，指定したリンク順指定ファイルに出力します。 なお，本ボタンは，［ファイル］に何も表示されていない場合は無効となります。

備考 リンク順指定ファイルの利用方法については，「2. 15. 2 ファイルのリンク順を設定する」を参照してください。

［機能ボタン］

ボタン	機能
OK	リンカへのファイルの入力順を［ファイル］の表示順に設定し，本ダイアログをクローズします。
キャンセル	リンク順の設定をキャンセルし，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

ビルド・モード設定 ダイアログ

ビルド・モードの追加と削除，および現在のビルド・モードの一括設定を行います。

図 A—36 ビルド・モード設定 ダイアログ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ［ビルド］メニュー→［ビルド・モードの設定 ...］を選択

［各エリアの説明］

(1) ［変更後のビルド・モード］エリア

［ビルド・モードの一覧］エリアで選択しているビルド・モードを表示します。

(a) ボタン

すべてに適用	表示しているビルド・モードを現在開いているプロジェクトのメイン・プロジェクト，およびすべてのサブプロジェクトに設定します。

(2) ［ビルド・モードの一覧］エリア

現在開いているプロジェクト（メイン・プロジェクト，およびサブプロジェクト）に存在するすべてのビルド・モードを一覧表示します。

デフォルトでは，すべてのプロジェクトの現在のビルド・モードが一致している場合は，そのビルド・モードを選択します。

一致していない場合は，“DefaultBuild”を選択します。

一部のメイン・プロジェクト，およびサブプロジェクトのみに存在するビルド・モードには，“*”を付加します。

なお，ビルド・モードには，あらかじめ“DefaultBuild”が用意されており，常に先頭に表示します。

(a) ボタン

複製 ...	選択しているビルド・モードを複製します。 文字列入力 ダイアログがオープンし，入力した名前でビルド・モードを複製し，現在開いているプロジェクトのメイン・プロジェクト，およびすべてのサブプロジェクトに追加します。 なお，“*”を付加しているビルド・モードを複製する場合，そのビルド・モードがメイン・プロジェクト，およびサブプロジェクトに存在しなければ，DefaultBuild を複製します。 登録可能なビルド・モード数は，20 個までです。
削除	選択しているビルド・モードを削除します。 ただし，DefaultBuild を削除することはできません。
名前の変更 ...	選択しているビルド・モードの名前を変更します。 文字列入力 ダイアログがオープンし，入力した名前でビルド・モードの名前を変更します。

注意 ビルド・モードを複製，およびビルド・モードの名前を変更する場合，すでに存在するビルド・モードと同名の名前を使用することはできません。

備考 1. ビルド・モード名として指定可能な文字数は 127 文字までです。

入力内容が正しくない場合，以下のメッセージをツールチップ表示します。

メッセージ	説明
同名のビルド・モードがすでに存在します。	同名のビルド・モードがすでに存在します。
127 文字を越える文字を指定できません。	長い名前（128 文字以上）のビルド・モードを指定しました。
ビルド・モード名が不正です。文字 (¥ , /, :, *, ?, ", <, >, |) は使用できません。	不正なビルド・モード名を指定しました。 ビルド・モード名のフォルダを作成するため，文字 (¥ , /, :, *, ?, ", <, >, |) は使用できません。

2. 登録可能なビルド・モード数は，20 個までです。

入力内容が正しくない場合，以下のメッセージをツールチップ表示します。

メッセージ	説明
1 つのプロジェクト／サブプロジェクトに設定できるビルド・モード数は，20 個までです。	登録するビルド・モード数が 20 個を越えました。

［機能ボタン］

ボタン	機能
閉じる	本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

バッチ・ビルド ダイアログ

プロジェクト（メイン・プロジェクト，およびサブプロジェクト）が持つビルド・モードを一括して，ビルド，リビルド，クリーンを行います。

備考 バッチ・ビルド順は，プロジェクトのビルド順に従い，サブプロジェクト，メイン・プロジェクトの順となります。
1 つのメイン・プロジェクト，またはサブプロジェクトについて複数のビルド・モードを選択した場合は，そのサブプロジェクトで選択しているすべてのビルド・モードでビルドを行ったのち，次のサブプロジェクト，またはメイン・プロジェクトのビルドを行います。

図 A—37 バッチ・ビルド ダイアログ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ［ビルド］メニュー→［バッチ・ビルド ...］を選択

［各エリアの説明］

(1)［ビルド・モード一覧］エリア

現在開いているプロジェクトが持つメイン・プロジェクト，およびサブプロジェクトの名前と，それらが持つビルド・モード，定義マクロの組み合わせの一覧を表示します。

(a)［プロジェクト］

現在開いているプロジェクトが持つメイン・プロジェクト，およびサブプロジェクトを表示します。

ビルドを行うメイン・プロジェクト，およびサブプロジェクトとビルド・モードの組み合わせをチェック・ボックスにより選択します。

プロジェクトを作成後，最初に本ダイアログをオープンした場合は，すべてのチェック・ボックスをチェックしません。2 回目以降は前回のチェック状態を保持します。

(b)［ビルド・モード］

メイン・プロジェクト，およびサブプロジェクトが持つビルド・モードを表示します。

(c)［定義マクロ］

メイン・プロジェクト，およびサブプロジェクトとそのビルド・モードの組み合わせに対して，プロパティ パネルの［コンパイル・オプション］タブ，および［アセンブル・オプション］タブで設定している定義マクロを“｜”で区切って表示します。

なお，コンパイル・オプションの定義マクロ，アセンブル・オプションの定義マクロの順で表示し，コンパイル・オプションの定義マクロとアセンブル・オプションの定義マクロの間は“,”で区切って表示します。

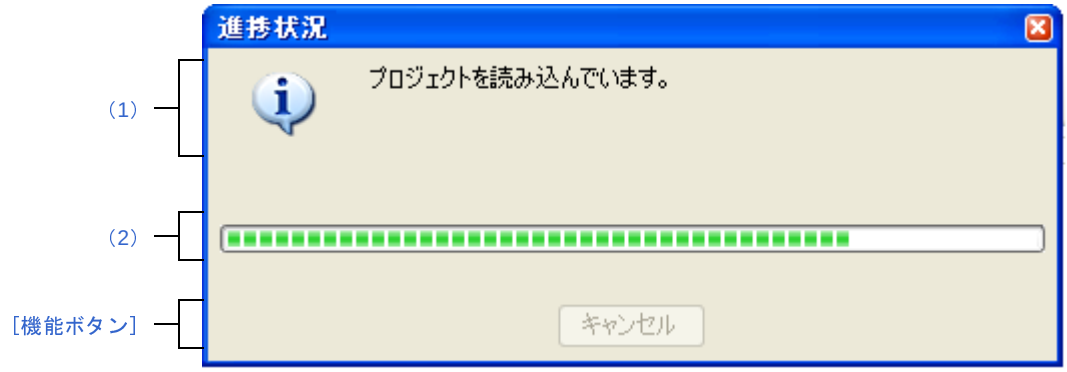
［機能ボタン］

ボタン	機能
ビルド	本ダイアログをクローズし，選択しているプロジェクトをそのビルド・モードでビルドします。 ビルドの実行結果は，出力 パネルに表示します。 ビルド完了後，ビルド・モードは本ダイアログをオープンする前の設定に戻ります。 なお，本ボタンは，プロジェクトを選択していない場合は無効となります。
リビルド	本ダイアログをクローズし，選択しているプロジェクトをそのビルド・モードでリビルドします。 リビルドの実行結果は，出力 パネルに表示します。 リビルド完了後，ビルド・モードは本ダイアログをオープンする前の設定に戻ります。 なお，本ボタンは，プロジェクトを選択していない場合は無効となります。
クリーン	本ダイアログをクローズし，選択しているプロジェクトのそのビルド・モードでビルドしたファイルを削除します。 クリーンの実行結果は，出力 パネルに表示します。 クリーン完了後，ビルド・モードは本ダイアログをオープンする前の設定に戻ります。 なお，本ボタンは，プロジェクトを選択していない場合は無効となります。
閉じる	本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

処理中表示 ダイアログ

時間を要する処理を行っている際に，その進捗状況の表示を行います。
本ダイアログは，実行中の処理が完了した場合，自動的にクローズします。

図 A―38　処理中表示 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- 時間を要する処理において，メッセージが発生した際に自動的に表示

［各エリアの説明］

(1) メッセージ表示エリア

処理中に発生したメッセージを表示します（編集不可）。

(2) プログレスバー

現在実行中の処理の進捗状況をバーの長さで表示します。
なお，進捗率が 100% に達した場合（右端までバーの長さが達した場合），本ダイアログは自動的にクローズします。

［機能ボタン］

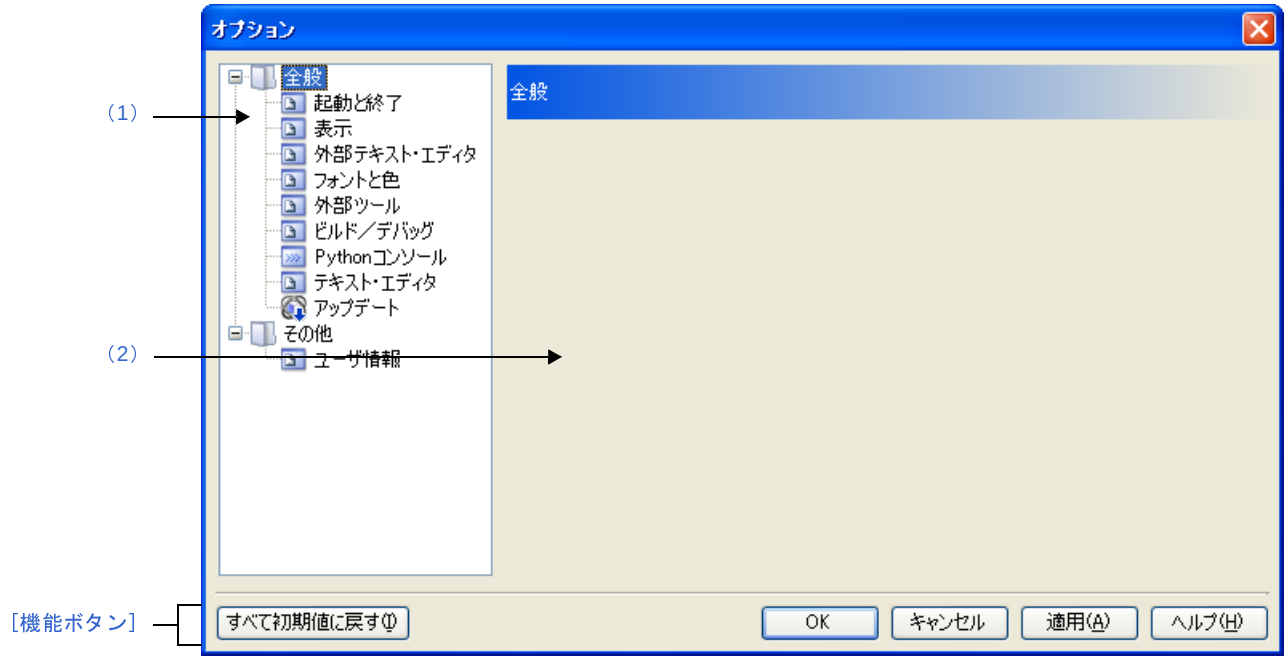
ボタン	機能
キャンセル	現在実行中の処理を中断し，本ダイアログをクローズします。 ただし，実行中の処理の中断が不可能な場合，本ボタンは無効となります。

オプション ダイアログ

CubeSuite+ の各種環境設定を行います。

本ダイアログでの設定は，使用中のユーザの設定として保存します。

図 A―39　オプション ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ［ツール］メニュー→［オプション ...］を選択

[各エリアの説明]

(1) カテゴリ選択エリア

設定したい項目を次のカテゴリから選択します。

カテゴリ	設定内容
[全般 - 起動と終了] カテゴリ	起動，または終了時に関連した設定を行います。
[全般 - 表示] カテゴリ	表示に関連した設定を行います。
[全般 - 外部テキスト・エディタ] カテゴリ	外部テキスト・エディタに関連した設定を行います。
[全般 - フォントと色] カテゴリ	各パネルで表示するフォントと色に関連した設定を行います。
[全般 - 外部ツール] カテゴリ	外部ツールを起動する際の設定を行います。
[全般 - ビルド／デバッグ] カテゴリ	ビルド，またはデバッグに関連した設定を行います。
[全般 - Python コンソール] カテゴリ	Python コンソールに関連した設定を行います。
[全般 - テキスト・エディタ] カテゴリ	テキスト・エディタに関連した設定を行います。
[全般 - アップデート] カテゴリ	アップデートに関連した設定を行います。
[その他 - ユーザ情報] カテゴリ	ユーザ情報に関連した設定を行います。

備考　[全般 - ビルド／デバッグ] 以外のカテゴリについては，「CubeSuite+ 統合開発環境 ユーザーズマニュアル 起動編」を参照してください。

(2) 設定エリア

選択したカテゴリに対して，各種オプションを設定するエリアです。

各カテゴリの設定方法についての詳細は，該当するカテゴリの項を参照してください。

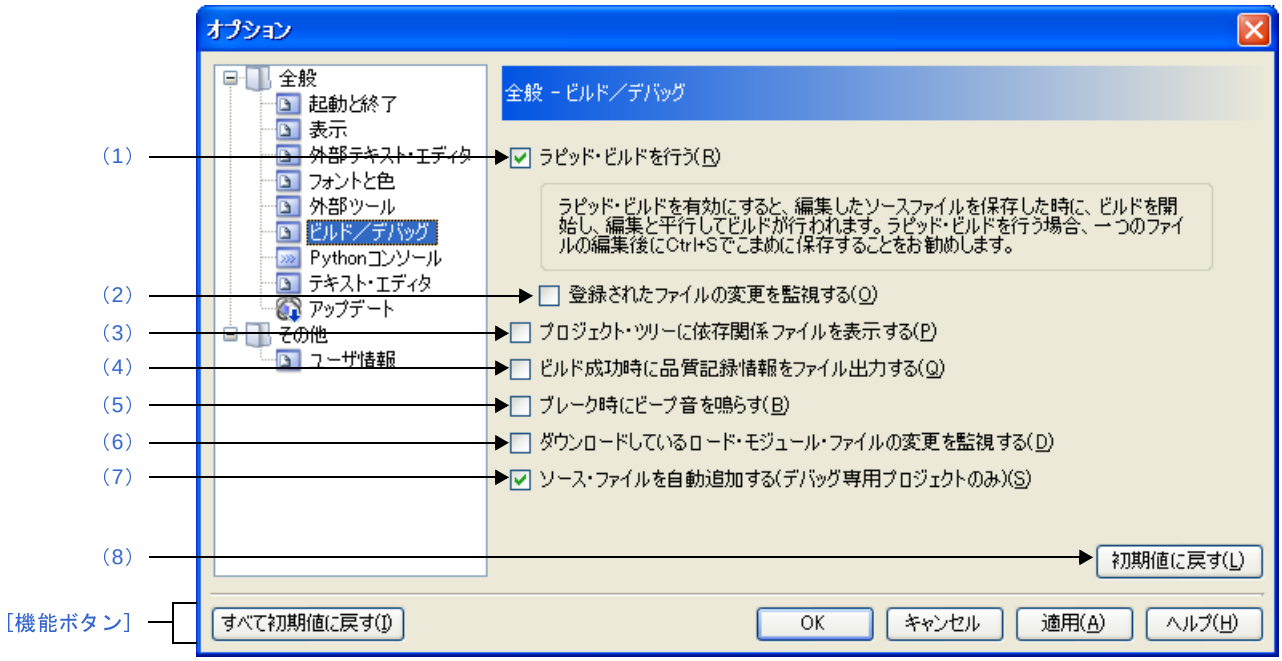
[機能ボタン]

ボタン	機能
すべて初期値に戻す	本ダイアログのすべての設定項目をデフォルトの状態に戻します。 ただし，[全般 - 外部ツール] カテゴリでは，新規登録した内容の削除は行いません。
OK	変更した設定内容を適用し，本ダイアログをクローズします。
キャンセル	変更した設定内容を無効とし，本ダイアログをクローズします。
適用	変更した設定内容を適用します（本ダイアログをクローズしません）。
ヘルプ	本ダイアログのヘルプを表示します。

［全般 - ビルド／デバッグ］カテゴリ

全般に関わる設定のうち，ビルド，またはデバッグに関連した設定を行います。

図 A―40　オプション ダイアログ（［全般 - ビルド／デバッグ］カテゴリ）



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ［ツール］メニュー→［オプション ...］を選択

［各エリアの説明］

(1) ［ラピッド・ビルドを行う］

☑	ラピッド・ビルド機能[注]を有効にします（デフォルト）。
☐	ラピッド・ビルド機能を使用しません。

注　編集したソース・ファイルの保存時に，ビルドを自動で開始する機能です。
本機能を有効にすることにより，ソース・ファイルの編集と同時にビルドを行うことができます。
なお，本機能を使用する場合，ソース・ファイル編集後，こまめに上書き保存することを推奨します。

(2)［登録されたファイルの変更を監視する］

☑	プロジェクトに登録したソース・ファイルの変更を監視し，外部テキスト・エディタなどで編集／保存したときに，ラピッド・ビルドを開始します。
☐	プロジェクトに登録したソース・ファイルの変更を監視し，外部テキスト・エディタなどで編集／保存したときに，ラピッド・ビルドを開始しません（デフォルト）。

備考 ［ラピッド・ビルドを行う］チェック・ボックスにチェックが付いている場合のみ有効です。

注意 1. 本項目をチェックし，かつ，ラピッド・ビルドの対象となったファイルをビルド前に実行するコマンド，ビルド後に実行するコマンドなどで自動で編集／上書きするように登録した場合，ラピッド・ビルドが終了しなくなります。
ラピッド・ビルドが終了しなくなった場合は，本項目のチェックを外して，ラピッド・ビルドを停止してください。

2. 本項目をチェックし，かつ，プロジェクトに登録されたソース・ファイルで存在しないファイル（プロジェクト・ツリーでグレー表示されたファイル）がある場合，エクスプローラなどでファイルを再登録しても，監視状態にはなりません。
監視状態にするためには，プロジェクト・ファイルを読み込み直すか，または本項目のチェックを一旦外してダイアログを閉じた後，再度本項目をチェックしてください。

(3)［プロジェクト・ツリーに依存関係ファイルを表示する］

☑	ソース・ファイルが依存しているファイル群をプロジェクト・ツリーに表示します。
☐	ソース・ファイルが依存しているファイル群をプロジェクト・ツリーに表示しません（デフォルト）。

(4)［ビルド成功時に品質記録情報をファイル出力する］

☑	ビルド成功時に品質記録情報ファイルを出力します。
☐	ビルド成功時に品質記録情報ファイルを出力しません（デフォルト）。

備考 1. 品質記録情報ファイルは，ラピッド・ビルドを行う場合，デバッグ専用プロジェクトをビルドする場合，ファイル単位でコンパイル／アセンブルする場合は出力しません。

2. 品質記録情報ファイルには，以下の情報を出力します。
- ファイルの作成日時
- ビルド結果のログ
- ビルド中に使用したコマンド・ファイルの情報
- 本製品の詳細バージョンや現在のプロジェクトの情報

3. 品質記録情報ファイルは，各プロジェクトのプロジェクト・フォルダに“品質記録情報 (*プロジェクト名*, *ビルド・モード名*).txt”というファイル名で出力します。
同名のファイルが存在する場合は上書きします。
また，プロジェクト・ツリーのビルド・ツール生成ファイル・ノードにも表示します。

(5)［ブレーク時にビープ音を鳴らす］

☑	プログラムの実行が，ブレーク・イベント（ハードウエア・ブレーク／ソフトウエア・ブレーク）により停止した際，ビープ音を鳴らします。
☐	プログラムの実行が，ブレーク・イベント（ハードウエア・ブレーク／ソフトウエア・ブレーク）により停止した際，ビープ音を鳴らしません（デフォルト）。

(6)［ダウンロードしているロード・モジュール・ファイルの変更を監視する］

☑	デバッグ・ツールにダウンロードしているロード・モジュール・ファイルの変更を監視し，変更があった場合は，ダウンロードの実行を確認するメッセージ ダイアログを表示します。
☐	デバッグ・ツールにダウンロードしているロード・モジュール・ファイルの変更の監視を行いません（デフォルト）。

(7)［ソース・ファイルを自動追加する (デバッグ専用プロジェクトのみ)］

☑	デバッグ専用プロジェクトにおいて，デバッグ・ツールにロード・モジュール・ファイルをダウンロードする際，プロジェクト・ツリーにソース・ファイルを自動追加します（デフォルト）。
☐	デバッグ専用プロジェクトにおいて，デバッグ・ツールにロード・モジュール・ファイルをダウンロードする際，プロジェクト・ツリーへのソース・ファイルの自動追加を行いません。

注意　本機能は，プロジェクト・ツリーのダウンロード・ファイル・ノードにロード・モジュール・ファイルを追加した場合のみ有効となります。
デバッグ・ツールのプロパティ パネルの［ダウンロード・ファイル設定］タブにてロード・モジュール・ファイルを追加した場合は，プロジェクト・ツリーにソース・ファイルは追加されません。

(8) ボタン・エリア

初期値に戻す	現在表示している項目をすべてデフォルトに戻します。

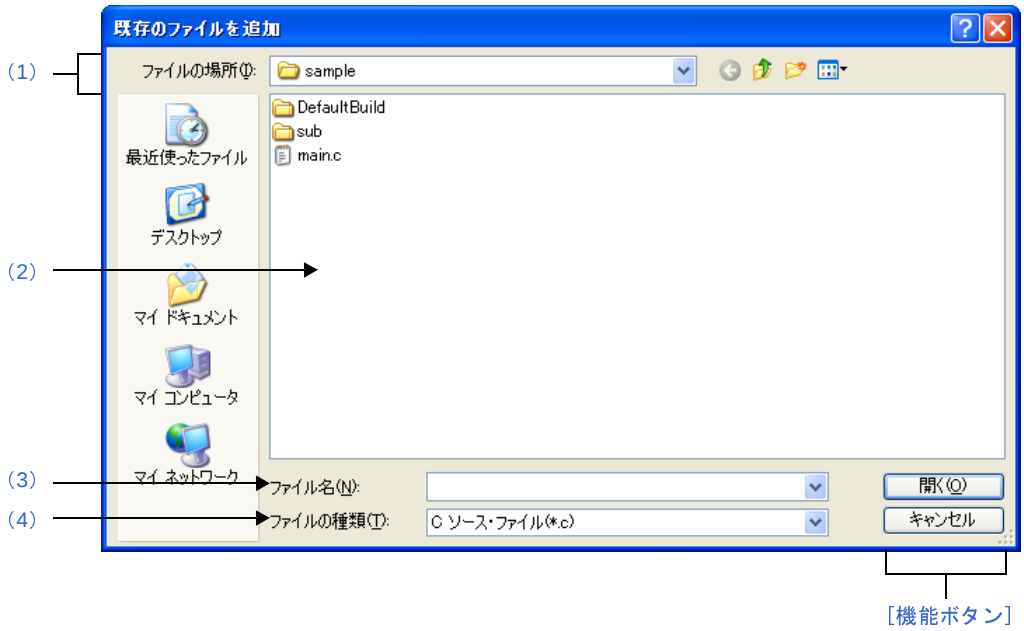
［機能ボタン］

ボタン	機能
すべて初期値に戻す	本ダイアログのすべての設定項目をデフォルトの状態に戻します。 ただし，［全般 - 外部ツール］カテゴリでは，新規登録した内容の削除は行いません。
OK	変更した設定内容を適用し，本ダイアログをクローズします。
キャンセル	変更した設定内容を無効とし，本ダイアログをクローズします。
適用	変更した設定内容を適用します（本ダイアログをクローズしません）。
ヘルプ	本ダイアログのヘルプを表示します。

既存のファイルを追加 ダイアログ

プロジェクトに追加する既存のファイルの選択を行います。

図 A―41　既存のファイルを追加 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ［ファイル］メニュー→［追加］→［既存のファイルを追加 ...］を選択
- プロジェクト・ツリー パネルにおいて，プロジェクト・ノード，サブプロジェクト・ノード，ファイル・ノード，ファイルのいずれかを選択したのち，コンテキスト・メニュー→［追加］→［既存のファイルを追加 ...］を選択

［各エリアの説明］

(1)［ファイルの場所］エリア

プロジェクトに追加するファイルが存在するフォルダを選択します。

デフォルトでは，プロジェクト・フォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

プロジェクトに追加するファイルのファイル名を指定します。

(4)［ファイルの種類］エリア

プロジェクトに追加するファイルのファイルの種類（ファイル・タイプ）を選択します。

C ソース・ファイル (*.c)	C ソース・ファイル
ヘッダ・ファイル (*.h; *.inc)	ヘッダ・ファイル
アセンブル・ファイル (*.asm; *.s)	アセンブラ・ソース・ファイル
リンク・ディレクティブ・ファイル (*.dir; *.dr)	リンク・ディレクティブ・ファイル
シンボル情報ファイル (*.sfg)	シンボル情報ファイル
ライブラリ・ファイル (*.lib; *.a)	ライブラリ・ファイル
オブジェクト・モジュール・ファイル (*.obj; *.o)	オブジェクト・モジュール・ファイル
テキスト・ファイル (*.txt)	テキスト形式
すべてのファイル (*.*)	すべての形式（デフォルト）

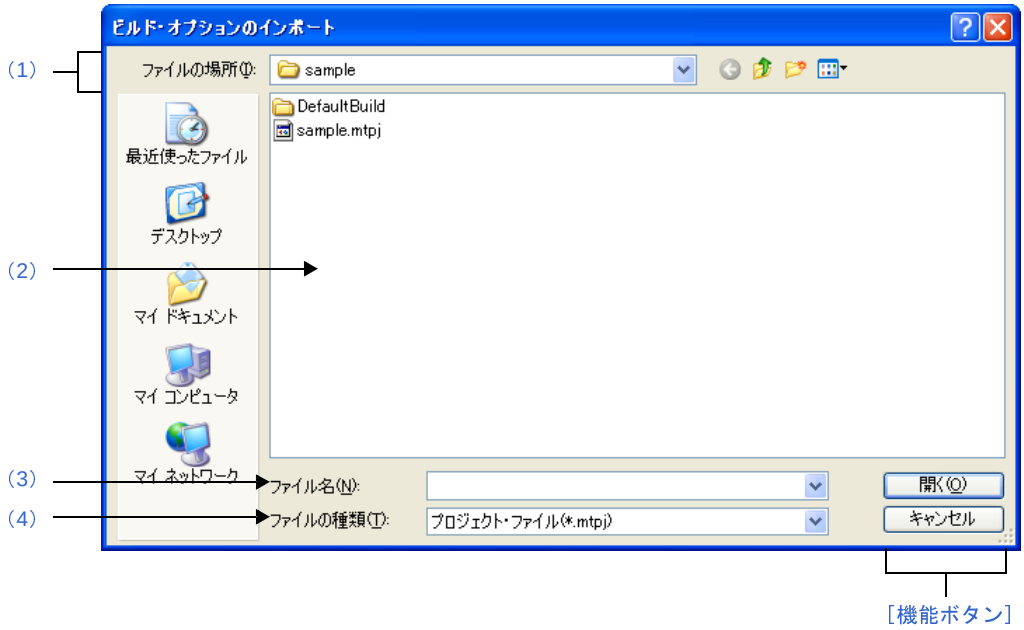
［機能ボタン］

ボタン	機能
開く	指定したファイルをプロジェクトに追加します。
キャンセル	本ダイアログをクローズします。

ビルド・オプションのインポート ダイアログ

ビルド・オプションのインポート対象となるプロジェクト・ファイルの選択を行います。

図 A—42　ビルド・オプションのインポート ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロジェクト・ツリー パネルにおいて，ビルド・ツール・ノードを選択したのち，コンテキスト・メニュー→［ビルド・オプションのインポート ...］を選択

［各エリアの説明］

(1)［ファイルの場所］エリア

ビルド・オプションのインポート対象となるプロジェクト・ファイルが存在するフォルダを選択します。

デフォルトでは，現在のプロジェクト・フォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

プロジェクト・ファイルのファイル名を指定します。

(4)［ファイルの種類］エリア

プロジェクト・ファイルの種類（ファイル・タイプ）を選択します。

プロジェクト・ファイル (*.mtpj)	プロジェクト・ファイル
サブプロジェクト・ファイル (*.mtsp)	サブプロジェクト・ファイル

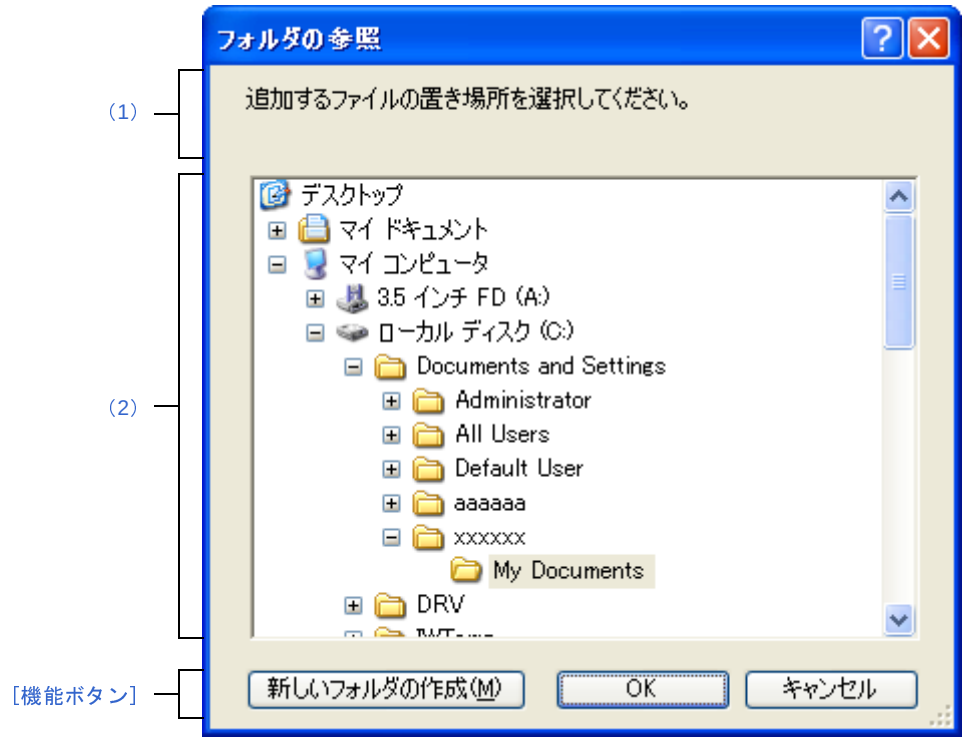
［機能ボタン］

ボタン	機能
開く	指定したプロジェクト・ファイルのビルド・オプションを現在のプロジェクトにインポートします。
キャンセル	本ダイアログをクローズします。

フォルダの参照 ダイアログ

本ダイアログの呼び出し元に設定するフォルダの選択を行います。

図 A―43　フォルダの参照 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- ファイル追加 ダイアログにおいて，［作成場所］エリア内の［参照 ...］ボタンをクリック
- パス編集 ダイアログにおいて，パス編集エリア内の［参照 ...］ボタンをクリック
- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［共通オプション］タブの［出力ファイルの種類と場所］カテゴリの［中間ファイル出力フォルダ］，［よく使うオプション（リンク）］カテゴリの［出力フォルダ］，［エラー出力］カテゴリの［エラー・メッセージ・ファイル出力フォルダ］，［その他］カテゴリの［一時作業フォルダ］
  - ［コンパイル・オプション］タブの［出力ファイル］カテゴリの［アセンブラ・ソース・ファイル出力フォルダ］，［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイル出力フォルダ］
  - ［リンク・オプション］タブの［出力ファイル］カテゴリの［出力フォルダ］，［リンク・マップ］カテゴリの［リンク・マップ・ファイル出力フォルダ］，［シンボル情報］カテゴリの［シンボル情報ファイル出力フォルダ］

- ［ヘキサ出力オプション］タブの［出力ファイル］カテゴリの［ヘキサ・ファイル出力フォルダ］
- ［ライブラリ生成オプション］タブの［出力ファイル］カテゴリの［出力フォルダ］
- ［個別コンパイル・オプション］タブの［出力ファイル］カテゴリの［アセンブラ・ソース・ファイル出力フォルダ］，［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイル出力フォルダ］，［エラー出力］カテゴリの［エラー・メッセージ・ファイル出力フォルダ］
- ［個別アセンブル・オプション］タブの［エラー出力］カテゴリの［エラー・メッセージ・ファイル出力フォルダ］

［各エリアの説明］

（1）メッセージ・エリア

本ダイアログで選択するフォルダに関するメッセージを表示します。

（2）フォルダの場所エリア

本ダイアログの呼び出し元に設定するフォルダを選択します。

なお，デフォルトで選択されるフォルダは，呼び出し元によって異なります。

（a）ファイル追加 ダイアログの場合

呼び出し元に設定しているフォルダが選択されます。

呼び出し元が空欄，または存在しないパスを設定している場合は，プロジェクト・フォルダが選択されます。

（b）パス編集 ダイアログ，およびプロパティ パネルの場合

プロジェクト・フォルダが選択されます。

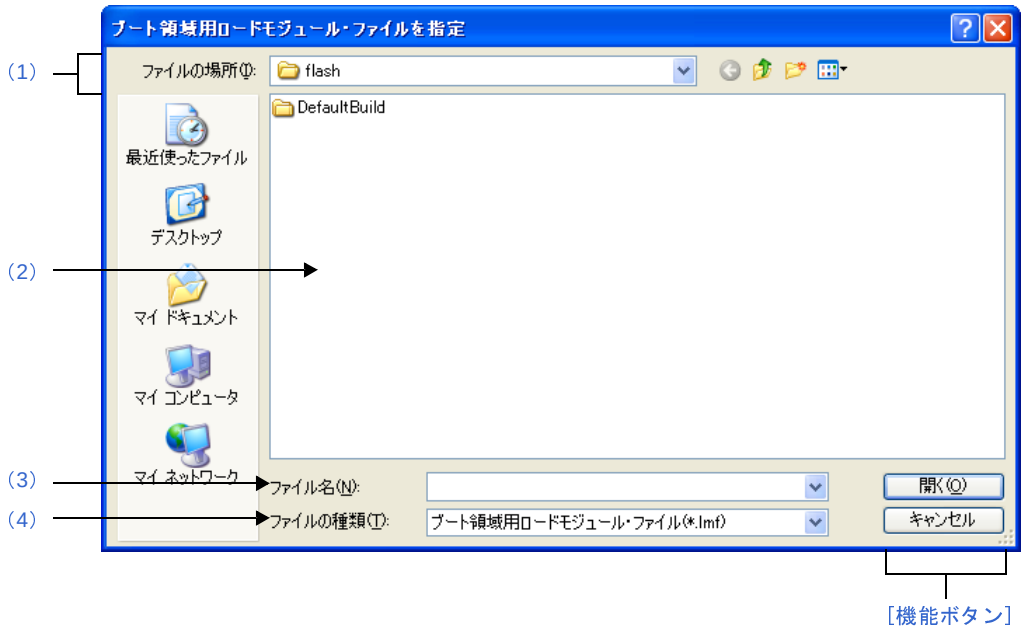
［機能ボタン］

ボタン	機能
新しいフォルダの作成	選択したフォルダの直下に新しいフォルダを作成します。 フォルダ名は，デフォルトで“新しいフォルダ”となります。
OK	指定したフォルダのパスを本ダイアログの呼び出し元に設定します。
キャンセル	本ダイアログをクローズします。

ブート領域用ロード・モジュール・ファイルを指定 ダイアログ

本ダイアログの呼び出し元に設定するブート領域用ロード・モジュール・ファイルの選択を行います。

図 A―44 ブート領域用ロード・モジュール・ファイルを指定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［共通オプション］タブの［フラッシュ対応］カテゴリの［ブート領域用ロード・モジュール・ファイル名］

［各エリアの説明］

(1)［ファイルの場所］エリア

本ダイアログの呼び出し元に設定するファイルが存在するフォルダを選択します。

デフォルトでは，プロジェクト・フォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

本ダイアログの呼び出し元に設定するファイルの名前を指定します。

(4)［ファイルの種類］エリア

本ダイアログの呼び出し元に設定するファイルの種類（ファイル・タイプ）を選択します。

ブート領域用ロード・モジュール・ファイル (*.lmf)	ブート領域用ロード・モジュール・ファイル（デフォルト）
すべてのファイル (*.*)	すべての形式

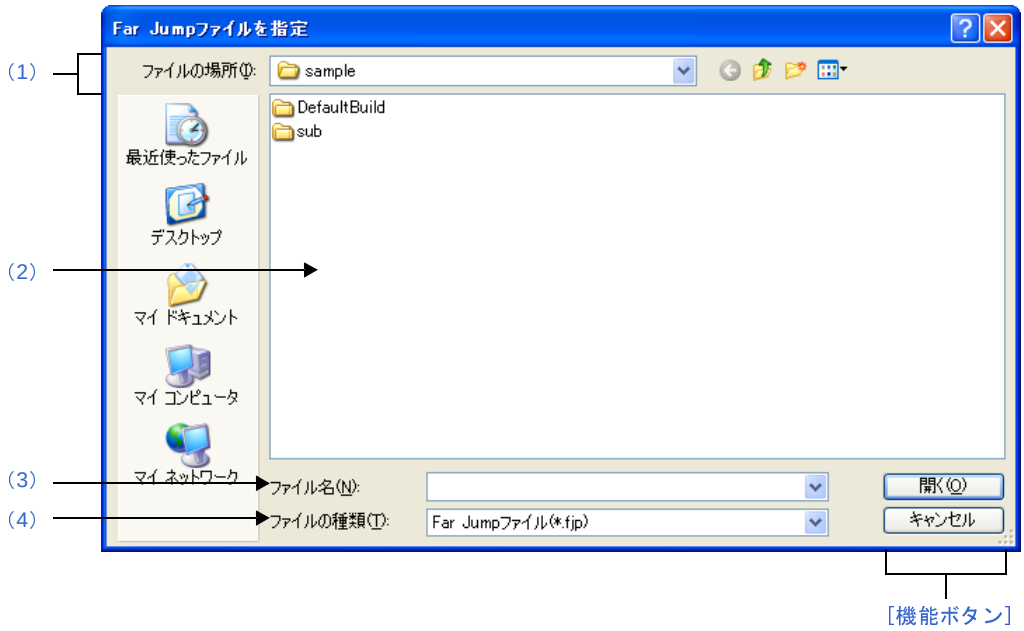
［機能ボタン］

ボタン	機能
開く	本ダイアログの呼び出し元に指定したファイルを設定します。
キャンセル	本ダイアログをクローズします。

Far Jump ファイルを指定 ダイアログ

本ダイアログの呼び出し元に設定する Far Jump ファイルの選択を行います。

図 A―45 Far Jump ファイルを指定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，［コンパイル・オプション］タブの［出力コード］カテゴリの［Far Jump ファイル名］プロパティを選択したのち，［...］ボタンをクリックしてパス編集 ダイアログをオープン
  →ダイアログ上で［参照 ...］ボタンをクリック

［各エリアの説明］

(1)［ファイルの場所］エリア

本ダイアログの呼び出し元に設定するファイルが存在するフォルダを選択します。

デフォルトでは，プロジェクト・フォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

本ダイアログの呼び出し元に設定するファイルの名前を指定します。

(4)［ファイルの種類］エリア

本ダイアログの呼び出し元に設定するファイルの種類（ファイル・タイプ）を選択します。

Far Jump ファイル (*.fjp)	Far Jump ファイル

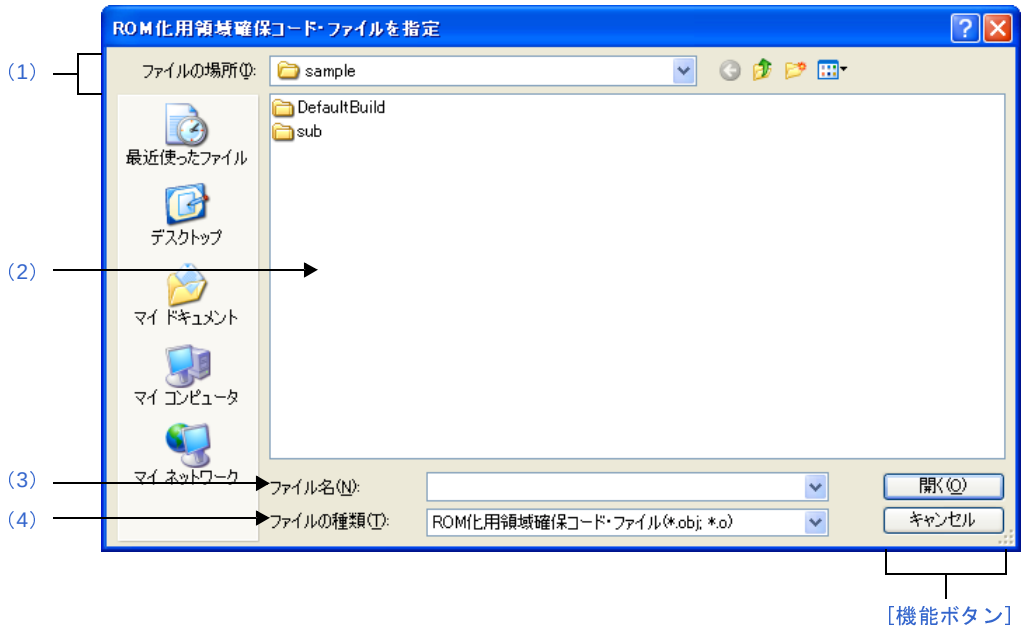
［機能ボタン］

ボタン	機能
開く	本ダイアログの呼び出し元に指定したファイルを設定します。
キャンセル	本ダイアログをクローズします。

ROM 化用領域確保コード・ファイルを指定 ダイアログ

本ダイアログの呼び出し元に設定する ROM 化用領域確保コード・ファイルの選択を行います。

図 A―46　ROM 化用領域確保コード・ファイルを指定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルにおいて，以下のプロパティを選択したのち，［...］ボタンをクリック
  - ［ROM 化オプション］タブの［入力ファイル］カテゴリの［ROM 化用領域確保コード・ファイル名］

［各エリアの説明］

(1)［ファイルの場所］エリア

本ダイアログの呼び出し元に設定するファイルが存在するフォルダを選択します。

デフォルトでは，プロジェクト・フォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

本ダイアログの呼び出し元に設定するファイルの名前を指定します。

(4)［ファイルの種類］エリア

本ダイアログの呼び出し元に設定するファイルの種類（ファイル・タイプ）を選択します。

ROM 化用領域確保コード・ファイル (*.obj; *.o)	ROM 化用領域確保コード・ファイル（デフォルト）
ROM 化用領域確保コード・ファイル (*.asm; *.s)	ROM 化用領域確保コード・ファイル

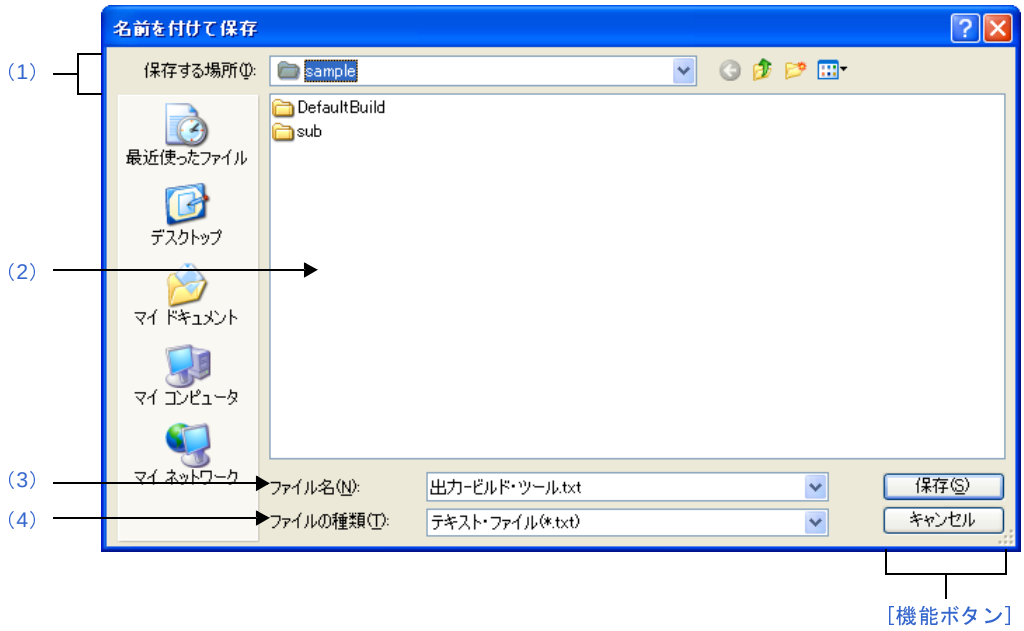
［機能ボタン］

ボタン	機能
開く	本ダイアログの呼び出し元に指定したファイルを設定します。
キャンセル	本ダイアログをクローズします。

名前を付けて保存 ダイアログ

編集中のファイル，または各パネルの内容に名前を付けてファイルに保存します。

図 A―47　名前を付けて保存 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- エディタ パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けて *ファイル名*を保存 ...］を選択
- 出力 パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けて *タブ名*を保存 ...］を選択

［各エリアの説明］

(1)［保存する場所］エリア

パネルに表示している内容をファイルに保存するためのフォルダを選択します。

デフォルトでは，以下のフォルダを選択します。

(a) エディタ パネルの場合

現在編集しているファイルが存在しているフォルダ

(b) 出力 パネルの場合

初めて保存する場合はプロジェクト・フォルダ，2 回目以降は前回選択したフォルダ

(2) ファイルの一覧エリア

［保存する場所］エリア，および［ファイルの種類］エリアで選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

保存する際のファイル名を指定します。

(4)［ファイルの種類］エリア

(a) エディタ パネルの場合

編集中のファイルの種類に依存して，次のファイルの種類（ファイル・タイプ）を表示します。

テキスト・ファイル (*.txt)	テキスト形式
C ソース・ファイル (*.c)	C ソース・ファイル
ヘッダ・ファイル (*.h; *.inc)	ヘッダ・ファイル
アセンブル・ファイル (*.asm)	アセンブラ・ソース・ファイル
アセンブル・ファイル (*.s)	アセンブラ・ソース・ファイル
リンク・ディレクティブ・ファイル (*.dir; *.dr)	リンク・ディレクティブ・ファイル
リンク順指定ファイル (*.mtls)	リンク順指定ファイル
シンボル情報ファイル (*.sfg)	シンボル情報ファイル
マップ・ファイル (*.map)	マップ・ファイル
ヘキサ・ファイル (*.hex)	ヘキサ・ファイル

(b) 出力 パネルの場合

次のファイルの種類（ファイル・タイプ）を表示します。

テキスト・ファイル (*.txt)	テキスト形式

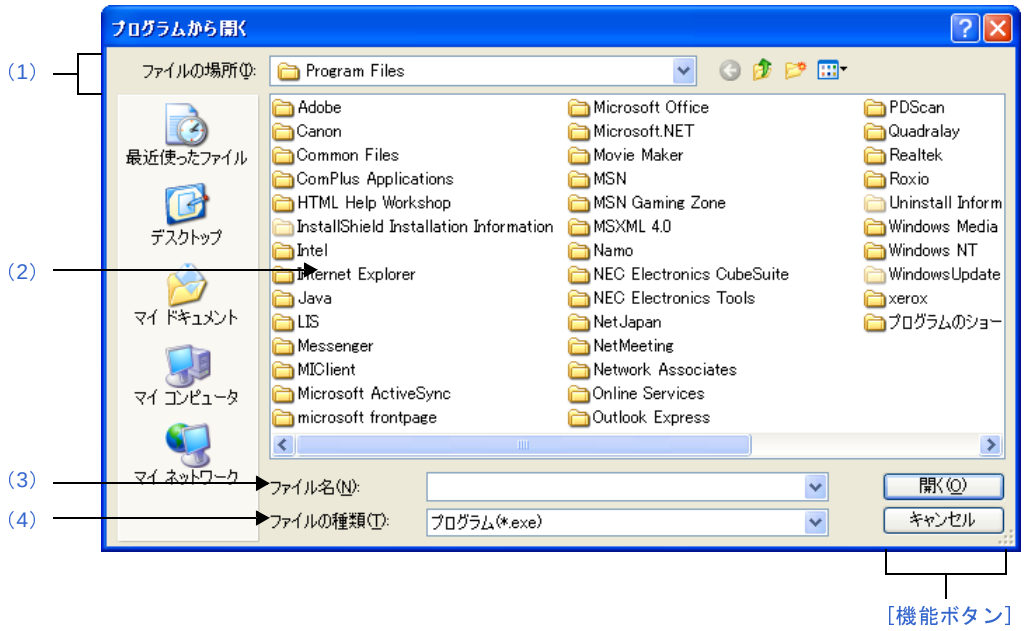
［機能ボタン］

ボタン	機能
保存	指定したファイル名でファイルを保存します。
キャンセル	本ダイアログをクローズします。

プログラムから開く ダイアログ

プロジェクト・ツリー上で選択しているファイルを開くアプリケーションの選択を行います。

図 A―48　プログラムから開く ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロジェクト・ツリー パネルにおいて，ファイルを選択したのち，コンテキスト・メニュー→［アプリケーションを指定して開く ...］を選択

［各エリアの説明］

(1) ［ファイルの場所］エリア

ファイルを開くアプリケーションが存在するフォルダを選択します。

デフォルトでは，プログラム・フォルダ（Windows XP の場合は“C:￥Program Files”）を選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

ファイルを開くアプリケーションの実行ファイル名を指定します。

(4)［ファイルの種類］エリア

ファイルを開くアプリケーションの実行ファイルの種類（ファイル・タイプ）を選択します。

プログラム (*.exe)	実行形式（デフォルト）
すべてのファイル (*.*)	すべての形式

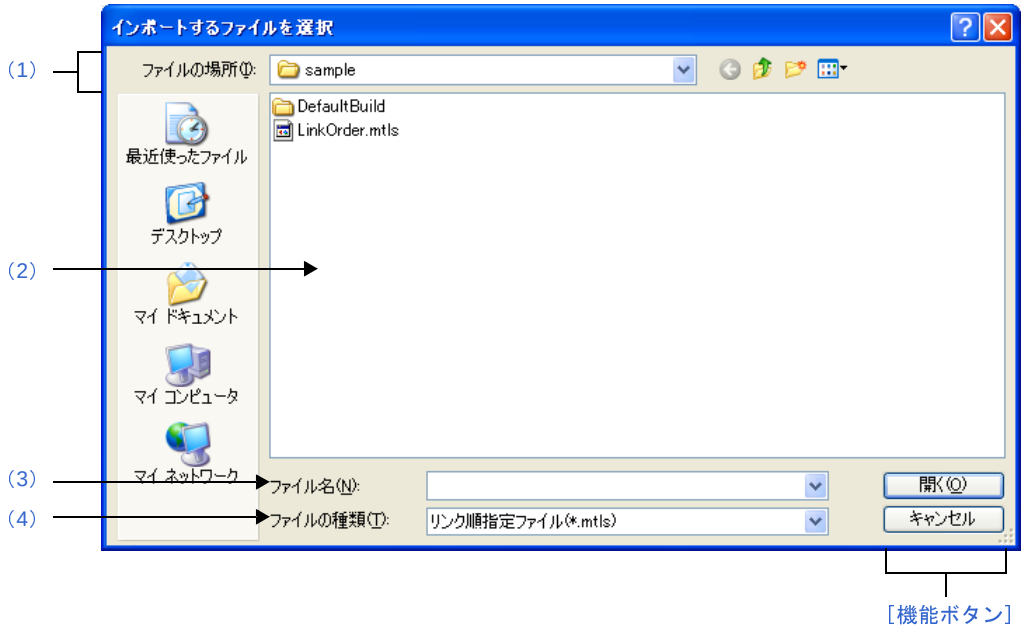
［機能ボタン］

ボタン	機能
開く	指定したアプリケーションでファイルをオープンします。
キャンセル	本ダイアログをクローズします。

インポートするファイルを選択 ダイアログ

リンク順設定 ダイアログにインポートするリンク順指定ファイルの選択を行います。

図 A—49　インポートするファイルを選択 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- リンク順設定 ダイアログにおいて，［インポート］ボタンをクリック

［各エリアの説明］

(1)［ファイルの場所］エリア

リンク順指定ファイルが存在するフォルダを選択します。

デフォルトでは，初めて選択する場合はプロジェクト・フォルダ，2 回目以降は前回選択したフォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

リンク順指定ファイルのファイル名を指定します。

(4)［ファイルの種類］エリア

リンク順指定ファイルの種類（ファイル・タイプ）を選択します。

リンク順指定ファイル (*.mtls)	リンク順指定ファイル

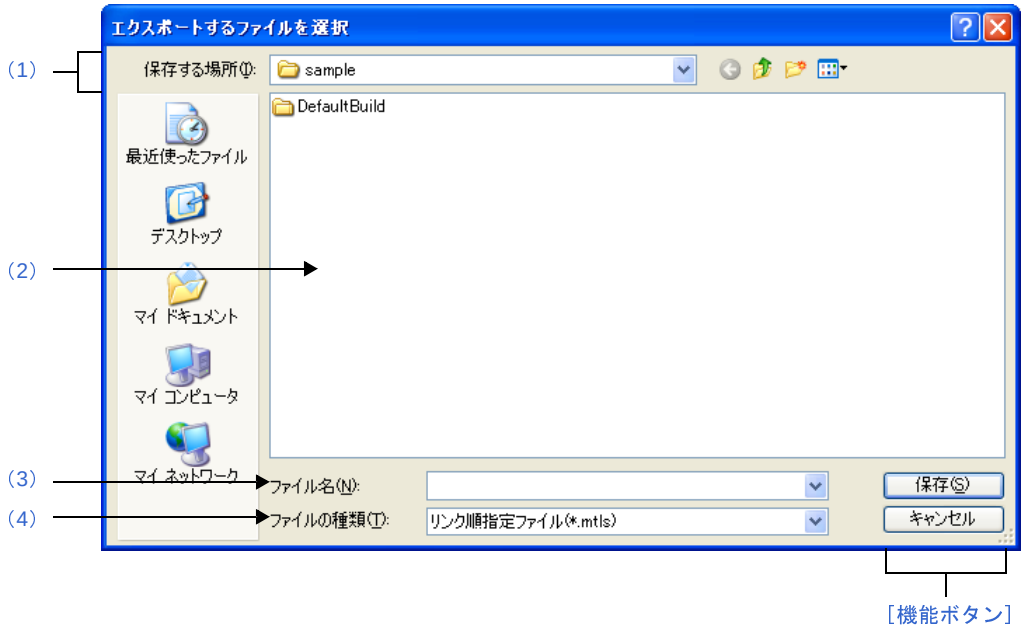
［機能ボタン］

ボタン	機能
開く	指定したファイルをリンク順設定 ダイアログにインポートします。
キャンセル	本ダイアログをクローズします。

エクスポートするファイルを選択 ダイアログ

リンク順指定ファイルの生成を行います。

図 A—50　エクスポートするファイルを選択 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- リンク順設定 ダイアログにおいて，［エクスポート］ボタンをクリック

［各エリアの説明］

(1) ［保存する場所］エリア

リンク順指定ファイルを出力するフォルダを選択します。

デフォルトでは，初めて選択する場合はプロジェクト・フォルダ，2 回目以降は前回選択したフォルダを選択します。

(2) ファイルの一覧エリア

［保存する場所］エリア，および［ファイルの種類］で選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

リンク順指定ファイルのファイル名を指定します。

(4)［ファイルの種類］エリア

次のファイルの種類（ファイル・タイプ）を表示します。

リンク順指定ファイル (*.mtls)	リンク順指定ファイル

［機能ボタン］

ボタン	機能
保存	指定したファイル名でリンク順指定ファイルを生成します。
キャンセル	本ダイアログをクローズします。

Stack Usage Tracer ウインドウ

スタック見積もりツールを起動した際，最初にオープンするウインドウです。

スタックの使用量を関数単位に確認／変更する際は，本ウインドウから行います。

図 A—51　Stack Usage Tracer ウインドウ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［注意］

［オープン方法］

- ［ツール］メニュー→［スタック見積もりツールの起動］を選択

[各エリアの説明]

(1) メニューバー

本エリアは，以下に示したメニュー群から構成されています。

(a) [ファイル] メニュー

選択した関数の最大経路の保存 ...	ツリー表示エリア／リスト表示エリアで選択した関数のスタック・サイズ（呼び出し関数のスタック・サイズを含む）が最大となる呼び出し経路を出力結果ファイルとして保存するための名前を付けて保存 ダイアログをオープンします。  ボタンのクリックと同様です。
選択した関数の全経路保存 ...	ツリー表示エリア／リスト表示エリアで選択した関数の全呼び出し経路を出力結果ファイルとして保存するための名前を付けて保存 ダイアログをオープンします。
全ルート関数の最大経路の保存 ...	ツリー表示エリアに表示している関数の中でスタック・サイズが最大となる呼び出し経路を出力結果ファイルとして保存するための名前を付けて保存 ダイアログをオープンします。
全ルート関数の全経路保存 ...	ツリー表示エリアに表示している全関数の全呼び出し経路を出力結果ファイルとして保存するための名前を付けて保存 ダイアログをオープンします。
スタックサイズ指定ファイルを開く ...	スタック・サイズ指定ファイルを読み込むためのファイルを開く ダイアログをオープンします。
スタックサイズ指定ファイルの保存 ...	スタックサイズ変更 ダイアログで行った各種操作結果（関数に対する情報の変更など）をスタック・サイズ指定ファイルとして保存するための名前を付けて保存 ダイアログをオープンします。
skcx の終了	本ウインドウをクローズします。

備考 出力結果ファイルの保存は，テキスト形式（*.txt），またはCSV形式（*.csv）に限られます。

(b) [表示] メニュー

スタックサイズの再計算	スタック・サイズの再計算を行います。  ボタンのクリックと同様です。
中止	スタック見積もりツールが実行中の処理（スタック・サイズの再計算など）を強制的に中止します。  ボタンのクリックと同様です。

アイコンの整列	リスト表示エリアの関数表示順序を変更します。	
	関数名順	関数名順に並べ替えます。
	アイコン順	アイコンの表示優先度順（高い：  ～ 低い：  ）に並べ替えます。
	スタックサイズ順	スタック・サイズ順に並べ替えます。
	単体スタックサイズ順	単体スタック・サイズ順に並べ替えます。
	加算サイズ順	加算サイズ順に並べ替えます。
	ファイル名順	ファイル名順に並べ替えます。

（c）［オプション］メニュー

サイズ不明関数・サイズ変更関数一覧	単体スタックサイズが不明な関数，情報（加算サイズ，再帰回数，呼び出し関数）の変更が行われた関数，およびスタック見積もりツールが強制的に加算サイズの設定を行った関数を一覧表示するためのサイズ不明関数・サイズ変更関数一覧 ダイアログをオープンします。
スタックサイズ変更 ...	ツリー表示エリア／リスト表示エリアで選択した関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するためのスタックサイズ変更 ダイアログをオープンします。 選択関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するダイアログです。  ボタンのクリックと同様です。
指定関数を初期値に戻す	選択関数の情報（加算サイズ，再帰回数，呼び出し関数）を初期状態に戻します。 選択関数の情報が初期状態から何ら変更されていない場合，本ボタンはグレー表記となります。
全関数を初期値に戻す	全関数の情報（加算サイズ，再帰回数，呼び出し関数）を初期状態に戻します。 関数の情報が初期状態から何ら変更されていない場合，本ボタンはグレー表記となります。

（d）［ヘルプ］メニュー

skcx のヘルプ	スタック見積もりツールのヘルプを表示します。  ボタンのクリックと同様です。
skcx のバージョン情報	skcx のバージョン情報 ダイアログをオープンします。

(2) ツールバー

本エリアは，以下に示したボタン群から構成されています。

	ツリー表示エリア／リスト表示エリアで選択した関数のスタック・サイズ（呼び出し関数のスタック・サイズを含む）が最大となる呼び出し経路を出力結果ファイルとして保存するための名前を付けて保存 ダイアログをオープンします。 ［ファイル］メニュー→［選択した関数の最大経路の保存 ...］の選択と同様です。
	スタック・サイズの再計算を行います。 ［表示］メニュー→［スタックサイズの再計算］の選択と同様です。
	スタック見積もりツールが実行中の処理（スタック・サイズの再計算など）を強制的に中止します。 ［表示］メニュー→［中止］の選択と同様です。
	ツリー表示エリア／リスト表示エリアで選択した関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するためのスタックサイズ変更 ダイアログをオープンします。 ［オプション］メニュー→［スタックサイズ変更 ...］の選択と同様です。
	スタック見積もりツールのヘルプを表示します。 ［ヘルプ］メニュー→［skcx のヘルプ］の選択と同様です。

(3) ツリー表示エリア

関数の呼び出し関係をツリー形式で表示します。

なお，関数名の直前に表示しているアイコンは，以下の意味を持ちます。

	同じ関数から直接呼び出す関数の中でスタック・サイズが最大となる関数
	スタックサイズ変更 ダイアログ，またはスタック・サイズ指定ファイルにより情報（加算サイズ，再帰回数，呼び出し関数）の変更が行われた関数
	再帰関数
	スタック見積もりツールがスタック情報を取得できていない関数
	上記以外の関数

備考　アイコンの表示優先度は，高い：  ～ 低い：  となります。

(a) コンテキスト・メニュー

本エリアの関数を選択したのち，マウスを右クリックすることにより表示するコンテキスト・メニューは，以下のとおりです。

スタックサイズ変更 ...	選択した関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するためのスタックサイズ変更 ダイアログをオープンします。

(4) リスト表示エリア

関数単位のスタック情報（関数名，スタック・サイズ，単体スタック・サイズ，加算サイズ，ファイル名）をリスト形式で表示します。

関数名	関数名を表示します。 なお，本エリアに表示する関数は，第１階層（選択関数）／第２階層（選択関数が直接呼び出している関数）に限られます。
スタック・サイズ	スタック・サイズ（呼び出し関数のスタック・サイズを含む，単位：バイト）を表示します。
単体スタック・サイズ	単体スタック・サイズ（呼び出し関数のスタック・サイズを含まない，単位：バイト）を表示します。
加算サイズ	単体スタック・サイズに対して強制的に加算する値（単位：バイト）を表示します。
ファイル名	ファイル名を表示します。

なお，関数名の直前に表示しているアイコンは，以下の意味を持ちます。

	同じ関数から直接呼び出す関数の中でスタック・サイズが最大となる関数
	スタックサイズ変更 ダイアログ，またはスタック・サイズ指定ファイルにより情報（加算サイズ，再帰回数，呼び出し関数）の変更が行われた関数
	再帰関数
	スタック見積もりツールがスタック情報を取得できていない関数
	上記以外の関数

(a) コンテキスト・メニュー

本エリアの関数を選択したのち，マウスを右クリックすることにより表示するコンテキスト・メニューは，以下のとおりです。

スタックサイズ変更 ...	選択した関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するためのスタックサイズ変更 ダイアログをオープンします。	
アイコンの整列	リスト表示エリアの関数表示順序を変更します。	
	関数名順	関数名順に並べ替えます。
	アイコン順	アイコンの表示優先度順（高い： ～ 低い：）に並べ替えます。
	スタックサイズ順	スタック・サイズ順に並べ替えます。
	単体スタックサイズ順	単体スタック・サイズ順に並べ替えます。
	加算サイズ順	加算サイズ順に並べ替えます。
	ファイル名順	ファイル名順に並べ替えます。

(5) メッセージ表示エリア

スタック見積もりツールの操作ログを表示します。

［注意］

- アセンブリ・ファイル

スタック見積もりツールでは，コンパイラが中間ファイルとして出力する“デッバグ情報の付与されたアセンブリ・ファイル”から各種情報を収集し，スタック・サイズの計算を行っています。
したがって，スタック見積もりツールを使用して，関数単位のスタック情報を得るためには，コンパイル・オプションで“デッバグ情報の付与されたアセンブリ・ファイル”の出力設定が必要となります。

- 静的な解析処理の実行タイミング

スタック見積もりツールでは，起動時に静的な解析処理を実行し，関数の呼び出し関係，および関数単位のスタック情報を本ウインドウに表示しています。
したがって，関数の呼び出し関係，または関数単位のスタック情報が変わるようなこと（ファイルの追加，コンパイル・オプションの変更，ソース・コードの変更など）を行っても，本ウインドウの該当情報は，連動して変化しません。

- 解析対象関数

スタック見積もりツールの解析対象関数は，コンパイラが中間ファイルとして出力した“デバッグ情報の付与されたアセンブリ・ファイル”，またはビルド・ツールが提供しているライブラリ・ファイルに内包されている関数に限られます。
したがって，ユーザが記述したアセンブラ・ソース・ファイル，およびユーザが作成したライブラリ・ファイルに内包されている関数については，解析対象外となるため，スタックサイズ変更 ダイアログを用いて該当情報を設定する必要があります。

- アイコンの表示色

本ウインドウのツリー表示エリア／リスト表示エリアに表示しているアイコンについては，表示優先度（高い：  ～ 低い：  ）を付与しています。
したがって，“同じ関数から直接呼び出す関数の中でスタック・サイズが最大となる関数”を意味する  を表示してる場合であっても，“単体スタック・サイズが不明：  ”などといった優先度の低い情報は GUI 上から隠れるため，注意が必要です。

- 最大スタック・サイズの確定

スタック見積もりツールでは，スタック・サイズが最大となる経路を検出する際，解析対処外の関数については，スタック・サイズが“0 バイト”であるものとして，該当経路の検出を行います。
したがって，最大スタック・サイズを確定する際には，サイズ不明関数・サイズ変更関数一覧 ダイアログの［サイズ不明関数リスト］に関数が表示されていないことを確認する必要があります。

- 再帰関数のツリー表示

本ウインドウのツリー表示エリアでは，再開関数の表示を“2 回目の呼び出しまで”としています。
したがって，“3 回目以降の呼び出し”については，非表示となります。

- ライブラリ関数 bsearch，exit，qsort

スタック見積もりツールでは，ビルド・ツールが提供しているライブラリ・ファイルに内包されている関数であっても，bsearch，exit，qsort については，不明関数として扱います。
したがって，これらの関数を使用する際には，スタックサイズ変更 ダイアログにおいて，各種情報（再帰回数，呼び出し関数など）を設定する必要があります。

- 呼び出し関数

スタック見積もりツールでは，スタックサイズ変更 ダイアログで追加可能な“呼び出し関数”をＣソース・ファイルに内包されている関数，および明示的な呼び出しが行われている関数（ポインタを用いた呼び出しでない）に限定しています。
したがって，スタックサイズ変更 ダイアログの［関数一覧］には，上記の条件に合致した関数のみを表示します。

- 複数の関数から呼び出される関数

スタック見積もりツールでは，複数の関数から呼び出される関数のスタック情報を一意としています。
したがって，該当関数のスタック情報を呼び出し元に応じて変化させることはできません。

例 本ウインドウのツリー表示エリアで func1 から呼び出される sub を選択してオープンしたスタックサイズ変更 ダイアログで各種設定を行った場合，func2 から呼び出される sub についても同様の情報を反映します。

```
int     sub ( int i );
void    func1 ( void );
void    func2 ( void );

void main ( void ) {
    func1 ( );
    func2 ( );
}
int sub ( int i ) {
    i++;
    return ( i );
}

void func1 ( void ) {
    int ret, i = 0;
    ret = sub ( i );
}

void func2 ( void ) {
    int ret, i = 100;
    ret = sub ( i );
}
```

- C ソース内の ASM 文

C ソース内に ASM 文を記述している際には，スタック見積もりツールが“W0594132：不正なフォーマットがアセンブラ・ソース・モジュール・ファイル（*path name*）で見つかりました（*line number* 行）。ファイルを確認してください。”といったメッセージを出力する場合があります。
このような場合，“該当部を #if などを用いて無効化する”，または“該当部をコメントアウトする”といった対処を行ってください。

- 間接的な再帰関数の呼び出し

再帰経路が複数の関数から構成される際には，“スタック・サイズの計算”が正しく行われない場合があります。

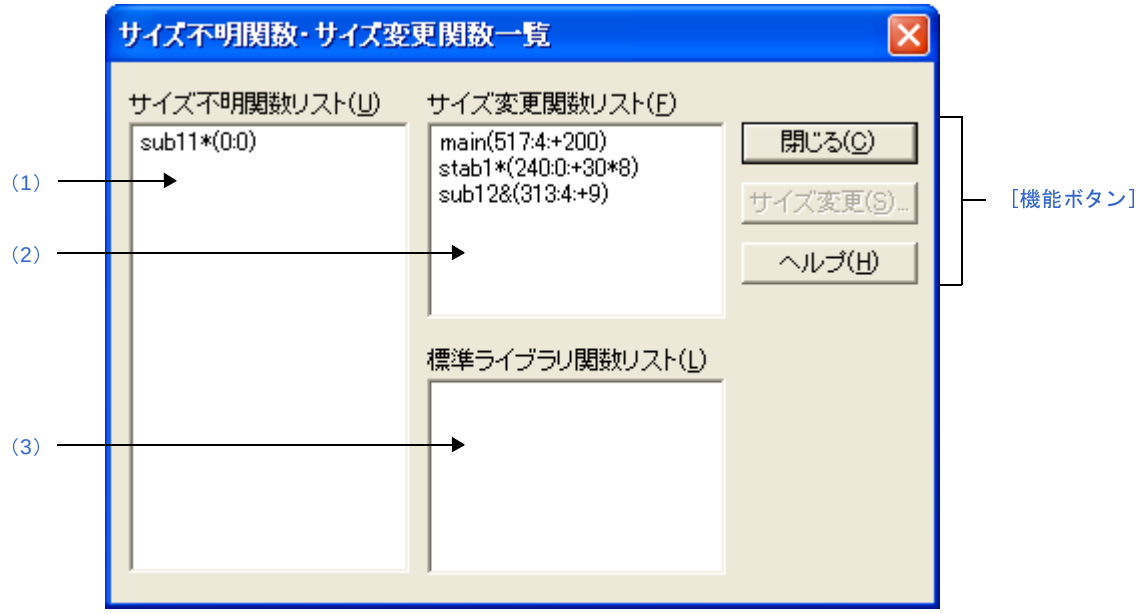
例 再帰関数 func_rec1 ／ func_rec2 の単体スタック・サイズを 8 バイトと仮定し，スタックサイズ変更 ダイアログにおいて，再帰関数 func_rec1 ／ func_rec2 の再帰回数を 3 回と設定した場合，func1 のスタック・サイズは“（8 ＋ 24）× 3”と正しく計算しますが， func2 のスタック・サイズについては“8 × 3”と func_rec1 の呼び出しを無視した計算を行います。

```
void    func_rec1 ( int i );
void    func_rec2 ( int i );
void    func1 ( void );
void    func2 ( void );

void main ( void ) {
    func1 ( );
    func2 ( );
}
void func_rec1 ( int i ) {
    func_rec2 ( i );
}

void func_rec2 ( int i ) {
    if ( i ) {
        func_rec1 ( i - 1 );
    }
}

void func1 ( void ) {
    func_rec1 ( 2 );
}

void func2 ( void ) {
    func_rec2 ( 2 );
}
```

サイズ不明関数・サイズ変更関数一覧 ダイアログ

スタック見積もりツールがスタック情報を取得できていない関数，意図的に情報（加算サイズ，再帰回数，呼び出し関数）の変更が行われた関数，およびスタック見積もりツールが強制的に加算サイズの設定を行った関数を一覧表示します。

図 A―52 サイズ不明関数・サイズ変更関数一覧 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- Stack Usage Tracer ウインドウの［オプション］メニュー→［サイズ不明関数・サイズ変更関数一覧 ...］を選択

［各エリアの説明］

(1)［サイズ不明関数リスト］

スタック見積もりツールがスタック情報を取得できていない関数“不明関数”を一覧表示します。

なお，本エリアでは，不明関数を基本的に以下の形式で表示します。

関数名（スタック・サイズ：単体スタック・サイズ）

備考 1. 不明関数が“アセンブリ言語で記述された関数”の場合，シンボル名の先頭に付与されている“_”を削ったのち，“[]”で囲んだものを関数名として表示します。

2. 不明関数が“再帰関数”の場合，関数名の直後に“*”を表示します。
3. 不明関数が“関数ポインタを用いた間接呼び出しを含む関数”の場合，関数名の直後に“&”を表示します。
4. 不明関数が“スタティック関数”の場合，関数名の直前に“ファイル名 #”を表示します。

(2)［サイズ変更関数リスト］

スタックサイズ変更 ダイアログ，またはスタック・サイズ指定ファイルにより意図的に情報（加算サイズ，再帰回数，呼び出し関数）の変更が行われた関数“変更関数”を一覧表示します。

なお，本エリアでは，変更関数を基本的に以下の形式で表示します。

関数名（スタック・サイズ：単体スタック・サイズ：加算サイズ）

備考 1. 変更関数が“アセンブリ言語で記述された関数”の場合，シンボル名の先頭に付与されている“_”を削ったのち，“[]”で囲んだものを関数名として表示します。
2. 変更関数が“再帰関数”の場合，関数名の直後に“*”を表示します。
3. 変更関数が“関数ポインタを用いた間接呼び出しを含む関数”の場合，関数名の直後に“&”を表示します。
4. 変更関数が“スタティック関数”の場合，関数名の直前に“ファイル名 #”を表示します。
5. スタックサイズ変更 ダイアログにおいて，“呼び出し関数の追加”のみが行われた関数については，本エリアの表示内容が以下のようになります。

   関数名（スタック・サイズ：単体スタック・サイズ）

(3)［標準ライブラリ関数リスト］

単体スタック・サイズが不明な関数のうち，スタック見積もりツールが強制的に加算サイズの設定を行ったライブラリ関数“自動設定関数”を一覧表示します。

なお，本エリアでは，自動設定関数を基本的に以下の形式で表示します。

関数名（スタック・サイズ：？：加算サイズ）

備考 1. シンボル名の先頭に付与されている“_”を削ったのち，“[]”で囲んだものを関数名として表示します。
2. スタック見積もりツールが保有するデータ・ベースの中から該当ライブラリ関数に適切なスタック・サイズを“加算サイズ”として設定しています。

［機能ボタン］

ボタン	機能
閉じる	本ダイアログをクローズします。
サイズ変更 ...	［サイズ不明関数リスト］／［サイズ変更関数リスト］／［標準ライブラリ関数リスト］で選択した関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するためのスタックサイズ変更 ダイアログをオープンします。
ヘルプ	本ダイアログのヘルプを表示します。

スタックサイズ変更 ダイアログ

選択関数に対する情報（加算サイズ，再帰回数，呼び出し関数）を変更するダイアログです。

図 A—53　スタックサイズ変更 ダイアログ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- Stack Usage Tracer ウインドウのツリー表示エリア／リスト表示エリアにおいて，関数を選択したのち，［オプション］メニュー→［スタックサイズ変更 ...］を選択
- Stack Usage Tracer ウインドウのツリー表示エリア／リスト表示エリアにおいて，関数を選択したのち，ツールバー→  ボタンをクリック
- Stack Usage Tracer ウインドウのツリー表示エリア／リスト表示エリアにおいて，関数を選択したのち，コンテキスト・メニューから［スタックサイズ変更 ...］を選択

- サイズ不明関数・サイズ変更関数一覧 ダイアログの［サイズ不明関数リスト］／［サイズ変更関数リスト］／［標準ライブラリ関数リスト］において，関数を選択したのち，［サイズ変更 ...］ボタンをクリック

［各エリアの説明］

(1)［関数名］

選択関数の関数名を表示します。

備考 1. 選択関数が“アセンブリ言語で記述された関数”，または“ライブラリ関数”の場合，シンボル名の先頭に付与されている“_”を削ったのち，“[]”で囲んだものを関数名として表示します。

2. 選択関数が“再帰関数”の場合，関数名の直後に“*”を表示します。

3. 選択関数が“関数ポインタを用いた間接呼び出しを含む関数”の場合，関数名の直後に“&”を表示します。

4. 選択関数が“スタティック関数”の場合，関数名の直前に“ファイル名 #”を表示します。

(2)［単体スタックサイズ］

選択関数の単体スタック・サイズ（呼び出し関数のスタック・サイズを含まない，単位：バイト）を表示します。

備考 単体スタック・サイズが不明な場合は“?”を，限界値を越えている場合は“SIZEOVER”を表示します。

(3)［加算サイズ］

選択関数の単体スタック・サイズに対して強制的に加算する値（単位：バイト）を 10 進数，または“0x”／“0X”で始まる 16 進数で指定します。

(4)［再帰回数］

選択関数の再帰回数を 10 進数，または“0x”／“0X”で始まる 16 進数で指定します。

備考 選択関数が“再帰関数以外”の場合，本項目はグレー表記となります。

(5)［呼び出し関数リスト（関数ポインタ向け）］エリア

(a)［呼び出し関数一覧］

選択関数からの呼び出し関数（関数ポインタなどを用いて間接的に呼び出される関数）を一覧表示します。

なお，本エリアでは，呼び出し関数を基本的に以下の形式で表示します。

関数名（スタック・サイズ：単体スタック・サイズ：加算サイズ）

備考 1. 呼び出し関数が“アセンブリ言語で記述された関数”，または“ライブラリ関数”の場合，シンボル名の先頭に付与されている“_”を削ったのち，“[]”で囲んだものを関数名として表示します。

2. 呼び出し関数が“再帰関数”の場合，関数名の直後に“*”を表示します。

3. 呼び出し関数が“関数ポインタを用いた間接呼び出しを含む関数”の場合，関数名の直後に“&”を表示します。

4. 呼び出し関数が“スタティック関数”の場合，関数名の直前に“ファイル名 #”を表示します。

5. ［追加］ボタンのクリックにより，［関数一覧］から意図的に追加した関数については，関数名の直前に“+”を表示します。

（b）［関数一覧］

選択関数からの呼び出し関数として追加可能な関数を一覧表示します。

なお，本エリアでは，追加可能な関数を基本的に以下の形式で表示します。

関数名（スタック・サイズ：単体スタック・サイズ：加算サイズ）

備考 1. 追加可能な関数が“アセンブリ言語で記述された関数”，または“ライブラリ関数”の場合，シンボル名の先頭に付与されている“_”を削ったのち，“[]”で囲んだものを関数名として表示します。

2. 追加可能な関数が“再帰関数”の場合，関数名の直後に“*”を表示します。

3. 追加可能な関数が“関数ポインタを用いた間接呼び出しを含む関数”の場合，関数名の直後に“&”を表示します。

4. 追加可能な関数が“スタティック関数”の場合，関数名の直前に“ファイル名 #”を表示します。

（c）ボタン・エリア

追加	［関数一覧］で選択した関数を［呼び出し関数一覧］に追加します。 ［関数一覧］で関数が未選択の場合，本ボタンはグレー表記となります。
削除	［呼び出し関数一覧］で選択した関数を［呼び出し関数一覧］から削除します。 ［呼び出し関数一覧］で関数が未選択の場合，本ボタンはグレー表記となります。

備考 ［呼び出し関数一覧］から削除可能な関数は，関数名の直前に“+”が付与されているもの（［追加］ボタンのクリックにより，［関数一覧］から意図的に追加した関数）に限られます。

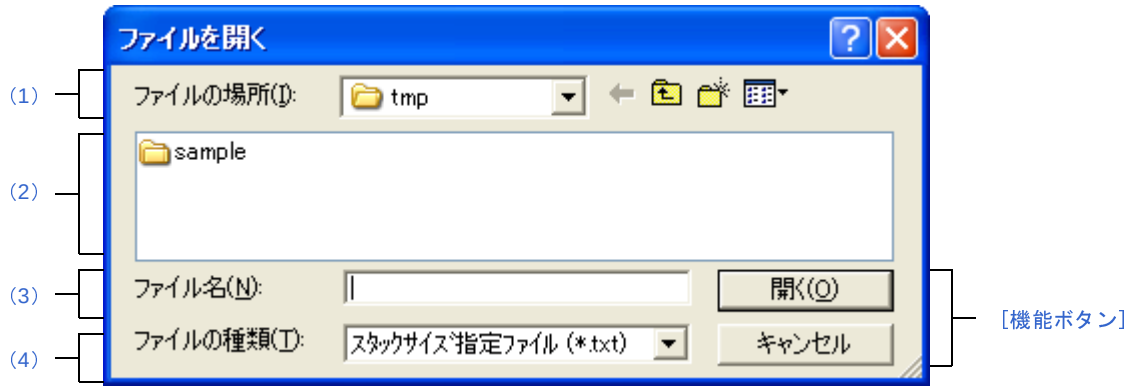
［機能ボタン］

ボタン	機能
OK	設定内容を Stack Usage Tracer ウインドウに反映／プロジェクト・ファイル（*.prj）に保存したのち，本ダイアログをクローズします。
キャンセル	設定内容を無効とし，本ダイアログをクローズします。
初期化	選択関数の情報（加算サイズ，再帰回数，呼び出し関数）を初期状態に戻します。 選択関数の情報が初期状態から何ら変更されていない場合，本ボタンはグレー表記となります。
ヘルプ	本ダイアログのヘルプを表示します。

ファイルを開く ダイアログ

既存のスタック・サイズ指定ファイルを開きます。

図 A—54 ファイルを開く ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

[オープン方法]

- Stack Usage Tracer ウインドウの［ファイル］メニュー→［スタックサイズ指定ファイルを開く ...］を選択

[各エリアの説明]

(1)［ファイルの場所］エリア

開きたいスタック・サイズ指定ファイルが存在するフォルダを選択します。

(2) ファイルの一覧エリア

［ファイルの場所］エリア，および［ファイルの種類］エリアで選択した条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

開くスタック・サイズ指定ファイルのファイル名を指定します。

(4)［ファイルの種類］エリア

開くファイルの種類（ファイル・タイプ）を選択します。

スタックサイズ指定ファイル（*.txt）	テキスト形式

［機能ボタン］

ボタン	機能
開く	指定したファイルを開きます。
キャンセル	設定内容を無効とし，本ダイアログをクローズします。

付録B　コマンド・リファレンス

ここでは，ビルド・ツールに含まれる各コマンドの仕様についての詳細を説明します。

B. 1　cx

cx は，C 言語やアセンブリ言語で記述したソース・プログラムから，ターゲット・システムで実行可能なファイルを生成します。

cx では，1 つのドライバがプリプロセッサからオブジェクト・コンバータまでの全フェーズを制御します。

各フェーズの処理について説明します。

（1）コンパイラ

C ソース・プログラムに対して，プリプロセス指令の処理，コメント処理，最適化を行い，アセンブラ・ソース・ファイルを生成します。

（a）プリプロセッサ

C ソース・プログラム中のプリプロセス指令の処理を行います。

-P オプション指定時のみ，プリプロセス処理済みファイルを出力します。

備考　デフォルトでは，-P オプションは未指定です。

（b）構文解析部

C ソース・プログラムの構文解析処理を行ったのち，コンパイラの内部データ表現に変換します。

（c）共通最適化部

C ソース・プログラムを変換した内部データ表現に対して最適化を行います。

（d）コード生成部

内部データ表現をアセンブラ・ソース・プログラムに変換します。

（2）アセンブラ

アセンブラ・ソース・プログラムを機械語命令に変換して，再配置可能なオブジェクト・モジュール・ファイルを生成します。

（3）リンカ

オブジェクト・モジュール・ファイル，リンク・ディレクティブ・ファイル，ライブラリ・ファイルをリンクし，ターゲット・システムで実行可能なオブジェクト・ファイル（ロード・モジュール・ファイル）を生成します。

（4）シンボル・ファイル・ジェネレータ

シンボル情報ファイルを生成します。

-Xsfg オプション指定時のみ起動します。

生成したシンボル情報ファイルは，-Xsymbol_file オプションで指定することにより，コンパイラが利用することができます。

備考 1. デフォルトでは，-Xsfg，および -Xsymbol_file オプションは未指定です。

2. シンボル情報ファイルを生成する際は，-Xcref オプションも同時に指定する必要があります。

3. コンパイラがシンボル情報ファイルを参照するには，シンボル情報ファイルを生成する際に -Xsfg_opt オプションも同時に指定する必要があります。

（5）ROM 化プロセッサ

ロード・モジュール・ファイルに対して ROM 化処理を行います。

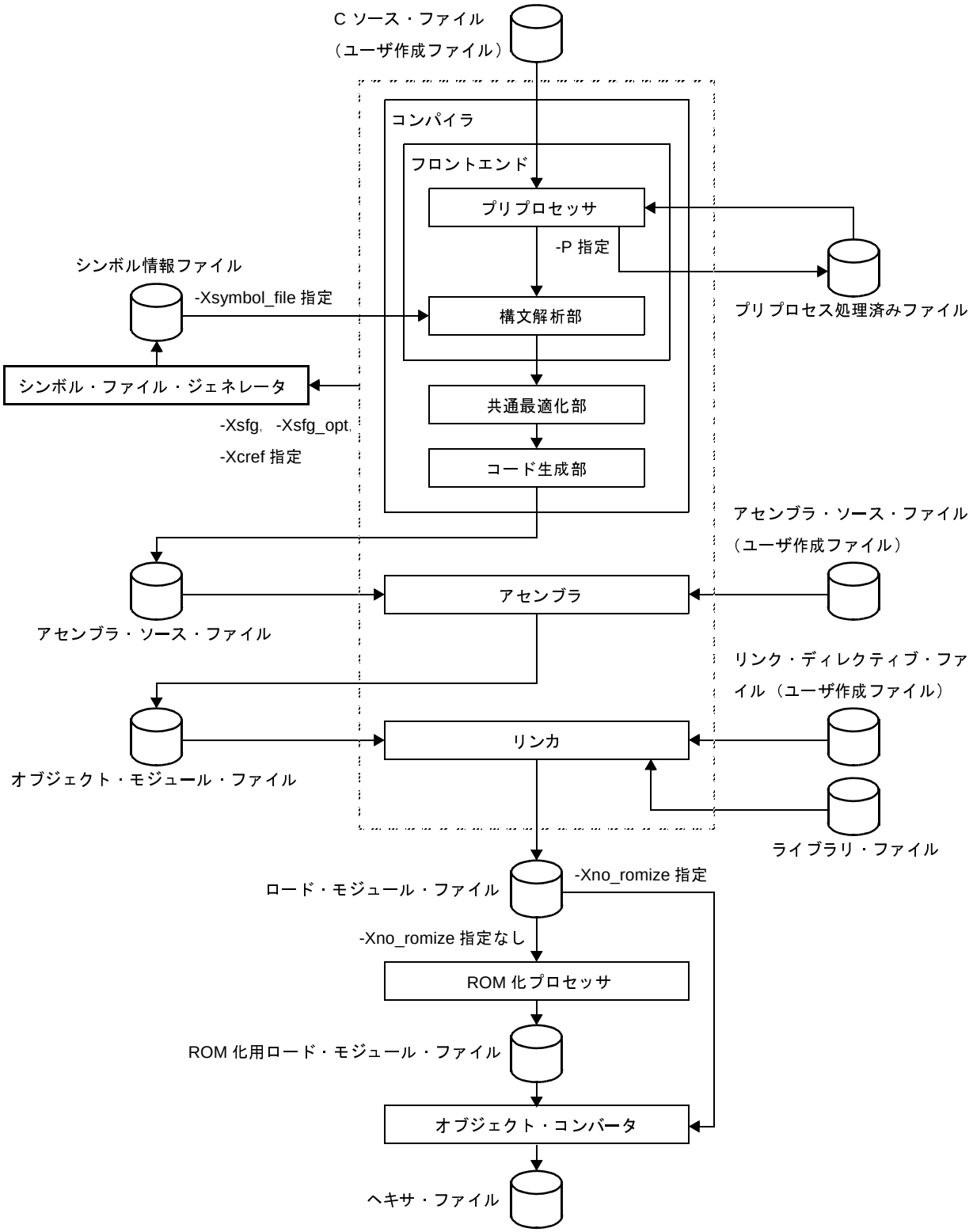
-Xno_romize オプション指定時は，起動しません。

備考 デフォルトでは，-Xno_romize オプションは未指定であるため，cx が生成するロード・モジュール・ファイルは，ROM 化処理後のものとなります。

（6）オブジェクト・コンバータ

ロード・モジュール・ファイルに対してヘキサ変換処理を行い，ヘキサ・ファイルを生成します。

図 B—1　cx における処理の流れ



B. 1. 1 入出力ファイル

cx の入出力ファイルを以下に示します。

表 B—1 cx の入出力ファイル

ファイル種別	拡張子	入出力	説明
C ソース・ファイル	.c	入力	C 言語で記述したソース・ファイル ユーザ作成ファイルです。
プリプロセス処理済みファイル	.i 注 1	出力	入力ファイルに対してプリプロセス処理を実行した結果を出力したファイル ASCII イメージ・ファイルです。 -P オプション指定時に出力します。
アセンブラ・ソース・ファイル	.asm 注 1	出力	コンパイルにより C ソースから生成したアセンブリ言語ファイル
	asm .s	入力	アセンブリ言語で記述したソース・ファイル ユーザ作成ファイルです。
ヘッダ・ファイル	任意	入力	ソース・ファイルで参照するファイル C 言語，またはアセンブリ言語で記述したファイルです。 ユーザ作成ファイルです。 拡張子は任意ですが，以下を推奨します。 - #include 指令 : .h - $include 制御命令 : .inc
シンボル情報ファイル 注 2	.sfg 注 1	入出力	C ソース・ファイル中で定義した変数に対して，配置先のセクションを指定するファイル -Xsfg オプション指定時に出力します。
オブジェクト・モジュール・ファイル	.obj 注 1	入出力	機械語情報と機械語の配置アドレスに関する再配置情報，およびシンボル情報を含んだ ELF 形式ファイル
	.o	入力	
アセンブル・リスト・ファイル 注 2	.prn 注 1	出力	アセンブル結果の情報を持つリスト・ファイル -Xprn_path オプション指定時に出力します。
リンク・ディレクティブ・ファイル	任意	入力	リンクの際のメモリ配置情報を記述したファイル ユーザ作成ファイルです。 拡張子は任意ですが，.dir を推奨します。
ライブラリ・ファイル	.lib .a	入力	複数のオブジェクト・モジュール・ファイルが登録されたファイル
ロード・モジュール・ファイル	.lmf 注 1	入出力	リンク結果のオブジェクト・コードの ELF 形式ファイル ヘキサ・ファイルを出力する際の入力ファイルとなります。
ヘキサ・ファイル 注 2	.hex 注 1	出力	ロード・モジュール・ファイルをヘキサ・フォーマットに変換したファイル

ファイル種別	拡張子	入出力	説明
リンク・マップ・ファイル注2	.map 注1	出力	リンク結果の情報を持つリスト・ファイル -Xmap オプション指定時に出力します。
静的解析情報ファイル	任意	入出力	本製品が利用する情報を持つファイル 拡張子は任意ですが，.cref を推奨します。 -Xcref オプション指定時に出力します。
エラー・メッセージ・ファイル	任意	出力	エラー・メッセージを内容とするファイル -Xerror_file オプション指定時に出力します。
コマンド・ファイル	任意	入力	実行プログラムのパラメータを内容とするファイル ユーザ作成ファイルです。

注 1. 出力ファイルについては，オプション指定により拡張子が変更可能です。

2. 出力ファイルについての詳細は，「第 3 章 ビルドの出力リスト」を参照してください。

B. 1. 2 操作方法

ここでは，cx の操作方法について説明します。

（1）コマンド・ラインでの操作方法

コマンドは，コマンド・ラインで以下のように入力します。

```
>cx [ △オプション ] . . . △ファイル名 [ △ファイル名，またはオプション ] . . .
```

[ ] ： [ ] 内は省略可能です。

... ： 直前の [ ] 内のパターンの繰り返しが可能です。

△ ： 1 個以上の空白を示します。

- オプションの大文字／小文字は区別します。
- オプションのパラメータとして数値を指定する場合は，10 進数，または“0x”（“0X”）で始まる 16 進数での指定が可能です。
  16 進数のアルファベットは，大文字／小文字を区別しません。
  ただし，-Xhex_fill オプションのパラメータは，16 進数のみ指定可能です。
- オプションのパラメータとしてファイル名を指定する場合は，パス付き（絶対パス，または相対パス）での指定が可能です。
  パスなし，および相対パスで指定する場合は，カレント・フォルダを基準とします。
- オプションのパラメータ中に空白を含める場合（パス名など）は，そのパラメータ全体をダブルクォーテーション（”“）で囲んでください。
- ファイル名は，Windows で認められるものであれば指定可能です。
  ただし，“@”はコマンド・ファイル指定と判断するため，“@”をファイル名の先頭文字として使用することはできません。

また，“-”，“+”はオプション指定と判断するため，“-”，“+”もファイル名の先頭文字として使用することはできません。

- 指定可能なファイル名の長さは，Windows に依存します（259 文字まで）。
- ファイル名のアルファベットは，大文字／小文字を区別しません。
- -P オプション指定時は，入力として指定可能なファイルは 1 個です。
  複数のファイルを指定した場合は，エラーとなります。
  それ以外の場合は，入力として複数のファイルが指定可能です。
  異なる種別のファイル（C ソース・ファイルとアセンブラ・ソース・ファイル，またはオブジェクト・モジュール・ファイルなど）を混在して指定することも可能です。
  ただし，同名のファイルを複数指定することはできません（異なるフォルダにある場合も含みます）。
  複数のファイルを指定した場合は，1 つのファイルでエラーとなっても，残りのファイルの処理が継続可能であれば，処理を継続します。
- 生成したオブジェクト・モジュール・ファイルはリンク後も削除しません。

コマンド・ラインでの操作例を以下に示します。

備考 各オプションについての詳細は，「B. 1. 3 オプション」を参照してください。

（a）コンパイルからリンクまでを一度に行う場合

file1.c，file2.asm，file3.obj を読み込み，ロード・モジュール・ファイル a.lmf を生成します。
また，ヘキサ・ファイル a.hex も生成します。

```
>cx -CF3746 file1.c file2.asm file3.obj
```

（b）コンパイルからアセンブルまでを行い，リンクは単独で行う場合

- file1.c，file2.asm を読み込み，オブジェクト・モジュール・ファイル file1.obj，file2.obj を生成します。

```
>cx -CF3746 -c file1.c file2.asm
```

- file1.obj，file2.obj，file3.obj をリンクし，ロード・モジュール・ファイル a.lmf を生成します。
  また，ヘキサ・ファイル a.hex も生成します。

```
>cx -CF3746 file1.obj file2.obj file3.obj
```

（c）コンパイル，アセンブル，リンクともに単独で行う場合

- file1.c を読み込み，オブジェクト・モジュール・ファイル file1.obj を生成します。

```
>cx -CF3746 -c file1.c
```

- file2.asm を読み込み，オブジェクト・モジュール・ファイル file2.obj を生成します。

```
>cx -CF3746 -c file2.asm
```

- file1.obj，file2.obj，file3.obj をリンクし，ロード・モジュール・ファイル a.lmf を生成します。
  また，ヘキサ・ファイル a.hex も生成します。

```
>cx -CF3746 file1.obj file2.obj file3.obj
```

（2）コマンド・ファイルによる操作方法

コマンド・ファイルとは，cx コマンドに対して指定するオプションやファイル名を記述したファイルです。
cx コマンドは，コマンド・ファイルの内容をコマンド・ラインの引数のように扱います。
コマンド・ファイルは，コマンド・ラインで引数を指定しきれない場合や，コマンドを実行するたびに同じオプションを繰り返し指定するような場合に使用します。

（a）コマンド・ファイルの記述に関する注意事項

- 指定する引数は，複数行に分けて記述することができます。
  ただし，オプションやファイル名の途中で改行することはできません。
- コマンド・ファイルをネストすることはできません。
- コマンド・ファイルの文字コードは，-Xcharacter_set オプションで指定することはできません。
  コマンド・ファイル中に ASCII 文字以外を使用する場合は，BOM 付き UTF-8，または SJIS のファイルにしてください。
- 以下の文字は，特殊文字として扱います。
  これらの特殊文字自体は，cx コマンドへのコマンド・ラインの中に含まずに削除します。

"（ダブルクォーテーション）	次のダブルクォーテーションまでの文字列を連続した文字列として扱います。
#（シャープ）	行頭に指定した場合は，行末までをコメントとして扱います。
^（ハット）	直後の文字を特殊文字として扱いません。

（b）コマンド・ファイルの指定例

コマンド・ファイル cfile をエディタで作成します。

```
-CF3746          … デバイスを指定
-Utest           … マクロ test の定義を解除
-oobject.obj     … オブジェクト・モジュール・ファイル名を指定
-c               … アセンブル・フェーズまでの実行を指定
file.c           … コンパイルするファイルを指定
```

コマンド・ラインで cx コマンドを実行する際，コマンド・ファイル指定オプション @ により，コマンド・ファイル cfile を指定します。

```
>type cfile      ← cfile の内容を表示します。
-CF3746
-Utest
-oobject.obj
-c
file.c
>cx @cfile       ← cx -CF3746 -Utest -oobject.obj -c file.c と同じ動作をします。
```

(3) CubeSuite+ でのオプション設定

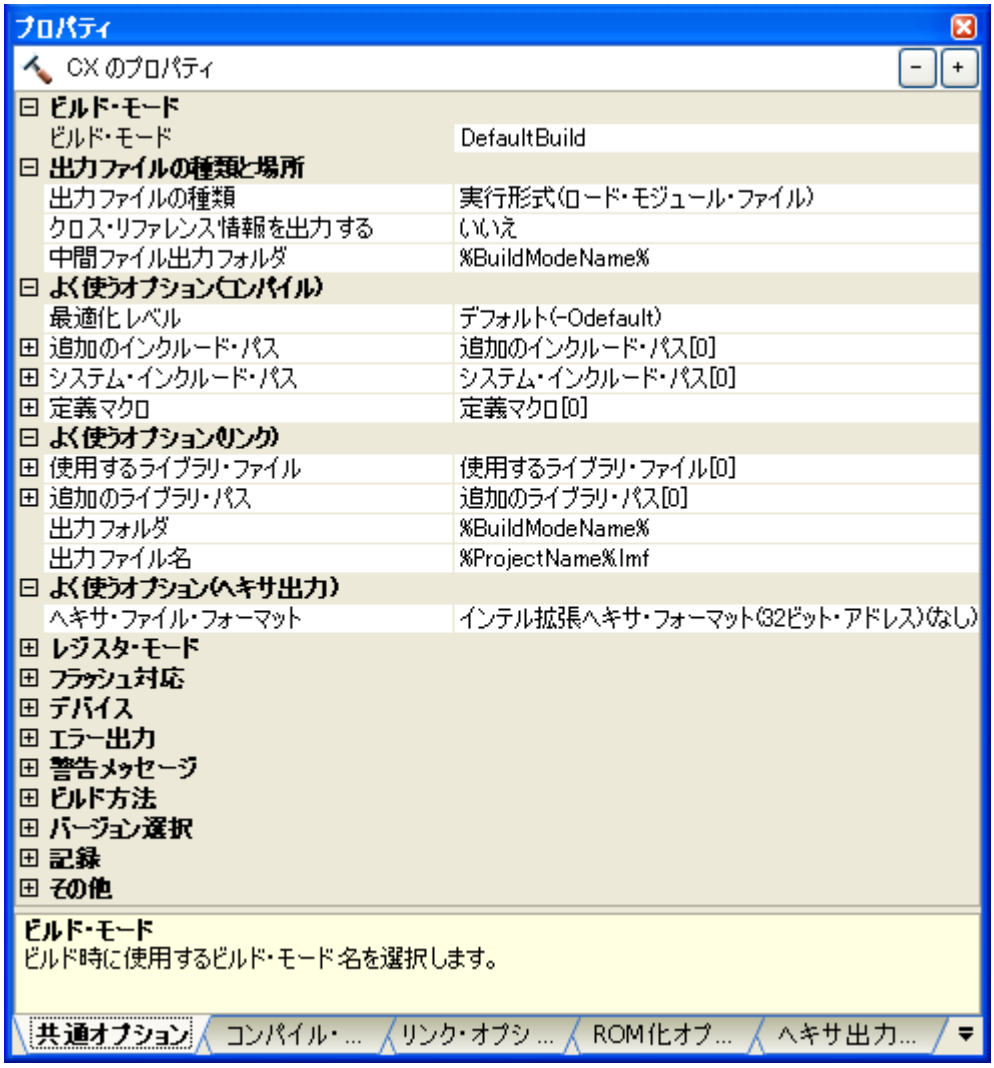
CubeSuite+ から cx オプションを設定する方法について説明します。

CubeSuite+ のプロジェクト・ツリー パネルにおいて，ビルド・ツール・ノードを選択したのち，［表示］メニュー→［プロパティ］を選択すると，プロパティ パネルがオープンします。

次に，［共通オプション］タブ／［コンパイル・オプション］タブ／［アセンブル・オプション］タブ／［リンク・オプション］タブ／［ROM 化オプション］タブ／［ヘキサ出力オプション］タブを選択します。

タブ上で各プロパティを設定することにより，対応する cx オプションを設定することができます。

図 B―2 プロパティ パネル



B. 1. 3　オプション

ここでは，cx のオプションについて，各フェーズごとに説明します。

コンパイル・フェーズ　→　「(1) コンパイル・オプション」参照
アセンブル・フェーズ　→　「(2) アセンブル・オプション」参照
リンク・フェーズ　→　「(3) リンク・オプション」参照
ROM 化フェーズ　→　「(4) ROM 化オプション」参照
ヘキサ出力フェーズ　→　「(5) ヘキサ出力オプション」参照

(1) コンパイル・オプション

コンパイル・フェーズのオプションの分類と説明を以下に示します。

表 B—2 コンパイル・オプション

分類	オプション	説明
バージョン／ヘルプ表示指定	-V	cx のバージョン情報を表示します。
	-h	cx のオプションの説明を表示します。
出力ファイル指定	-o	出力ファイル名を指定します。
	-Xobj_path	コンパイル途中に生成されるオブジェクト・モジュール・ファイルの保存先を指定します。
	-Xasm_path	コンパイル途中に生成されるアセンブラ・ソース・ファイルの保存先を指定します。
	-Xprn_path	アセンブル・リスト・ファイルの保存先を指定します。
	-Xtemp_path	作業用フォルダを指定します。
	-Xlink_output	ROM 化処理前のロード・モジュール・ファイルを保存します。
ソース・デバッグ制御	-g	ソース・デバッグ用の情報を出力します。
	-Xkeep_access_size	メモリ・アクセス・サイズの変更を禁止します。
デバイス指定	-C	ターゲット・デバイスを指定します。
	-Xcommon	デバイス共通のオブジェクト・モジュール・ファイルを生成することを指定します。
	-Xdev_path	デバイス・ファイルを検索するフォルダを指定します。
	-Xprogrammable_io	プログラマブル周辺 I/O レジスタの開始アドレスを指定します。
処理中断指定	-P	入力ファイルに対してプリプロセス処理のみ実行します。
	-S	アセンブル以降の処理を実行しません。
	-c	リンク以降の処理を実行しません。
プリプロセッサ制御	-D	プリプロセッサ・マクロを定義します。
	-U	-D オプションによるプリプロセッサ・マクロの定義を解除します。
	-I	インクルード・ファイルを検索するフォルダを指定します。
	-Xpreprocess	プリプロセス結果の出力を制御します。
C 言語制御	-Xansi	C ソース・プログラムを ANSI 規格に厳密にあわせて処理します。
	-Xchar	char 型に対して，符号を指定します。
	-Xenum_type	列挙型に対して，どの整数型として扱うかを指定します。
	-Xdef_var	変数の仮定義を定義として扱います。

分類	オプション	説明
変数／関数情報指定	-Xsymbol_file	シンボル情報ファイルを利用します。
日本語／中国語文字列制御	-Xcharacter_set	日本語／中国語の文字コードを指定します。
最適化指定	-O	最適化のレベル，または各最適化項目の詳細を指定します。
	-Xdelete_func	未使用関数を削除します。
	-Xsort_var	外部変数をソートします。
	-Xinline_strcpy	標準ライブラリ関数 strcpy，strcmp，memcpy，memset の呼び出しをインライン展開します。
	-Xpro_epi_runtime	関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にするかどうかを指定します。
	-Xcall_lib	ライブラリ関数のインライン展開を抑止します。
	-Xmerge_string	文字列定数をマージします。
生成コード制御	-Xpack	構造体パッキングを行います。
	-Xpass_source	出力するアセンブラ・ソース・ファイル中に C ソース・プログラムをコメントとして出力します。
	-Xswitch	switch 文のコード出力方式を指定します。
	-Xword_case	4 バイトの分岐テーブルを生成します。
	-Xr	外部変数の割り付け先のレジスタを指定します。
	-Xreg_mode	レジスタ・モードを指定します。
	-Xsdata	.sdata セクション，または .sbss セクションに配置するデータの最大サイズを指定します。
	-Xsconst	.sconst セクションへの定数データの配置を指定します。
	-Xfloat【V850E2V3】	浮動小数点演算命令の生成を制御します。
	-Xfar_jump	far jump の出力を制御します。
	-Xdiv【V850E2V3】	除算に対して，div，および divu 命令を生成します。
アセンブラ制御指定	-Xasm_far_jump【V850E2】【V850E2V3】	アセンブラ・ソース・ファイルに対して，far jump の出力を制御します。
ライブラリ・リンク制御	-l	リンク時に使用するライブラリ・ファイルを指定します。
	-L	ライブラリ・ファイルを検索するフォルダを指定します。
	-Xno_stdlib	標準ライブラリのリンクを抑止します。
	-Xno_startup	標準スタートアップ・ルーチンのリンクを抑止します。
	-Xstartup	スタートアップ・ルーチンを指定します。

分類	オプション	説明
リンク制御	-Xlink_directive	リンク・ディレクティブ・ファイルを指定します。
	-Xmap	リンク・マップ・ファイルを出力します。
	-Xsymbol_dump	リンク・マップ・ファイルにシンボル情報を出力します。
	-Xsecurity_id	セキュリティ ID を設定します。
	-Xoption_byte	ユーザ・オプション・バイトを設定します。
	-Xentry_address	エントリ・ポイント・アドレスを指定します。
	-Xrelinkable_object	再配置可能なオブジェクト・モジュール・ファイルを生成します。
	-Xregmode_info	異なるレジスタ・モードが混在している場合，詳細情報を出力します。
	-Xforce_link	内部 ROM/RAM のオーバフロー時に，リンク処理を続行します。
	-Xsdata_info	-Xsdata オプションのパラメータに対して，目安として用いることのできる情報を標準出力に出力します。
	-Xtwo_pass_link	2 パス・モードでリンクを行います。
	-Xignore_address_error	リンク時のリロケーション処理において，不正箇所があった場合，リンク処理を続行します。
	-Xmultiple_symbol	多重定義されたすべての外部シンボルに対してエラー・メッセージを出力します。
	-Xlink_check_off	リンク時のチェックを抑止します。
	-Xalign_fill	アライン・ホールの充てん値を指定します。
	-Xrescan	-l オプションで指定したライブラリ・ファイルの再スキャンを行います。
	-Xstrip	デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。
	-Xflash	フラッシュ領域側のロード・モジュール・ファイルを生成します。
	-Xflash_ext_table	ブート－フラッシュ再リンク機能用の分岐テーブル先頭アドレス値を指定します。

分類	オプション	説明
ROM 化制御	-Xno_romize	ROM 化処理を抑止します。
	-Xrompcrt	ROM 化用領域確保コード・ファイルを指定します。
	-Xrompsec_start	rompsec セクションの先頭アドレスを指定します。
	-Xrompsec_data	rompsec セクションに含めるデータ・セクションを指定します。
	-Xrompsec_text	rompsec セクションに含めるテキスト・セクションを指定します。
	-Xrompsec_only	rompsec セクションのみを持つロード・モジュール・ファイルを生成します。
	-Xromize_check_off	ROM 化時のエラー・チェックを省略します。
ヘキサ出力制御	-Xhex	ヘキサ・ファイル名を指定します。
	-Xhex_only	ヘキサ出力のみ実行します。
	-Xhex_format	出力するヘキサ・ファイルのフォーマットを指定します。
	-Xhex_fill	ヘキサ・ファイルの充てん処理を指定します。
	-Xhex_section	指定したセクションのコードをヘキサ変換して，出力します。
	-Xhex_block_size	ブロック長の最大値を指定します。
	-Xhex_offset	出力するアドレスのオフセットを指定します。
	-Xhex_null	初期値なしデータのセクションに対して，セクションのサイズ分だけ null 文字を生成します。
	-Xhex_symtab	シンボル・テーブルを変換して，出力します。
	-Xhex_rom_less	ヘキサ・ファイルの充てん時に，内蔵 ROM 領域情報を使用しません。
	-Xcrc	CRC 演算結果を出力します。
	-Xcrc_method	CRC 演算方法を指定します。
マルチコア対応指定	-Xmulti	マルチコア用プログラムのサブプログラムの生成を指定します。
	-Xmulti_link	マルチコア用サブプログラムのリンクを指定します。
情報ファイル出力制御	-Xcref	静的解析情報ファイルを出力します。
	-Xno_cref	静的解析情報ファイルの出力を抑止します。
	-Xsfg	シンボル情報ファイルを生成します。
	-Xsfg_opt	最適な配置情報を出力します。
	-Xsfg_size_tidata	.tidata セクションのサイズを指定します。
	-Xsfg_size_tidata_byte	.tidata.byte セクションのサイズを指定します。
	-Xsfg_size_sidata	.sidata セクションのサイズを指定します。
	-Xsfg_size_sedata	.sedata セクションのサイズを指定します。
	-Xsfg_size_sdata	.sdata セクションのサイズを指定します。
エラー出力制御	-Xerror_file	エラー・メッセージをファイルに出力します。

分類	オプション	説明
警告メッセージ出力制御	-Xwarning	指定した警告メッセージを出力します。
	-Xno_warning	指定した警告メッセージの出力を抑止します。
フェーズ個別オプション指定	-Xasm_option	アセンブル対象ファイルを指定します。
	-Xlk_option	リンク対象ファイルを指定します。
	-Xopt_option	共通最適化部オプションを指定します。
コマンド・ファイル指定	@	コマンド・ファイルを指定します。

表 B―3　オプション説明でのマーク

【V850E2】	V850E2 コアで命令セット・アーキテクチャが V850E2 であるデバイス専用のオプション
【V850E2V3】	V850E2 コアで命令セット・アーキテクチャが V850E2V3 であるデバイス専用のオプション

バージョン／ヘルプ表示指定

バージョン／ヘルプ表示指定オプションには，次のものがあります。

- -V
- -h

-V

cx のバージョン情報を表示します。

[指定形式]

```
-V
```

- 省略時解釈

  cx のバージョン情報を表示せずに，コンパイルを行います。

[詳細説明]

- cx のバージョン情報を標準エラー出力に出力します。

  コンパイルは行いません。

[使用例]

- cx のバージョン情報を標準エラー出力に出力します。

```
>cx -CF3746 -V
```

-h

cx のオプションの説明を表示します。

[指定形式]

```
-h
```

- 省略時解釈

  cx のオプションの説明を表示しません。

[詳細説明]

- cx のオプションの説明を標準エラー出力に出力します。

  コンパイルは行いません。

[使用例]

- cx のオプションの説明を標準エラー出力に出力します。

```
>cx -CF3746 -h
```

出力ファイル指定

出力ファイル指定オプションには，次のものがあります。

- -o
- -Xobj_path
- -Xasm_path
- -Xprn_path
- -Xtemp_path
- -Xlink_output

-o

出力ファイル名を指定します。

[指定形式]

```
-ofile
```

- 省略時解釈

  カレント・フォルダにファイルを出力します。

  - -P オプションと同時に指定した場合

    出力ファイル名は，入力ファイル名の拡張子を .i に置き換えたものとなります。

  - -S オプションと同時に指定した場合

    出力アセンブラ・ソース・ファイル名は，ソース・ファイル名の拡張子を .asm に置き換えたものとなります。

  - -c オプションと同時に指定した場合

    出力オブジェクト・モジュール・ファイル名は，ソース・ファイル名の拡張子を .obj に置き換えたものとなります。

  - 上記以外の場合

    出力ロード・モジュール・ファイル名は，a.lmf となります。

[詳細説明]

- 出力ファイル名を *file* に指定します。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- -P ／ -S ／ -c オプションと同時に指定することにより処理を中断した場合にも，本オプションは有効となります。
  - -P オプションと同時に指定した場合

    file には，入力ファイルに対してプリプロセス処理を行った結果のファイル名を指定したものとみなします。

  - -S オプションと同時に指定した場合

    file には，アセンブラ・ソース・ファイル名を指定したものとみなします。

  - -c オプションと同時に指定した場合

    file には，オブジェクト・モジュール・ファイル名を指定したものとみなします。

  - 上記以外の場合

    file には，ロード・モジュール・ファイル名を指定したものとみなします。

- 出力ファイルが複数の場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［個別コンパイル・オプション］タブの［出力ファイル］カテゴリの［オブジェクト・モジュール・ファイル名］

［使用例］

- ロード・モジュール・ファイルをファイル名 sample.lmf で出力します。

```
>cx -CF3746 -osample.lmf main.c
```

-Xobj_path

コンパイル途中に生成されるオブジェクト・モジュール・ファイルの保存先を指定します。

[指定形式]

```
-Xobj_path[=path]
```

- 省略時解釈

  カレント・フォルダに，ソース・ファイル名の拡張子を .obj で置き換えたファイル名でオブジェクト・モジュール・ファイルを保存します。

  ただし，入力として，1 つのソース・ファイルを指定し，かつ -c オプションを指定しない場合は，オブジェクト・モジュール・ファイルは保存しません。

[詳細説明]

- コンパイル途中に生成されるオブジェクト・モジュール・ファイルを *path* に保存します。
  - *path* に存在するファイル名を指定した場合

    出力するオブジェクト・モジュール・ファイルが 1 つの場合は，*path* という名前で保存します。

    出力するオブジェクト・モジュール・ファイルが複数の場合は，エラーとなります。
  - *path* に存在するフォルダ名を指定した場合

    path に，ソース・ファイル名の拡張子を .obj で置き換えたファイル名でオブジェクト・モジュール・ファイルを保存します。
  - *path* に指定した名前のフォルダ，およびファイルが存在しない場合

    エラーとなります。
  - =*path* を省略した場合

    カレント・フォルダに，ソース・ファイル名の拡張子を .obj で置き換えたファイル名でオブジェクト・モジュール・ファイルを保存します。
- ソース・ファイルとして同じ名前のファイル（異なるフォルダにある場合を含む）を複数指定した場合は，警告を出力して，最後に指定したソース・ファイルに対するオブジェクト・モジュール・ファイルのみを保存します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［出力ファイルの種類と場所］カテゴリの［中間ファイル出力フォルダ］

[使用例]

- コンパイル途中に生成されるオブジェクト・モジュール・ファイルをファイル名 sample.obj で保存します。

```
>cx -CF3746 -Xobj_path=sample.obj main.c
```

-Xasm_path

コンパイル途中に生成されるアセンブラ・ソース・ファイルの保存先を指定します。

[指定形式]

```
-Xasm_path[=path]
```

- 省略時解釈

  アセンブラ・ソース・ファイルを出力しません（-S オプション指定時を除く）。

[詳細説明]

- コンパイル途中に生成されるアセンブラ・ソース・ファイルを *path* に保存します。
  - *path* に存在するファイル名を指定した場合

    出力するアセンブラ・ソース・ファイルが 1 つの場合は，*path* という名前で保存します。

    出力するアセンブラ・ソース・ファイルが複数の場合は，エラーとなります。
  - *path* に存在するフォルダ名を指定した場合

    path に，C ソース・ファイル名の拡張子を .asm で置き換えたファイル名でアセンブラ・ソース・ファイルを保存します。
  - *path* に指定した名前のフォルダ，およびファイルが存在しない場合

    エラーとなります。
  - =*path* を省略した場合

    カレント・フォルダに，C ソース・ファイル名の拡張子を .asm で置き換えたファイル名でアセンブラ・ソース・ファイルを保存します。
- ソース・ファイルとして同じ名前のファイル（異なるフォルダにある場合を含む）を複数指定した場合は，警告を出力して，最後に指定したソース・ファイルに対するアセンブラ・ソース・ファイルのみを保存します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力ファイル］カテゴリの［アセンブラ・ソース・ファイルを出力する］，［アセンブラ・ソース・ファイル出力フォルダ］
  - ［個別コンパイル・オプション］タブの［出力ファイル］カテゴリの［アセンブラ・ソース・ファイルを出力する］，［アセンブラ・ソース・ファイル出力フォルダ］

[使用例]

- コンパイル途中に生成されるアセンブラ・ソース・ファイルをファイル名 sample.asm で保存します。

```
>cx -CF3746 -Xasm_path=sample.asm main.c
```

-Xprn_path

アセンブル・リスト・ファイルの保存先を指定します。

[指定形式]

```
-Xprn_path[=path]
```

- 省略時解釈

  アセンブル・リスト・ファイルを出力しません。

[詳細説明]

- アセンブル時にアセンブル・リスト・ファイルを出力して，*path* に保存します。
  - *path* に存在するファイル名を指定した場合

    出力するアセンブル・リスト・ファイルが 1 つの場合は，*path* という名前で保存します。

    出力するアセンブル・リスト・ファイルが複数の場合は，エラーとなります。
  - *path* に存在するフォルダ名を指定した場合

    path に，ソース・ファイル名の拡張子を .prn で置き換えたファイル名でアセンブル・リスト・ファイルを保存します。
  - *path* に指定した名前のフォルダ，およびファイルが存在しない場合

    エラーとなります。
  - =*path* を省略した場合

    カレント・フォルダに，ソース・ファイル名の拡張子を .prn で置き換えたファイル名でアセンブル・リスト・ファイルを保存します。
- ソース・ファイルとして同じ名前のファイル（異なるフォルダにある場合を含む）を複数指定した場合は，警告を出力して，最後に指定したソース・ファイルに対するアセンブル・リスト・ファイルのみを保存します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイルを出力する］，［アセンブル・リスト・ファイル出力フォルダ］
  - ［個別コンパイル・オプション］タブの［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイルを出力する］，［アセンブル・リスト・ファイル出力フォルダ］

[使用例]

- アセンブル時に出力されるアセンブル・リスト・ファイルをファイル名 sample.prn で保存します。

```
>cx -CF3746 -Xprn_path=sample.prn main.c
```

-Xtemp_path

作業用フォルダを指定します。

[指定形式]

```
-Xtemp_path=path
```

- 省略時解釈

  以下の順番で作業用フォルダを決定します。

（1）環境変数 TEMP で指定したフォルダ

（2）環境変数 TMP で指定したフォルダ

（3）カレント・フォルダ

[詳細説明]

- 内部的に用いるテンポラリ・ファイルを生成する作業用フォルダを *path* に指定します。
- *path* が存在しない場合は，警告を出力して，以下の順番で作業用フォルダを決定します。

（1）環境変数 TEMP で指定したフォルダ

（2）環境変数 TMP で指定したフォルダ

（3）カレント・フォルダ

- *path* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［その他］カテゴリの［一時作業フォルダ］

[使用例]

- 作業用フォルダをフォルダ D:￥tmp に指定します。

```
>cx -CF3746 -Xtemp_path=D:￥tmp main.c
```

-Xlink_output

ROM 化処理前のロード・モジュール・ファイルを保存します。

[指定形式]

```
-Xlink_output=file
```

- 省略時解釈

  ROM 化処理前のロード・モジュール・ファイルを保存しません。

[詳細説明]

- ROM 化処理前のロード・モジュール・ファイルをファイル名 *file* で保存します。
- *file* を省略した場合は，エラーとなります。
- -Xno_romize オプションと同時に指定した場合は，警告を出力して，本オプションを無視します。

[使用例]

- ROM 化処理前のロード・モジュール・ファイルをファイル名 sample.lmf で保存します。

```
>cx -CF3746 -Xlink_output=sample.lmf main.c
```

ソース・デバッグ制御

ソース・デバッグ制御オプションには，次のものがあります。

- -g
- -Xkeep_access_size

-g

ソース・デバッグ用の情報を出力します。

[指定形式]

```
-g
```

- 省略時解釈

  ソース・デバッグ用の情報を出力しません。

[詳細説明]

- ソース・デバッグ用の情報を出力ファイル中に出力します。
- 本オプションを指定することにより，ソース・デバッグが可能となります。
- 最適化オプションと同時に指定した場合は，デバッグのしやすさに影響があります。

  詳細については，「(2) 最適化によるデバッグへの影響」を参照してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［デバッグ情報］カテゴリの［デバッグ情報を生成する］
  - ［個別コンパイル・オプション］タブの［デバッグ情報］カテゴリの［デバッグ情報を生成する］

[使用例]

- ソース・デバッグ用の情報を出力ファイル中に出力します。

```
>cx -CF3746 -g main.c
```

-Xkeep_access_size

メモリ・アクセス・サイズの変更を禁止します。

[指定形式]

```
-Xkeep_access_size
```

- 省略時解釈

  ロード／ストア命令の代わりに 1 ビット操作命令（set1，clr1，tst1，not1）を生成することがあります。

[詳細説明]

- ロード／ストア命令（byte アクセスは除く）の 1 ビット操作命令（set1，clr1，tst1，not1）への置き換えを禁止します。
- 本オプションは，デバッグ時に変数の read，write イベントを設定する場合に有用です。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［メモリ・アクセス・サイズの変更を抑止する］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［メモリ・アクセス・サイズの変更を抑止する］

[使用例]

- ロード，ストア命令の 1 ビット操作命令への置き換えを禁止します。

```
>cx -CF3746 -Xkeep_access_size main.c
```

デバイス指定

デバイス指定オプションには，次のものがあります。

- -C
- -Xcommon
- -Xdev_path
- -Xprogrammable_io

-C

ターゲット・デバイスを指定します。

[指定形式]

```
-Cdevice
```

- 省略時解釈

  -Xcommon オプションを指定している場合は，その指定内容に従います。

  それ以外の場合は，エラーとなります（-V ／ -h ／ -P オプション指定時を除く）。

  なお，リンクを行う場合は，エラーとなります。

[詳細説明]

- ターゲット・デバイスを指定します。
- *device* に指定可能なデバイス品種については，各デバイス・ファイルのユーザーズ・マニュアルを参照してください。
- *device* が存在しない（対応するデバイス・ファイルがない）場合は，エラーとなります。
- *device* を省略した場合は，エラーとなります。
- リンクを行う場合は，本オプションを省略することはできません。

[使用例]

- ターゲット・デバイスとして，μPD70F3746 を指定します。

```
>cx -CF3746 main.c
```

-Xcommon

デバイス共通のオブジェクト・モジュール・ファイルを生成することを指定します。

[指定形式]

```
-Xcommon=series
```

- 省略時解釈

  -C オプションを指定している場合は，その指定内容に従います。

  それ以外の場合は，エラーとなります。

[詳細説明]

- デバイス共通のオブジェクト・モジュール・ファイルを生成することを指定します。
- 本オプションを指定した場合，ターゲットの命令セット・アーキテクチャの命令のみを使用し，また，命令セット・アーキテクチャに対応した共通のマジック・ナンバ *series* をオブジェクト・モジュール・ファイルに埋め込みます。
- *series* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

v850e	命令セット・アーキテクチャが V850E より上位（V850E，V850E2，V850E2V3）の品種をターゲット・デバイスとして指定したものとのリンクが可能です。
v850e2	命令セット・アーキテクチャが V850E2 より上位（V850E2，および V850E2V3）の品種をターゲット・デバイスとして指定したものとのリンクが可能です。
v850e2v3	命令セット・アーキテクチャが V850E2V3 である品種をターゲット・デバイスとして指定したものとのリンクが可能です。 ターゲット・デバイスの命令セット・アーキテクチャが V850E2V3 である場合は，その性能を引き出すために，本項目を指定することを推奨します。

- *series* を省略した場合は，エラーとなります。
- -C オプションと同時に指定した場合の処理は，以下のようになります。

-Xcommon の *series* パラメータ	-C の指定デバイス		
	V850E	V850E2	V850E2V3
v850e	通常処理	-Xcommon=v850e2 に置換（警告を出力）	-Xcommon=v850e2v3 に置換（警告を出力）
v850e2	通常処理（警告出力）	通常処理	-Xcommon=v850e2v3 に置換（警告を出力）
v850e2v3	通常処理（警告出力）	通常処理（警告出力）	通常処理

-C オプションで指定したデバイスの命令セット・アーキテクチャが本オプションで指定したものである場合，両方のオプションを処理します。
-C オプションで指定したデバイスの命令セット・アーキテクチャが本オプションで指定したものより下位（V850E2V3 ＞ V850E2 ＞ V850E）の場合，警告を出力して，両方のオプションを処理します。
-C オプションで指定したデバイスの命令セット・アーキテクチャが本オプションで指定したものより上位の場合，警告を出力して，本オプションの *series* パラメータを -C オプションで指定したデバイスの命令セット・アーキテクチャに置換して処理します。

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［出力ファイルの種類と場所］カテゴリの［デバイス共通オブジェクト・モジュールを出力する］

［使用例］

- 生成するオブジェクト・モジュール・ファイルに，命令セット・アーキテクチャが V850E より上位の品種に共通のマジック・ナンバを埋め込みます。

```
>cx -Xcommon=v850e -c main.c
```

-Xdev_path

デバイス・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Xdev_path=path
```

- 省略時解釈

  デバイス・ファイルを標準のデバイス・ファイル・フォルダから検索します。

[詳細説明]

- デバイス・ファイルを *path* で指定したフォルダから検索します。
- *path* で指定したフォルダが存在しない場合，または -C オプションで指定したデバイス・ファイルが *path* で指定したフォルダに見つからない場合は，警告を出力して，標準のデバイス・ファイル・フォルダ[注]を検索します。それでも見つからない場合は，エラーとなります。

  注　以下の順に検索します。

  【V850E2V3】

  1. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850E2 ¥ Devicefile
  2. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850 ¥ Devicefile
  3. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device_Custom ¥ Devicefile

  【V850E/V850E2】

  1. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850 ¥ Devicefile
  2. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device_Custom ¥ Devicefile

- *path* を省略した場合は，エラーとなります。

[使用例]

- デバイス・ファイルをフォルダ D:¥dev から検索します。

```
>cx -CF3746 -Xdev_path=D:¥dev main.c
```

-Xprogrammable_io

プログラマブル周辺 I/O レジスタの開始アドレスを指定します。

[指定形式]

```
-Xprogrammable_io=num
```

- 省略時解釈

  プログラマブル周辺 I/O レジスタの開始アドレスは，各デバイスで既定されているアドレスとなります。

[詳細説明]

- アドレス変更可能なプログラマブル周辺 I/O レジスタの開始アドレス *num* を指定します。
- ターゲット・デバイスがアドレス変更可能なプログラマブル周辺 I/O レジスタを持つ場合（V850E/IA1 など），コンパイル（アセンブル）時に，その開始アドレスを確定させる必要があります。

  そこで，本オプションにより，プログラマブル周辺 I/O レジスタの開始アドレスを指定します。
- プログラマブル周辺 I/O レジスタの開始アドレスの下位ビット（ビット数はデバイスにより異なります）は固定であるため，*num* は下位ビットが 0 となるように指定します。

  下位ビットに 0 以外を指定した場合は，警告を出力して，下位ビットを切り捨てます。
- *num* に不正な値，または各デバイスで指定可能な範囲を越える値を指定した場合は，警告を出力して，本オプションを無視します。
- *num* に指定する値は，同一アプリケーションのすべてのファイルで同じにする必要があります。

  ただし，プログラマブル周辺 I/O レジスタを使用しないファイルを個別にコンパイルする場合は，本オプションを指定する必要はありません。

  また，プログラマブル周辺 I/O レジスタ機能を持たないデバイスの場合，および V850E/V850E2/V850E2V3 共通としてアセンブルする場合に本オプションを指定すると，警告を出力して，本オプションを無視します。
- *num* を省略した場合は，エラーとなります。
- プログラマブル周辺 I/O レジスタを使用するためには，BPC レジスタに値を指定する必要がありますが，本オプションはプログラマブル周辺 I/O レジスタのアドレスをコンパイル（アセンブル）時に確定するためのものであり，BPC レジスタに実際に値を反映させるものではありません。

  動作のためには，別途，スタートアップ・ルーチンなどで BPC レジスタに値を指定する必要があります。

  例えば，V850E/IA1 の場合は，以下の例のように指定コードを追加してください。

```
USEBPC  set 0x8000                          ; プログラマブル周辺 I/O レジスタのアクセス許可（固定）
PIOADDR set 0x38d000                        ; -Xprogrammable_io オプションで渡したのと同じ値
mov     (USEBPC or (PIOADDR shr 14)), r13   ; PIOADDR（=num）は自動的に渡されます
st.h    r13, BPC
```

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［デバイス］カテゴリの［プログラマブル I/O 領域開始アドレス］

［使用例］

- ターゲット・デバイスが V850E/IA1 の場合，プログラマブル周辺 I/O レジスタの開始アドレスを 0x38d0000 に指定します。

```
>cx -CF3746 -Xprogrammable_io=0x38d0000 main.c
```

処理中断指定

処理中断指定オプションには，次のものがあります。

- -P
- -S
- -c

-P

入力ファイルに対してプリプロセス処理のみ実行します。

［指定形式］

```
-P
```

- 省略時解釈

  プリプロセス処理以降も処理を継続します。

  ファイルは出力しません。

［詳細説明］

- 入力ファイルに対してプリプロセス処理のみ実行して，結果をファイルに出力します。
- 入力ファイルを複数指定した場合は，エラーとなります。
- 出力ファイル名は，入力ファイル名の拡張子を .i に置き換えたものになります。
- -o オプションと同時に指定することにより，出力ファイル名を指定することができます。
- -Xpreprocess オプションを指定することにより，出力ファイルの内容を制御することができます。
- -Xchar オプションと同時に指定した場合は，-Xchar オプションが無効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力ファイル］カテゴリの［プリプロセス処理したソースを出力する］
  - ［個別コンパイル・オプション］タブの［出力ファイル］カテゴリの［プリプロセス処理したソースを出力する］

［使用例］

- 入力ファイルに対してプリプロセス処理のみ実行して，結果をファイル main.i に出力します。

```
>cx -CF3746 -P main.c
```

-S

アセンブル以降の処理を実行しません。

[指定形式]

```
-S
```

- 省略時解釈

  アセンブル以降も処理を継続します。

[詳細説明]

- アセンブル以降の処理を実行しません。
- ソース・ファイル名の拡張子を .asm で置き換えた名前でアセンブラ・ソース・ファイルを出力します。
- -o オプションと同時に指定することにより，出力ファイル名を指定することができます。

[使用例]

- アセンブル以降の処理を実行せず，アセンブラ・ソース・ファイル main.asm を出力します。

```
>cx -CF3746 -S main.c
```

-c

リンク以降の処理を実行しません。

[指定形式]

```
-c
```

- 省略時解釈

  リンク以降も処理を継続します。

[詳細説明]

- リンク以降の処理を実行しません。
- ソース・ファイル名の拡張子を .obj で置き換えた名前でオブジェクト・モジュール・ファイルを出力します。
- -o オプションと同時に指定することにより，出力ファイル名を指定することができます。

[使用例]

- リンク以降の処理を実行せず，オブジェクト・モジュール・ファイル main.obj を出力します。

```
>cx -CF3746 -c main.c
```

プリプロセッサ制御

プリプロセッサ制御オプションには，次のものがあります。

- -D
- -U
- -I
- -Xpreprocess

-D

プリプロセッサ・マクロを定義します。

[指定形式]

```
-Dname[=def][name[=def]]...
```

- 省略時解釈

  なし

[詳細説明]

- プリプロセッサ・マクロとして *name* を定義します。
- C ソース・プログラムの前に，#define *name def* を記述するのと同等です。
- *name* が，アセンブラ・シンボルには使用可能であるがプリプロセッサ・マクロには使用できない文字（@，.，~）を含む場合，警告を出力してアセンブラ・シンボルとしてのみ定義します。
- *name* を省略した場合は，エラーとなります。
- =*def* を省略した場合，*def* は 1 とみなします。
- 本オプションは，複数指定が可能です。
- 同じプリプロセッサ・マクロに対して，本オプションと -U オプションを同時に指定した場合は，あとから指定したものが有効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［定義マクロ］
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］
  - ［個別コンパイル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］

[使用例]

- プリプロセッサ・マクロとして sample=256 を定義します。

```
>cx -CF3746 -Dsample=256 main.c
```

-U

-D オプションによるプリプロセッサ・マクロの定義を解除します。

[指定形式]

```
-Uname[,name]...
```

- 省略時解釈

  なし

[詳細説明]

- -D オプションによるプリプロセッサ・マクロ *name* の定義を解除します。
- C ソース・プログラムの前に，#undef *name* を記述するのと同等です。
- *name* を省略した場合は，エラーとなります。
- 本オプションでは，#define *name def* の記述による定義は解除できません。
- 本オプションにより，C 言語の既定義マクロを解除することもできますが，__LINE__，__FILE__，__DATE__，__TIME__，__CX__ を解除することはできません。

  name にこれらを指定した場合は，エラーとなります。
- 本オプションは，複数指定が可能です。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［定義解除マクロ］
  - ［個別コンパイル・オプション］タブの［プリプロセス］カテゴリの［定義解除マクロ］

[使用例]

- -D オプションによるプリプロセッサ・マクロ test の定義を解除します。

```
>cx -CF3746 -Utest main.c
```

-I

インクルード・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Ipath[,path]...
```

- 省略時解釈

  インクルード・ファイルを 標準インクルード・ファイル・フォルダからのみ検索します。

[詳細説明]

- プリプロセッサ指令 #include で読み込むインクルード・ファイルを検索するフォルダを *path* に指定します。

  インクルード・ファイルの検索は，以下の順番で行います。

(1) ソース・ファイルのあるフォルダ（ファイルを " " で指定した場合）

(2) -I オプションで指定したフォルダ

(3) 標準インクルード・ファイル・フォルダ[注]

注 *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ inc

- *path* が存在しない場合は，警告を出力します。
- *path* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［追加のインクルード・パス］，［システム・インクルード・パス］
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］，［システム・インクルード・パス］
  - ［個別コンパイル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］，［ビルド・ツールに指定した全体インクルード・パスも使用する］

[使用例]

- インクルード・ファイルをカレント・フォルダ，フォルダ D:¥ include，標準フォルダの順で検索します。

```
>cx -CF3746 -ID:¥include main.c
```

-Xpreprocess

プリプロセス結果の出力を制御します。

[指定形式]

```
-Xpreprocess=string[,string]
```

- 省略時解釈

  プリプロセス結果のファイルに，C ソースのコメント，および行番号情報を出力しません。

[詳細説明]

- プリプロセス結果のファイルに，C ソースのコメント，および行番号情報を出力します。
- 本オプションは，-P オプション指定時のみ有効です。

  -P オプションを指定しない場合は，本オプションを無視します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

comment	C ソースのコメントを出力します。
line	行番号情報[注]を出力します。

注　行番号情報のフォーマットを以下に示します。

```
#line 行番号 " ファイル名 "
```

  - *行番号*は 10 進数で，最大値は unsigned int の最大数となります。
  - *ファイル名*はフルパスで，パス中の \\ は \ に，" は \" に変換します。

    印字可能文字（スペースを含む）でない場合は，*\3 桁の 8 進数* ("\\%03o") で出力します。

    改行文字は \\n に変換します。
  - 入力ソース・ファイル中に前処理指令 # *数値* " *文字列* "，または #line *数値* " *文字列* " を記述している場合は，*数値を行番号*，*文字列を ファイル名*として適用します。

- *string* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［プリプロセス］カテゴリの［プリプロセス結果に C ソース・コメントを出力する］，［プリプロセス結果に行番号情報を出力する］
  - ［個別コンパイル・オプション］タブの［プリプロセス］カテゴリの［プリプロセス結果に C ソース・コメントを出力する］，［プリプロセス結果に行番号情報を出力する］

［使用例］

- プリプロセス結果のファイルに，C ソースのコメント，および行番号情報を出力します。

```
>cx -CF3746 -Xpreprocess=comment,line -P main.c
```

以下は上記の例と同等です。

```
>cx -CF3746 -Xpreprocess=comment -Xpreprocess=line -P main.c
```

C 言語制御

C 言語制御オプションには，次のものがあります。

- -Xansi
- -Xchar
- -Xenum_type
- -Xdef_var

-Xansi

C ソース・プログラムを ANSI 規格に厳密にあわせて処理します。

[指定形式]

```
-Xansi
```

- 省略時解釈

  従来の C 言語の仕様との両立性を持たせ，警告を出力して処理を続行します。

[詳細説明]

- C ソース・プログラムを ANSI 規格[注]に厳密にあわせて処理し，規格に反する記述に対してエラーや警告を出力します。
- 本オプション指定時は，マクロ名 __STDC__ を，値が 1 のマクロとして定義します。
- 言語仕様に厳密なコンパイル時の処理は，以下のようになります。
  - ビット・フィールド

    ビット・フィールドに int 型以外の型を指定した場合は，エラーとなります。

    本オプションを指定しない場合は，警告を出力して，int 型以外の型の指定を許可します。
  - # 行番号

    エラーとなります。

    本オプションを指定しない場合は，“#line 行番号”と同様に扱います。
  - #pragma inline された関数の引数

    指定した関数の呼び出しと定義の間で，戻り値の型や引数の型が異なるが型変換が可能である場合は，エラーとなります。

    本オプションを指定しない場合は，戻り値の型は呼び出し側の型に，引数は関数定義での型に変換して，インライン展開を行います。
  - 2 進定数

    エラーとなります。

    本オプションを指定しない場合は，“0b”，または“0B”と，その後ろに続く 1 個以上の“0”，または“1”の数字の並びを 2 進定数とします。

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［C 言語］カテゴリの［ANSI 規格に厳密に合わせてコンパイルする］
  - ［個別コンパイル・オプション］タブの［C 言語］カテゴリの［ANSI 規格に厳密に合わせてコンパイルする］

注　ISO/IE C9899:1990（C90）で規程されている規格のことです。
cx は，ISO/IE C9899:1999（C99）で追加された仕様の一部も受容しますが，本オプションを指定した場合は，規格に反する記述はエラーとなります。

［使用例］

- C ソース・プログラムを ANSI 規格に厳密にあわせて処理し，規格に反する記述に対してエラーや警告を出力します。

```
>cx -CF3746 -Xansi main.c
```

-Xchar

char 型に対して，符号を指定します。

[指定形式]

```
-Xchar=string
```

- 省略時解釈

  符号が付かない単なる char 型に対して，符号付きとして扱います。

[詳細説明]

- 符号が付かない単なる char 型に対して，符号付きとするか符号なしとするかを指定します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

signed	符号付きとして扱います。
unsigned	符号なしとして扱います。

- *string* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［C 言語］カテゴリの［char の符号］

[使用例]

- 符号が付かない単なる char 型に対して，符号付きとして扱います。

```
>cx -CF3746 -Xchar=signed main.c
```

-Xenum_type

列挙型に対して，どの整数型として扱うかを指定します。

［指定形式］

```
-Xenum_type=string
```

- 省略時解釈

  列挙型に対して，signed int として扱います。

［詳細説明］

- 列挙型に対して，どの整数型として扱うかを指定します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

auto	各列挙型について，その型のすべての列挙子の値を表現可能な最小の整数型として扱います。

- *string* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［C 言語］カテゴリの［enum の型］

［使用例］

- 各列挙型について，その型のすべての列挙子の値を表現可能な最小の整数型として扱います。

```
>cx -CF3746 -Xenum_type=auto main.c
```

-Xdef_var

変数の仮定義を定義として扱います。

[指定形式]

```
-Xdef_var
```

- 省略時解釈

  変数の仮定義を定義として扱いません。

[詳細説明]

- 変数の仮定義を定義として扱います。
- 複数のファイルで同名の仮定義が存在した場合は，リンク時に１つに結合されずに多重定義エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［C 言語］カテゴリの［仮定義を定義に変更する］

[使用例]

- 変数の仮定義を定義として扱います。

```
>cx -CF3746 -Xdef_var main.c
```

変数／関数情報指定

変数／関数情報指定オプションには，次のものがあります。

- -Xsymbol_file

-Xsymbol_file

シンボル情報ファイルを利用します。

[指定形式]

```
-Xsymbol_file=file
```

- 省略時解釈

  シンボル情報ファイルを読み込みません。

[詳細説明]

- 変数の配置セクション指定，および関数の参照情報を記述したシンボル情報ファイル *file* を読み込み，コンパイル時にその情報を利用します。
- 詳細については，「B. 1. 4　シンボル情報ファイル」を参照してください。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［シンボル情報］カテゴリの［使用するシンボル情報ファイル］

[使用例]

- シンボル情報ファイル symbol を読み込み，コンパイル時にその情報を利用します。

```
>cx -CF3746 -Xsymbol_file=symbol.sfg main.c
```

日本語／中国語文字列制御

日本語／中国語文字列制御オプションには，次のものがあります。

- -Xcharacter_set

-Xcharacter_set

日本語／中国語の文字コードを指定します。

[指定形式]

```
-Xcharacter_set=code
```

- 省略時解釈

  日本語の文字コードを SJIS として扱います。

[詳細説明]

- ソース・ファイル中の日本語／中国語のコメント，文字列に対して，使用する文字コードを指定します。
- *code* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

  なお，ソース・ファイル中で使用している文字コードと異なるものを指定した場合，動作は保証されません。

none	日本語／中国語の文字コードを処理しません
euc_jp	EUC（日本語）
sjis	SJIS
utf8	UTF-8
big5	繁体字中国語
gb2312	簡体字中国語

- *code* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［文字コード］カテゴリの［文字コード］
  - ［個別コンパイル・オプション］タブの［文字コード］カテゴリの［文字コード］

[使用例]

- ソース・ファイル中の日本語のコメント，文字列に対して，使用する文字コードに EUC を指定します。

```
>cx -CF3746 -Xcharacter_set=euc_jp main.c
```

最適化指定

最適化指定オプションには，次のものがあります。

- -O
- -Xdelete_func
- -Xsort_var
- -Xinline_strcpy
- -Xpro_epi_runtime
- -Xcall_lib
- -Xmerge_string

-O

最適化のレベル，または各最適化項目の詳細を指定します。

[指定形式]

```
-O[level]
-O[item[=value][,item[=value]]...
```

- 省略時解釈

  デバッグに影響しない最適化のみを行います（-Odefault オプションの指定と同じです）。

[詳細説明]

- 最適化のレベル，または各最適化項目の詳細を指定します。

  詳細については，「B. 1. 5　最適化機能」を参照してください。

- *level* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

nothing	デバッグ優先の最適化 デバッグのしやすさを重視し，デフォルトで実行する最適化を含むすべての最適化を抑止します。
default	デフォルト デバッグに影響しない範囲の最適化（式の最適化，およびレジスタ割り付けなど）を行います。
size	オブジェクト・サイズ優先の最適化 ROM/RAM 容量の削減を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
speed	実行速度優先の最適化 実行速度の短縮を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。

- *level* を省略した場合は，size を指定したものとみなします。

- *item*，および *value* に指定可能なものを以下に示します。

これ以外のものを指定した場合は，エラーとなります。

最適化項目 （*item*）	パラメータ （*value*）	説明
unroll	0 ～ 4294967295 （整数値）	ループ展開 0，または 1 を指定した場合は，展開を抑止します。 2 以上を指定した場合は，実行回数が *N* 回（*N* は定数）のループと，*value* 回展開されたコードを含むループに変換します。 ただし，展開後のコード・サイズが大きいか，またはループの実行回数が少ない場合は，展開数が少なくなったり，展開しない場合があります。 また，内側にループを含むような複雑な構造のループは，展開しない場合があります。 *value* を省略した場合は，4 を指定したものとみなします。 本項目は，-Ospeed オプションを指定した場合は，-Ounroll=4 オプションを指定したものとみなします。
inline	0 ～ 3 （整数値）	関数のインライン展開 *value* は，展開のレベルを表します。 0：#pragma inline 指定した関数を含めて，すべてのインライン展開を抑止します。 1：#pragma inline 指定した関数のみ展開します。 2：自動的に展開対象の関数を判別して展開します。 3：コード・サイズがなるべく増加しない範囲で，自動的に展開対象の関数を判別して展開します。 ただし，1 ～ 3 を指定した場合でも，関数の内容やコンパイル状況により，#pragma inline 指定した関数が展開されない場合があります。 *value* を省略した場合は，2 を指定したものとみなします。 本項目は，-Osize，または -Ospeed オプションを指定した場合に有効です（-Osize オプションを指定した場合は -Oinline=3，-Ospeed オプションを指定した場合は -Oinline=2 オプションを指定したものとみなします）。 -Osize，-Ospeed，-Oinline オプションのいずれも指定していない場合は，-Oinline=1 オプション指定したものとみなします。 -Onothing オプションを指定した場合は，-Oinline=0 オプションを指定したものとみなします。
delete_static_func	on，または off	未使用 static 関数の削除 *value* を省略した場合は，on を指定したものとみなします。 本項目は，-Onothing オプション指定時以外に有効です。
pipeline 【V850E2V3】	on，または off	パイプライン最適化 *value* を省略した場合は，on を指定したものとみなします。 本項目は，-Ospeed オプション指定時は，常に有効です。

最適化項目 （*item*）	パラメータ （*value*）	説明
subroutine	0，または 1	共通コードのサブルーチン化 0：サブルーチン化を行いません。 1：サブルーチン化を行います。 *value* を省略した場合は，1 を指定したものとみなします。 本項目は，どの最適化レベルでも自動的に有効になることはないため，有効にしたい場合は -Osubroutine=1 オプションを別途指定する必要があります。
tail_call	on，または off	関数末尾の関数呼び出しの jr 化 on を指定した場合，関数の末尾が関数呼び出しであり，かつ一定の条件を満たす場合に，その呼び出しに対して jarl 命令ではなく jr 命令を生成して lp の退避／復帰コードを削除し，コード・サイズを削減します。 ただし，一部のデバッグ機能を使用することができなくなります。 *value* を省略した場合は，on を指定したものとみなします。 本項目は，どの最適化レベルでも自動的に有効になることはないため，有効にしたい場合は -Otail_call=on オプションを別途指定する必要があります。

- *item* を省略した場合，本オプションは -Osize を指定したものとみなします。
- 同じ *item* について，本オプションを複数指定した場合は，あとから指定したものが有効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［最適化レベル］
  - ［コンパイル・オプション］タブの［最適化］カテゴリの［最適化レベル］
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［ループ展開最大数］，［未使用 static 関数の削除を行う］，［関数のインライン展開を行う］，［パイプライン最適化を行う］，［共通コードのサブルーチン化を行う］，［関数末尾の関数呼び出しに jr 命令を使用する］
  - ［個別コンパイル・オプション］タブの［最適化］カテゴリの［最適化レベル］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［ループ展開最大数］，［未使用 static 関数の削除を行う］，［関数のインライン展開を行う］，［パイプライン最適化を行う］，［共通コードのサブルーチン化を行う］，［関数末尾の関数呼び出しに jr 命令を使用する］

［使用例］

- オブジェクト・サイズ優先の最適化を行います。

```
>cx -CF3746 -Osize main.c
```

-Xdelete_func

未使用関数を削除します。

[指定形式]

```
-Xdelete_func[=name[,name]...]
```

- 省略時解釈

  未使用関数を削除しません。

[詳細説明]

- 未使用関数を削除します。
- *name* には，C 言語で記述した関数名を指定します。
- パラメータを省略した場合は，main を指定したものとみなします。
- *name* が存在しない場合は，パラメータを省略した場合と同様の処理を行います。
- static 関数を指定することはできません。
- 関数 *name* をエントリ関数として，そこから呼び出される関数を追跡して，呼び出されない関数を削除します。

  ただし，割り込み関数とリアルタイム OS 用のタスクは，削除対象から除外します。

  また，この処理は，C ソース・ファイルの解析に基づいて実行するため，アセンブラ・ソース・ファイルからのみ呼び出される関数は，不要な関数として削除します。

  アセンブラ・ソース・ファイルで定義している，その他の関数は削除しません。
- 本オプションは，インライン展開後に不要になった関数を削除するのに有効です。

  しかし，関数の呼び出しの有無をその関数の定義を含む C ソース・ファイル内のみで判断するため，そのファイル内で呼び出しがなければ，ほかの C ソース・ファイルから呼び出しがあっても削除してしまいます。

  -Xsymbol_file オプションを同時に指定することにより，ほかの C ソース・ファイルから呼び出されている関数の削除を回避することができます。

[使用例]

- 関数 func をエントリ関数として，未使用関数を削除します。

```
>cx -CF3746 -Xdelete_func=func main.c sub.c
```

-Xsort_var

外部変数をソートします。

[指定形式]

```
-Xsort_var
```

- 省略時解釈

  外部変数をソートしません。

[詳細説明]

- const，sconst セクション以外のセクションに配置されている外部変数をアライメントの大きい順に並び替えます。
- 本オプションは，RAM 容量を削減するためのものです。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［外部変数のソートを行う］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［外部変数のソートを行う］

[使用例]

- const，sconst セクション以外のセクションに配置されている外部変数をアライメントの大きい順に並び替えます。

```
>cx -CF3746 -Xsort_var main.c
```

-Xinline_strcpy

標準ライブラリ関数 strcpy，strcmp，memcpy，memset の呼び出しをインライン展開します。

[指定形式]

```
-Xinline_strcpy
```

- 省略時解釈

  標準ライブラリ関数 strcpy，strcmp，memcpy，memset のインライン展開を行いません。

[詳細説明]

- 標準ライブラリ関数 strcpy，strcmp，memcpy，memset の呼び出しをインライン展開します。
- 本オプションは，-Xpack オプションと同時に指定することはできません。
- strcpy については，第二引数が文字列の場合にのみ，インライン展開を行います。
- 本オプションを指定した場合，配列，および文字列は自動的に 4 バイト境界に配置されます。
- 生成されるプログラムの実行速度は高速になりますが，コード・サイズは増大します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［strcpy ／ strcmp ／ memcpy ／ memset の展開を行う］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［strcpy ／ strcmp ／ memcpy ／ memset の展開を行う］

[使用例]

- 標準ライブラリ関数 strcpy，strcmp，memcpy，memset の呼び出しをインライン展開します。

```
>cx -CF3746 -Xinline_strcpy main.c
```

-Xpro_epi_runtime

関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にするかどうかを指定します。

[指定形式]

```
-Xpro_epi_runtime[=on|=off]
```

- 省略時解釈

  関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にします。

  ただし，-Ospeed オプション指定時は，ランタイム・ライブラリ呼び出しによる処理は行いません。

[詳細説明]

- 関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にするかどうかを指定します。
- on を指定した場合，関数のプロローグ／エピローグ処理がランタイム・ライブラリ呼び出しになります。
- =on，または =off を省略した場合は，=on を指定したものとみなします。
- =on，=off 以外を指定した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［プロローグ／エピローグ・ライブラリを使用する］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［プロローグ／エピローグ・ライブラリを使用する］

[使用例]

- 関数のプロローグ／エピローグ処理をランタイム・ライブラリ呼び出しによる処理にします。

```
>cx -CF3746 -Xpro_epi_runtime=on main.c
```

-Xcall_lib

ライブラリ関数のインライン展開を抑止します。

[指定形式]

```
-Xcall_lib
```

- 省略時解釈

以下の条件を同時に満たす場合，ライブラリ関数 memcmp，memcpy，memset の呼び出し箇所において，関数呼び出し命令の代わりに適切な命令列を直接生成します。

- 最適化オプション -O，-Osize，-Ospeed のいずれかが指定されている。
- memcmp，memcpy，memset の呼び出しを含むソース・ファイルで標準ヘッダ・ファイル string.h をインクルードしている，またはこれらの関数のプロトタイプ宣言を含んでいる。
- 以下の条件のいずれかが成立する。
  - memcmp の場合，第 3 引数（サイズ）が 1 であり，かつ戻り値を使用する。
  - memcpy の場合，第 3 引数（サイズ）が 1 であり，かつ戻り値を使用しない。
  - memset の場合，第 3 引数（サイズ）が 1，または 2 であり，かつ戻り値を使用しない。

また，以下の条件を同時に満たす場合，ライブラリ関数 sqrtf の呼び出し箇所において，関数呼び出し命令の代わりに sqrtf.s 命令を生成します。

それ以外の場合には，関数の呼び出し命令を生成します。

- 最適化オプション -O，-Osize，-Ospeed のいずれかが指定されている。
- sqrtf の呼び出しを含むソース・ファイルで標準ヘッダ・ファイル math.h をインクルードしている，または sqrtf のプロトタイプ宣言を含んでいる。
- 以下の条件のいずれかが成立する。
  - -C オプションにより V850E2V3 の FPU を持つデバイスが指定されていて，かつ -Xfloat=soft オプションが指定されていない。
  - -Xcommon=v850e2v3 オプションと -Xfloat=fpu オプションが同時に指定されている。

注意　sqrtf.s 命令を直接生成する場合，ライブラリ関数呼び出しの場合と異なり，例外処理が行われません。そのため，ライブラリ関数と同様に例外処理を行いたい場合は，本オプションを指定する必要があります。具体的には以下の点が異なります。

・引数が負の実数であっても，グローバル変数 errno にマクロ EDOM が設定されません。

・エラー処理関数 matherrf（matherr）が有効になりません。

［詳細説明］

- 該当するライブラリ関数の呼び出し箇所において，常に各ライブラリ関数を呼び出す命令を生成します。
- 本オプションは，［省略時解釈］に示す条件が成立する場合であっても，命令への直接展開を行わずにライブラリ関数を呼び出す命令を生成したい場合に使用します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［ライブラリ関数のインライン展開を行う］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［ライブラリ関数のインライン展開を行う］

［使用例］

- ライブラリ関数 sqrtf の呼び出し箇所において，常に各ライブラリ関数を呼び出す命令を生成します。

```
>cx -CF3746 -Xcall_lib main.c
```

-Xmerge_string

文字列定数をマージします。

[指定形式]

```
-Xmerge_string
```

- 省略時解釈

  ソース・ファイル内で同じ文字列定数が複数存在する場合，それぞれを別々の領域に割り付けます。

[詳細説明]

- ソース・ファイル内で同じ文字列定数が複数存在する場合，これらをまとめて１つの領域に割り付けます。
- #pragma section sconst（const）の指定に依らず，同じ文字列定数を同一領域に割り付けます。

  ただし，異なるセクションを指定した場合，文字列定数の割り付け先のセクションはソース中での出現順序に依存します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［文字列定数のマージを行う］
  - ［個別コンパイル・オプション］タブの［最適化 (詳細)］カテゴリの［文字列定数のマージを行う］

[使用例]

- ソース・ファイル内で同じ文字列定数が複数存在する場合，これらをまとめて１つの領域に割り付けます。

```
>cx -CF3746 -Xmerge_string main.c
```

生成コード制御

生成コード制御オプションには，次のものがあります。

- -Xpack
- -Xpass_source
- -Xswitch
- -Xword_case
- -Xr
- -Xreg_mode
- -Xsdata
- -Xsconst
- -Xfloat【V850E2V3】
- -Xfar_jump
- -Xdiv【V850E2V3】

-Xpack

構造体パッキングを行います。

[指定形式]

```
-Xpack=num
```

- 省略時解釈

  構造体パッキングを行いません。

[詳細説明]

- 構造体パッキングを行います。
- 本オプションを指定した場合，構造体のメンバをその型でアライメントせず，指定した *num* バイトのアライメントに詰めてコードを生成します。
- *num* には，1，2，4，8 のいずれかを指定することができます。

  これ以外のものを指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- 本オプションは，-Xinline_strcpy オプションと同時に指定することはできません。
- C ソース中に #pragma 指令で構造体パッキングを指定している場合に本オプションを指定すると，最初の #pragma 指令が出現するまでは，オプションの指定値をすべての構造体に適用します。

  それ以降は，#pragma 指令の値を適用します。

  ただし，#pragma 指令の出現後でも，指定がデフォルトになった部分（#pragma 指令でパッキング値を指定しない場合）は，オプションの指定値を適用します。

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［構造体パッキングを行う］

［使用例］

- 構造体のメンバを 1 バイトのアライメントに詰めてコードを生成します。

```
>cx -CF3746 -Xpack=1 main.c
```

-Xpass_source

出力するアセンブラ・ソース・ファイル中にＣソース・プログラムをコメントとして出力します。

[指定形式]

```
-Xpass_source
```

- 省略時解釈

  出力するアセンブラ・ソース・ファイル中にＣソース・プログラムをコメントとして出力しません。

[詳細説明]

- 出力するアセンブラ・ソース・ファイル中にＣソース・プログラムをコメントとして出力します。
- 出力するコメントは，あくまで参考であり，厳密にはコードと対応していない場合もあります。
  また，実行文以外の行にはコメントとして出力されないものもあります（型宣言やラベルなど）。
  たとえば，グローバル変数とローカル変数，関数宣言などのコメントの出力位置がずれることがあります。
  また，最適化オプションを指定した場合などにより，コードが削除され，コメントのみが残ることもあります。
- 本オプションは，-S，または -Xasm_path オプションと同時に指定する必要があります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［アセンブラ・ソースにコメントを出力する］
  - ［個別コンパイル・オプション］タブの［出力コード］カテゴリの［アセンブラ・ソースにコメントを出力する］

[使用例]

- 出力するアセンブラ・ソース・ファイル中にＣソース・プログラムをコメントとして出力します。

```
>cx -CF3746 -Xpass_source -S main.c
```

-Xswitch

switch 文のコード出力形式を指定します。

［指定形式］

```
-Xswitch=type
```

- 省略時解釈

  cx が switch 文ごとに最適な出力形式を 自動的に選択します。

［詳細説明］

- switch 文のコード出力形式を指定します。
- *type* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

ifelse	case 文の並びに沿った if-else 文と同じ形式で出力します。 頻度が多い順に case 文を書いているときやラベル数が少ないときに，本項目を指定します。 上から順に比較するので，頻繁に合致する case 文を先に記述すると余計な比較が減り，実行速度向上につながります。
binary	バイナリ・サーチ形式で出力します。 バイナリ・サーチ・アルゴリズムに用いて合致する case 文を探します。 ラベル数が多いときに本項目を選択すると，どの case 文も同じくらいの速さで見つけることができます。
table	テーブル・ジャンプ形式で出力します。 case 文の値を基にインデックス化したテーブルを参照し，switch 文の値により case ラベルを選択して処理を行います。 どの case 文にも同じくらい速く分岐します。 ただし，case 値が連続していないときは無駄な領域ができます。 また，case ラベルの最大値と最小値の差が 8192 を越える場合は，本オプションを無視して，switch 文ごとに最適な出力形式を 自動的に選択します。

- *type* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［switch 文の出力コードの選択］
  - ［個別コンパイル・オプション］タブの［出力コード］カテゴリの［switch 文の出力コードの選択］

［使用例］

- switch 文のコードに対して，バイナリ・サーチ形式で出力します。

```
>cx -CF3746 -Xswitch=binary main.c
```

-Xword_case

4 バイトの分岐テーブルを生成します。

[指定形式]

```
-Xword_case
```

- 省略時解釈

  switch 文の case ラベルに対する分岐テーブルを，1 ラベルあたり 2 バイトで生成します。

[詳細説明]

- switch 文の case ラベルに対する分岐テーブルを，1 ラベルあたり 4 バイトで生成します。
- 本オプションは，switch 文が長いためにコンパイル・エラーとなる場合に指定します。
- 本オプションは，-Xswitch=ifelse，または -Xswitch=binary オプションと同時に指定した場合は無効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［switch テーブルのラベル・サイズ (バイト)］
  - ［個別コンパイル・オプション］タブの［出力コード］カテゴリの［switch テーブルのラベル・サイズ (バイト)］

[使用例]

- switch 文の case ラベルに対する分岐テーブルを，1 ラベルあたり 4 バイトで生成します。

```
>cx -CF3746 -Xword_switch main.c
```

-Xr

外部変数の割り付け先のレジスタを指定します。

[指定形式]

```
-Xrnum=sym
```

- 省略時解釈

  プログラム全体で最適なレジスタ割り付けを行います。

[詳細説明]

- 外部変数 *sym* をレジスタ r*num* に割り付けます。
- *num* には，-Xreg_mode オプションを指定して空けたレジスタを指定します。

  num に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

レジスタ・モード	指定可能なレジスタ（*num*）	
	作業用レジスタ	レジスタ変数用レジスタ
22	15 ～ 19	20 ～ 24
26	17 ～ 19	20 ～ 22
32	なし	なし

- *sym* には，外部変数名（C 言語で記述した変数名）で指定します。

  volatile 変数，アドレス演算子を使用した変数，集成体，配列，内部リンケージを持つ変数，および周辺 I/O レジスタは指定できません。

  これらの変数を指定した場合は，エラーとなります。

  sym が存在しない場合は，無視します。
- *num*，または *sym* を省略した場合は，エラーとなります。
- 本オプションは，8 バイトの外部変数に対しても指定可能ですが，その場合，*num* には偶数のみ指定可能です。

  奇数を指定した場合は，エラーとなります。
- 本オプションを複数指定して，同じレジスタに異なる変数を指定した場合，または異なるレジスタに同じ変数を指定した場合は，警告を出力して，最後に指定したものが有効となります。
- *sym* の宣言時に初期値を設定している場合でも，r*num* は自動的に初期化されません。

  そのため，代入文を用いて明示的に *sym* に値を代入するか，スタートアップ・ルーチンで r*sym* に値を設定する必要があります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［外部変数レジスタ］カテゴリの［r15 レジスタに割り当てる外部変数］，［r16 レジスタに割り当てる外部変数］，［r17 レジスタに割り当てる外部変数］，［r18 レジスタに割り当てる外

部変数]，[r19 レジスタに割り当てる外部変数]，[r20 レジスタに割り当てる外部変数]，[r21 レジスタに割り当てる外部変数]，[r22 レジスタに割り当てる外部変数]，[r23 レジスタに割り当てる外部変数]，[r24 レジスタに割り当てる外部変数]

［使用例］

- 外部変数 arg をレジスタ r18 に割り付けます（22 レジスタ・モード使用時）。

```
>cx -CF3746 -Xreg_mode=22 -Xr18=arg main.c
```

-Xreg_mode

レジスタ・モードを指定します。

[指定形式]

```
-Xreg_mode=mode
```

- 省略時解釈

  32 レジスタ・モードのオブジェクト・モジュール・ファイルを生成します。

[詳細説明]

- 指定したレジスタ・モードのオブジェクト・モジュール・ファイルを生成します。
- cx が使用するレジスタを 32 本（32 レジスタ・モード），26 本（26 レジスタ・モード），22 本（22 レジスタ・モード，または common レジスタ・モード）のいずれかに制限し，オブジェクト・モジュール・ファイル内にレジスタ・モードを示すマジック・ナンバを埋め込みます。
- common レジスタ・モードは，レジスタ・モードに依存しないオブジェクト・モジュール・ファイルを生成するために使用します。
- *mode* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

レジスタ・モード（*mode*）	作業用レジスタ	レジスタ変数用レジスタ
common	r10 ～ r14	r25 ～ r29
22	r10 ～ r14	r25 ～ r29
26	r10 ～ r16	r23 ～ r29
32	r10 ～ r19	r20 ～ r29

- *mode* を省略した場合は，エラーとなります。
- -Xreg_mode=common 指定時，setjmp，longjmp は -Xreg_mode=32 指定時と同じ動作をします。

  このため，r20 ～ r24 の値を setjmp 呼び出し後に変更しても，longjmp 呼び出し後には setjmp 呼び出し前の値に戻ります。
- -Xreg_mode=common 指定時，ランタイム関数は -Xreg_mode=32 指定時と同じ動作をします。

  このため，例外発生時に r15 ～ r19 の値を matherrf，または matherrd 内で更新しても，ランタイム関数を呼び出したプログラムには反映されません。
- C ソース・ファイルに対して，使用可能なレジスタのみを用いたコードを生成します。
- 本オプションは，すべてのソース・ファイルに対して同じレジスタ・モードを指定する必要があります。

  ソース・ファイルごとに指定を変えることはできません。

  レジスタ・モードが異なるオブジェクト・モジュール・ファイルが混在している場合は，リンク時に警告を出力して，リンク処理を続行します。

- 本オプションを指定することにより，ソフトウエア・レジスタ・バンク機能のレジスタ・モードを切り替えることができます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［レジスタ・モード］カテゴリの［レジスタ・モード］

［使用例］

- 22 レジスタ・モードのオブジェクト・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xreg_mode=22 main.c
```

-Xsdata

.sdata セクション，または .sbss セクションに配置するデータの最大サイズを指定します。

[指定形式]

```
-Xsdata=num
```

- 省略時解釈

  すべてのデータを .sdata ／ .sbss セクションに配置します。

  ただし，const 修飾された静的変数は，.const セクションに割り付けます。

[詳細説明]

- *num* バイト以下のデータを .sdata セクション，または .sbss セクションに配置します。
- #pragma section 指令で .sdata ／ .sbss セクションを指定したデータは，そのサイズに関係なく .sdata ／ .sbss セクションに配置します。
- 不完全型の配列（ファイル内でサイズが不明な配列）に対しては，本オプションは適用しません。
- *num* には，0 ～ 65535 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- *num* に指定する値の目安は，-Xsdata_info オプションで出力することができます。
- ソース・ファイルごとに異なるオプションを指定すると，変数の配置，および参照方法が異なるコードを生成して，リンク時にエラー，または警告を出力することがあります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［sdata/sbss セクションに配置するデータ長の上限値 (バイト)］

[使用例]

- 16 バイト以下のデータを .sdata セクション，または .sbss セクションに配置します。

```
>cx -CF3746 -Xsdata=16 main.c
```

-Xsconst

.sconst セクションへの定数データの配置を指定します。

[指定形式]

```
-Xsconst[=num]
```

- 省略時解釈

  すべての const データを .const セクションに配置します。

[詳細説明]

- *num* バイト以下の const データ（const 属性のデータ，および文字列リテラル）を .sconst セクションに配置します。
- セクションが指定されないデータのみ有効です。
- 不完全型の配列（ファイル内でサイズが不明な配列）に対しては，本オプションは適用しません。
- *num* には，0 ～ 32767 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合，すべての const データを .sconst セクションに配置します。
- ソース・ファイルごとに異なるオプションを指定すると，変数の配置，および参照方法が異なるコードを生成して，リンク時にエラー，または警告を出力することがあります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［sconst セクションのデータ長の上限値 (バイト)］

[使用例]

- 16 バイト以下の const データを .sconst セクションに配置します。

```
>cx -CF3746 -Xsconst=16 main.c
```

-Xfloat【V850E2V3】

浮動小数点演算命令の生成を制御します。

［指定形式］

```
-Xfloat=type
```

- 省略時解釈

  -C オプションで指定したターゲット・デバイスが FPU を持つ場合は，浮動小数点演算命令を生成します。
  また，FPU 数学対応ライブラリをリンクします。
  それ以外の場合（-C オプションで指定したターゲット・デバイスが FPU を持たない場合，または -C オプションを指定せず -Xcommon オプションを指定した場合）は，浮動小数点演算命令を生成せず，ランタイム・ライブラリの呼び出し命令を生成します。また，FPU 数学対応ライブラリはリンクしません。

［詳細説明］

- 浮動小数点演算命令の生成を制御します。
- 本オプションは，ターゲット・デバイスが FPU を持つ場合に指定します。
- *type* に指定可能なものを以下に示します。
  これ以外のものを指定した場合は，エラーとなります。

soft	浮動小数点演算に対して，ランタイム・ライブラリの呼び出し命令を生成します。 また，FPU 数学対応ライブラリはリンクしません。
fpu	浮動小数点演算に対して，FPU（浮動小数点ユニット）の浮動小数点演算命令を生成します。 また，FPU 数学対応ライブラリをリンクします。 FPU を持たないターゲット・デバイス（V850E2V3 以外のデバイスも含む）に対して，本項目を指定した場合は，警告を出力して無視します。

- *type* を省略した場合は，エラーとなります。
- ターゲット・デバイスが FPU を持たない場合，CX V1.10 以降では，FPU 命令を含むオブジェクト・モジュール・ファイルをリンクするとエラーとなり，CX V1.01 以前のバージョンで生成したオブジェクト・モジュール・ファイルをリンクすると，FPU 命令の有無にかかわらず警告を出力します（CA850 で生成したオブジェクト・モジュール・ファイルはそのままリンクします）。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［浮動小数点演算方法］
  - ［個別コンパイル・オプション］タブの［出力コード］カテゴリの［浮動小数点演算方法］

［使用例］

- 浮動小数点演算に対して，ランタイム・ライブラリの呼び出し命令を生成します。

```
>cx -CF3746 -Xfloat=soft main.c
```

-Xfar_jump

far jump の出力を制御します。

[指定形式]

```
-Xfar_jump=file
```

- 省略時解釈

  関数への分岐に対して，jarl，または jr 命令を生成します。

[詳細説明]

- C ソース・ファイルについて，far jump 呼び出し関数一覧ファイル *file* 中で指定した関数（C 言語で記述した割り込み関数を含む）への分岐に対して，jmp 命令（V850E の場合），または jarl32，および jr32 命令（V850E2，および V850E2V3 の場合）を使用したコードを生成します。
- *file* の推奨拡張子は，.fjp です。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 関数本体が，jarl，および jr 命令では分岐できない範囲（± 2M バイト以上）にあり，リンク時にエラーとなる場合，本オプションを使用してコンパイルし直します。
- 本オプションを複数指定した場合は，最後に指定したものが有効となります。
- 出力コード例を以下に示します。
  - C ソース

```
        far func();     /* デフォルトでは jarl  far func, lp を出力する */
```

  - 出力アセンブラ・ソース（V850E の場合）

```
        movea   far func, tp, r10
        movea   .BB.LABEL.15, tp, lp
        jmp     [r10]
.BB.LABEL.15:
```

  - 出力アセンブラ・ソース（V850E2，および V850E2V3 の場合）

```
        jarl32  far func, lp
```

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［Far Jump ファイル名］
  - ［個別コンパイル・オプション］タブの［出力コード］カテゴリの［Far Jump ファイル名］

備考 far jump 呼び出し関数一覧ファイルの書式に関する注意事項を以下に示します。

- 1 行に 1 個の関数名を記述します。
  複数の関数名を記述した場合は，最初の名前のみを有効とします。
- 記述する関数名は，C 言語の関数名の先頭に“_”を付けた名前（アセンブラ・ソースでのラベル名）とします。
  ただし，関数名の代わりに，以下の形式で指定することもできます。

形式	意味
{all function}	すべての関数呼び出しを対象とします。
{all interrupt}	すべての割り込み関数呼び出しを対象とします。

- 関数だけでなく，ランタイム・ルーチンも指定可能です。
  その場合は，関数名としてランタイム・ルーチン名をそのまま記述してください。
- 関数名の前後に，空白やタブを挿入することが可能です。
- 使用可能な文字は，ASCII 文字のみです。
- コメントを挿入することはできません。
- 1 行に記述可能な文字数は，1023 文字までです（空白やタブも含みます）。

関数の指定例を以下に示します。

```
_func_led
_func_beep
_func_motor
    :
_func_switch
```

［使用例］

- func.fjp 中で指定した関数への分岐に対して，jmp 命令を使用したコードを生成します。

```
>cx -CF3746 -Xfar_jump=func.fjp main.c
```

-Xdiv【V850E2V3】

除算に対して，div，および divu 命令を生成します。

［指定形式］

```
-Xdiv
```

- 省略時解釈

  除算に対して，divq，および divqu 命令を生成します。

［詳細説明］

- 除算に対して，divq，および divqu 命令を生成する代わりに，div，および divu 命令を生成します。
- divq，および divqu 命令は高速ですが，実行サイクル数がオペランドの値により変わります。
  そのため，リアルタイム性の保証などのために，常に実行サイクル数が一定であることが必要な場合に，本オプションを指定します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［div ／ divu 除算命令を生成する］
  - ［個別コンパイル・オプション］タブの［出力コード］カテゴリの［div ／ divu 除算命令を生成する］

［使用例］

- 除算に対して，div，および divu 命令を生成します。

```
>cx -CF3746 -Xdiv main.c
```

アセンブラ制御指定

アセンブラ制御指定オプションには，次のものがあります。

- -Xasm_far_jump【V850E2】【V850E2V3】

-Xasm_far_jump【V850E2】【V850E2V3】

アセンブラ・ソース・ファイルに対して，far jump の出力を制御します。

［指定形式］

```
-Xasm_far_jump
```

- 省略時解釈

  jarl，または jr 命令としてアセンブルを行います。

［詳細説明］

- アセンブラ・ソース・ファイルに対して，ソース中に記述されたすべての jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。
- 個別の命令ごとに制御したい場合は，ソース中で jarl22 ／ jarl32，jr22 ／ jarl32 のように明記します。
- 本オプションは，jump 命令には影響しません。
- C ソース・ファイルに対して本オプションを指定した場合，警告を出力せずに無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［コンパイル・オプション］タブの［出力コード］カテゴリの［32 ビット分岐命令を使用する］

［使用例］

- アセンブラ・ソース中に記述されたすべての jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。

```
>cx -CF3746 -Xasm_far_jump main.c
```

ライブラリ・リンク制御

ライブラリ・リンク制御オプションには，次のものがあります。

- -l
- -L
- -Xno_stdlib
- -Xno_startup
- -Xstartup

-l

リンク時に使用するライブラリ・ファイルを指定します。

[指定形式]

```
-lstring[,string]...
```

- 省略時解釈

  標準ライブラリ，数学ライブラリ，および標準スタートアップ・ルーチンのみをリンクします。

  -Xno_romize オプションを指定しない場合は，ROM 化用領域確保コード・ファイルもリンクします。

[詳細説明]

- リンク時に使用するライブラリ・ファイル lib*string*.lib を指定します。
- cx は，すべてのオブジェクト・モジュール・ファイルのリンク後，未解決な外部シンボルの参照を解決する際に，ライブラリ・ファイル（lib*string*.lib）を参照します。
- *string* を省略した場合は，エラーとなります。
- -L オプションと同時に指定した場合は，-L オプションで指定したフォルダからライブラリ・ファイルを検索します。

  -L オプションを指定しない場合は，標準フォルダ[注]から検索します。
- 本オプションで指定したライブラリ・ファイルが見つからない場合は，メッセージを出力せずに，正常にリンク処理を続行します。
- 複数のライブラリ・ファイルを指定した場合は，指定した順番に検索します。
- cx は，本オプションで指定したライブラリ以外にも，標準ライブラリ（libc.lib），および標準スタートアップ・ルーチン（cstart.obj）を自動的にリンクします。

  -Xno_romize オプションを指定しない場合は，ROM 化用領域確保コード・ファイル（rompcrt.obj）も自動的にリンクします。

  これを抑止するには，-Xno_stdlib オプション，および -Xno_startup オプションを使用します。

注　*本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ lib ¥ 850e

［使用例］

- リンク時に使用するライブラリ・ファイル libsmp.lib を指定します。

```
>cx -CF3746 -lsmp main.c
```

-L

ライブラリ・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Lpath[,path]...
```

- 省略時解釈

  ライブラリを標準ライブラリ・フォルダからのみ検索します。

[詳細説明]

- ライブラリをフォルダ *path*，標準ライブラリ・フォルダ[注]の順に検索します。
- *path* が存在しない場合は，警告を出力します。
- *path* を省略した場合は，エラーとなります。

注　*本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ lib ¥ 850e

なお，V850E2V3 で FPU 対応数学ライブラリをリンクする場合は，以下も標準ライブラリ・フォルダとします。

本製品のインストール・フォルダ¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ lib ¥ 850e2v3f

[使用例]

- ライブラリをフォルダ lib，標準フォルダの順に検索します。

```
>cx -CF3746 -Llib main.c
```

-Xno_stdlib

標準ライブラリのリンクを抑止します。

[指定形式]

```
-Xno_stdlib
```

- 省略時解釈

  標準ライブラリをリンクします。

[詳細説明]

- 標準ライブラリをリンクしません。

[使用例]

- 標準ライブラリをリンクしません。

```
>cx -CF3746 -Xno_stdlib main.c
```

-Xno_startup

標準スタートアップ・ルーチンのリンクを抑止します。

[指定形式]

```
-Xno_startup
```

- 省略時解釈

  標準スタートアップ・ルーチンをリンクします。

[詳細説明]

- 標準スタートアップ・ルーチン（デフォルト：cstart.obj，-Xno_romize オプション指定時：cstartN.obj）をリンクしません。
- 本オプションは，スタートアップ・ルーチンとして特別なファイルを用意するのではなく，スタートアップ・ルーチンを含むファイルをほかのソース・ファイルと同等に扱いたい場合に指定します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［入力ファイル］カテゴリの［標準のスタートアップを使用する］

[使用例]

- 標準スタートアップ・ルーチンをリンクしません。

```
>cx -CF3746 -Xno_startup main.c
```

-Xstartup

スタートアップ・ルーチンを指定します。

[指定形式]

```
-Xstartup=file
```

- 省略時解釈

  標準スタートアップ・ルーチンをリンクします。

[詳細説明]

- 標準スタートアップ・ルーチン（デフォルト：cstart.obj，-Xno_romize オプション指定時：cstartN.obj）の代わりに，*file* をスタートアップ・ルーチンとしてリンクします。
- *file* には，オブジェクト・モジュール・ファイルを指定します。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 入力ファイルを指定していない場合は，*file* のみで構成させるロード・モジュール・ファイルを生成します。

[使用例]

- start.obj をスタートアップ・ルーチンとしてリンクします。

```
>cx -CF3746 -Xstartup=start.obj main.c
```

リンク制御

リンク制御オプションには，次のものがあります。

- -Xlink_directive
- -Xmap
- -Xsymbol_dump
- -Xsecurity_id
- -Xoption_byte
- -Xentry_address
- -Xrelinkable_object
- -Xregmode_info
- -Xforce_link
- -Xsdata_info
- -Xtwo_pass_link
- -Xignore_address_error
- -Xmultiple_symbol
- -Xlink_check_off
- -Xalign_fill
- -Xrescan
- -Xstrip
- -Xflash
- -Xflash_ext_table

-Xlink_directive

リンク・ディレクティブ・ファイルを指定します。

[指定形式]

```
-Xlink_directive=file
```

- 省略時解釈

  デフォルトのリンク・ディレクティブ・ファイルを使用します。

[詳細説明]

- リンク・ディレクティブ・ファイル *file* 内のリンク・ディレクティブに従ってリンクを行います。
- *file* の推奨拡張子は，.dir です。
- *file* を省略した場合は，エラーとなります。
- 本オプションを複数指定した場合は，あとから指定したオプションが有効となり，先に指定したオプションは無視します。
- リンク・ディレクティブ・ファイルについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。

[使用例]

- リンク・ディレクティブ・ファイル link.dir 内のリンク・ディレクティブに従ってリンクを行います。

```
>cx -CF3746 -Xlink_directive=link.dir main.c
```

-Xmap

リンク・マップ・ファイルを出力します。

[指定形式]

```
-Xmap[=file]
```

- 省略時解釈

  リンク・マップ・ファイルを出力しません。

[詳細説明]

- リンク・マップ・ファイル *file* を出力します。
- リンク・マップ・ファイルの内容を以下に示します。
  - 指定したオブジェクト・モジュール・ファイルに含まれる入力セクションのメモリ空間への割り付け
  - 入力セクションを結合して生成するロード・モジュール・ファイルを構成する出力セクションのメモリ空間への割り付け
  - シンボルのアドレス情報
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* にファイル名のみを指定した場合は，ロード・モジュール・ファイルと同じフォルダに，指定したファイル名で出力します。
- *file* を省略した場合は，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .map に置き換えた名前で出力します。
- -Xno_romize オプションの有無により，出力するリンク・マップ・ファイルは以下のようになります。
  - -Xno_romize オプションの指定あり

    リンク処理後までのリンク・マップ・ファイルを出力します。
  - -Xno_romize オプションの指定なし

    ROM 化処理後までのリンク・マップ・ファイルを出力します。

[使用例]

- リンク処理後のリンク・マップ・ファイル smp1.map を出力します。

```
>cx -CF3746 -Xmap=smp1.map main.c -Xno_romize
```

- ROM 化処理後のリンク・マップ・ファイル smp2.map を出力します。

```
>cx -CF3746 -Xmap=smp2.map main.c
```

-Xsymbol_dump

リンク・マップ・ファイルにシンボル情報を出力します。

[指定形式]

```
-Xsymbol_dump
```

- 省略時解釈

  リンク・マップ・ファイルにシンボル情報を出力しません。

[詳細説明]

- -Xmap オプションで指定したリンク・マップ・ファイルにシンボル情報を出力します。
- 本オプションは，-Xmap オプション指定時のみ有効です。

  -Xmap オプションを指定しない場合は，警告を出力して，本オプションを無視します。

[使用例]

- リンク・マップ・ファイル smp.map にシンボル情報を出力します。

```
>cx -CF3746 -Xmap=smp.map -Xsymbol_dump main.c
```

-Xsecurity_id

セキュリティ ID を設定します。

[指定形式]

```
-Xsecurity_id=id
-Xsecurity_id=none
```

- 省略時解釈

  セキュリティ ID として 0xFFFFFFFFFFFFFFFFFFFF を自動生成します。

  セキュリティ ID 機能を持たないデバイスを指定した場合は，何も行いません。

[詳細説明]

- セキュリティ ID を 70H ～ 79H 番地に設定します。
- *id* には，10 バイト以内の 16 進数を指定します。

  指定した値が 10 バイトに満たない場合は，上位ビットを 0 で補てんします。
- *id* にデバイスで対応している桁数を越えた値を指定した場合は，エラーとなります。

  ただし，配置アドレス，および *id* のデータ・サイズは，デバイスにより異なります。
- none を指定した場合は，セキュリティ ID の自動生成を抑止します。
- *id* ，または none を省略した場合は，エラーとなります。
- セキュリティ ID の設定は，アセンブラ・ソース中で以下のように記述することでも可能です。

```
.CSEG   SECUR_ID
.DB     0x11
    :
.DB     0xAA
```

- 本オプションとアセンブラ・ソースでの設定が重なった場合は，警告を出力して，本オプションを優先します。
- セキュリティ ID 機能を持たないデバイスに対して，本オプションを指定した場合は，エラーとなります。
- -Xmulti オプションと同時に指定した場合は，警告を出力して，本オプションを無視します。

[使用例]

- セキュリティ ID として 0x112233445566778899AA （0x70 番地に 0x11，0x71 番地に 0x22，0x72 番地に 0x33，0x73 番地に 0x44，0x74 番地に 0x55，0x75 番地に 0x66，0x76 番地に 0x77，0x77 番地に 0x88，0x78 番地に 0x99，0x79 番地に 0xAA）を設定します。

```
>cx -CF3746 -Xsecurity_id=0x112233445566778899AA file1.c file2.c
```

-Xoption_byte

ユーザ・オプション・バイトを設定します。

[指定形式]

```
-Xoption_byte=none
```

- 省略時解釈

  デバイス・ファイルの初期値をユーザ・オプション・バイト領域に埋め込みます。

  オプション・バイト機能をサポートしていないデバイスに対しては，何も行いません。

[詳細説明]

- ユーザ・オプション・バイト領域の生成を抑止します。
- ユーザ・オプション・バイトは，7AH ～ 7FH 番地に設定されます。

  7AH ～ 7FH 番地のユーザ・オプション・バイトは，ソース中に記述することも可能です。

  ただし，配置アドレスは，デバイスにより異なります。
- none を省略した場合は，エラーとなります。
- ユーザ・オプション・バイトの設定は，アセンブラ・ソース中で以下のように記述することにより，任意に設定することも可能です。

```
.CSEG OPT_BYTE
.DB 0x11
.DB 0x22
.DB 0x33
.DB 0x44
.DB 0x55
.DB 0x66
```

- オプション・バイト機能をサポートしていないデバイスに対して，本オプションを指定した場合は，本オプションを無視します。
- ユーザ・オプション・バイトをユーザ空間に設定することができないデバイスに対して，本オプションを指定した場合は，エラーとなります。
- -Xmulti オプションと同時に指定した場合は，警告を出力して，本オプションを無視します。

[使用例]

- ユーザ・オプション・バイト領域の生成を抑止します。

```
>cx -CF3746 -Xoption_byte=none file1.c file2.c
```

-Xentry_address

エントリ・ポイント・アドレスを指定します。

[指定形式]

```
-Xentry_address=symbol
```

- 省略時解釈

  生成するロード・モジュール・ファイルのエントリ・ポイント・アドレス値を，以下の規則で決定します。

  - シンボル __start が存在する場合は，その値
  - __start が存在しない場合は，生成するロード・モジュール・ファイル内の最下位に割り付けられた text 属性のセクションの先頭アドレス
  - text 属性セクションが存在しない場合は，0

[詳細説明]

- シンボル *symbol* の値を生成するロード・モジュール・ファイルのエントリ・ポイント・アドレス値（ヘキサ変換時に使用します）とします。
- *symbol* が存在しない場合は，エラーとなります。
- *symbol* を省略した場合は，エラーとなります。

[使用例]

- シンボル __my_start の値を生成するロード・モジュール・ファイルのエントリ・ポイント・アドレス値とします。

```
>cx -CF3746 -Xentry_address=__my_start main.c
```

-Xrelinkable_object

再配置可能なオブジェクト・モジュール・ファイルを生成します。

[指定形式]

```
-Xrelinkable_object
```

- 省略時解釈

  実行可能なオブジェクト・モジュール・ファイルを生成しようとします。

  リンク終了後に未解決な外部参照が残されていた場合，エラーを出力して，リンクを中止します。

  その際，ロード・モジュール・ファイルは生成しません。

[詳細説明]

- 再配置可能なオブジェクト・モジュール・ファイルを生成します。
- 本オプションを指定すると，リンク終了後に未解決な外部参照が残されていた場合，メッセージを出力せず，リンクを正常終了します。
- cx によって生成したオブジェクト・モジュール・ファイルを再配置の対象として指定する場合，本オプションを使用して，再配置の対象となるオブジェクト・モジュール・ファイルを生成します。
- 本オプションを指定した場合，リンク・ディレクティブはセグメント・ディレクティブ部分のタイプ／属性のみが有効になり，その他は無視します。
- 本オプションを指定した場合，予約シンボルの作成は行いません。
- 本オプションを指定した場合，-Xno_startup，および -Xno_romize オプションが暗黙に指定されたものとみなします。

[使用例]

- 再配置可能なオブジェクト・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xrelinkable_object main.c -Xno_startup -Xno_romize
```

-Xregmode_info

異なるレジスタ・モードが混在している場合，詳細情報を出力します。

[指定形式]

```
-Xregmode_info
```

- 省略時解釈

  異なるレジスタ・モードが混在している場合，詳細情報を出力しません。

[詳細説明]

- すべての入力オブジェクト・モジュール・ファイルに対して，異なるレジスタ・モードが混在している場合，詳細情報を出力して，警告の原因となっている入力オブジェクト・モジュール・ファイルを特定します。
- レジスタ・モードが統一されている場合は，情報は出力しません。

[使用例]

- すべての入力オブジェクト・モジュール・ファイルに対して，異なるレジスタ・モードが混在している場合，詳細情報を出力します。

```
>cx -CF3746 -Xregmode_info file1.c file2.c
```

-Xforce_link

内部 ROM/RAM のオーバフロー時に，リンク処理を続行します。

[指定形式]

```
-Xforce_link
```

- 省略時解釈

  内部 ROM/RAM のオーバフロー時に，エラーを出力して，リンク処理を終了します。

[詳細説明]

- 内部 ROM/RAM のオーバフロー時に，警告を出力して，リンク処理を続行します。
- 出力ファイルの生成は行いません。
- 超過したサイズを標準エラー出力に出力します。

[使用例]

- 内部 ROM/RAM のオーバフロー時に，警告を出力して，リンク処理を続行します。

```
>cx -CF3746 -Xforce_link main.c
```

-Xsdata_info

-Xsdata オプションのパラメータに対して，目安として用いることのできる情報を標準出力に出力します。

[指定形式]

```
-Xsdata_info
```

- 省略時解釈

  -Xsdata オプションのパラメータに対して，目安として用いることのできる情報を出力しません。

[詳細説明]

- -Xsdata オプション（sdata ／ sbss セクションに配置するデータの最大サイズの指定）のパラメータに対して，目安として用いることのできる情報を標準出力に出力します。
- *OK* と表示された数値を用いると，sdata ／ sbss の領域へは，その数値以下のサイズを持つデータが割り当てられます。

[使用例]

- -Xsdata オプションのパラメータに対して，目安として用いることのできる情報を標準出力に出力します。

```
>cx -CF3746 -Xsdata_info main.c
```

-Xtwo_pass_link

2パス・モードでリンクを行います。

[指定形式]

```
-Xtwo_path_link
```

- 省略時解釈

  1パス・モードでリンクを行います。

[詳細説明]

- 2パス・モードでリンクを行います。
- 2パス・モードは，1パス・モードよりも処理が遅いですが，より大きなサイズのファイルを処理することができます。

[使用例]

- 2パス・モードでリンクを行います。

```
>cx -CF3746 -Xtwo_path_link main.c
```

-Xignore_address_error

リンク時のリロケーション処理において，不正箇所があった場合，リンク処理を続行します。

[指定形式]

```
-Xignore_address_error
```

- 省略時解釈

  リンク時のリロケーション処理において，不正箇所があった場合，エラーとします。

[詳細説明]

- リンク時のリロケーション処理において，以下の不正箇所があった場合，エラーとはせず，警告を出力して，リンク処理を続行します。
  - 未解決な外部参照のアドレス計算結果が不正な場合
  - 配置されるセクションとの関係が不正な場合

  具体的には，以下のような場合を指します。
  - GP 相対リロケーションのための GP シンボルが存在しない場合（LOCAL ／ GLOBAL ／ EXTERN）
  - PC22/26 のリロケーション結果が奇数アドレスへの分岐である場合
  - PC 相対リロケーション結果が許容範囲を越えた場合（シンボルあり／シンボルなし）
  - PC 相対以外のリロケーション結果が許容範囲を越えた場合（シンボルあり／シンボルなし）
- 不正と判断された未解決な外部参照に対しては，アドレスの計算結果の値は入れず，元の値を残します。

[使用例]

- リンク時のリロケーション処理において，未解決な外部参照のアドレス計算結果が不正な場合，警告を出力して，リンク処理を続行します。

```
>cx -CF3746 -Xignore_address_error main.c
```

-Xmultiple_symbol

多重定義されたすべての外部シンボルに対してエラー・メッセージを出力します。

[指定形式]

```
-Xmultiple_symbol
```

- 省略時解釈

  多重定義された最初の外部シンボルに対してエラー・メッセージを出力して，リンク処理を中止します。

[詳細説明]

- 多重定義されたすべての外部シンボルとファイル名に対してエラー・メッセージを出力して，リンク処理を中止します。

[使用例]

- 多重定義されたすべての外部シンボルに対してエラー・メッセージを出力して，リンク処理を中止します。

```
>cx -CF3746 -Xmultiple_symbol main.c
```

-Xlink_check_off

リンク時のチェックを抑止します。

[指定形式]

```
-Xlink_check_off=string[,string]
```

- 省略時解釈

外部シンボルをリンクする際，シンボルのサイズのチェックを行い，サイズの違いを検出した場合は，警告を出力して，リンク処理を続行します。
その際，実際にシンボルが定義されているファイルのシンボル・サイズが有効となります。
未定義シンボルをリンクする際，シンボルのサイズ，および整列条件のチェックを行い，違いを検出した場合は，警告を出力して，リンク処理を続行します。
内蔵 ROM 領域となっているアドレスに，アプリケーションの配置がオーバラップしている場合は，警告を出力します。

[詳細説明]

- リンク時のチェックを抑止します。
- *string* に指定可能なものを以下に示します。
  これ以外のものを指定した場合は，エラーとなります。

symbol	外部シンボルをリンクする際，シンボルのサイズ，および整列条件のチェックを行いません。
undefined	未定義外部シンボルをリンクする際，シンボルのサイズ，および整列条件のチェックを行いません。
irom	内蔵 ROM 領域への配置に対して，チェックを行いません。 つまり，内蔵 ROM 領域となっているアドレスに，アプリケーションの配置がオーバラップしていても，警告を出力しません。

- *string* を省略した場合は，エラーとなります。
- アプリケーションを ROM レス・モードで作成する場合は，irom を指定したものとみなします。

注意　シングルチップ・モード選択時の内蔵 ROM オーバのチェックには対応していません。
-Xlink_check_off=irom を指定して内蔵 ROM オーバフローのチェックを無効とし，リンク・マップで確認してください。

[使用例]

- 外部シンボルをリンクする際，サイズ，および整列条件のチェックを行いません。

```
>cx -CF3746 -Xlink_check_off=symbol main.c
```

-Xalign_fill

アライン・ホールの充てん値を指定します。

[指定形式]

```
-Xalign_fill=value
```

- 省略時解釈

  生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値を 0x0000 とします。

[詳細説明]

- 生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値 *value* を指定します。
- *value* に指定可能な値の範囲は，0x0000 ～ 0xFFFF です。

  この範囲外の値を指定した場合は，エラーとなります。
- 指定した値が 4 桁に満たない場合は，上位の桁を 0 で補てんします。
- *value* を省略した場合は，エラーとなります。
- 本オプションを指定する場合は，-Xtwo_path_link オプションを指定して 2 パス・モードでリンクを行う必要があります。

  -Xtwo_path_link オプションを指定しない場合は，エラーとなります。

[使用例]

- 生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値を 0xFFFF とします。

```
>cx -CF3746 -Xalign_fill=0xFFFF -Xtwo_path_link main.c
```

-Xrescan

-l オプションで指定したライブラリ・ファイルの再スキャンを行います。

[指定形式]

```
-Xrescan
```

- 省略時解釈

  -l オプションで指定したライブラリ・ファイルの再スキャンを行いません。

[詳細説明]

- -l オプションで指定したライブラリ・ファイルの再スキャンを行います。
- 本オプションを指定すると，ライブラリのリンク順によるシンボル未解決を防ぐことができます。

[使用例]

- ライブラリ・ファイル libtest1.lib，libtest2.lib の再スキャンを行います。

```
>cx -CF3746 -Xrescan main.c -ltest1 -ltest2
```

-Xstrip

デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xstrip
```

- 省略時解釈
- ロード・モジュール・ファイルの生成において，デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルが存在する場合，これらの削除を行いません。

[詳細説明]

- ロード・モジュール・ファイルの生成において，デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。

[使用例]

- ロード・モジュール・ファイルの生成において，デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xstrip main.c
```

-Xflash

フラッシュ領域側のロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xflash=file
```

- 省略時解釈

  ブートーフラッシュ再リンク機能を使用する場合は，ブート領域側のロード・モジュール・ファイルを生成します。

  ブートーフラッシュ再リンク機能を使用しない場合は，通常のリンク処理を行います。

[詳細説明]

- ブートーフラッシュ再リンク機能を使用する場合に，フラッシュ領域側のロード・モジュール・ファイルを生成します。

  その際，ブート領域側のロード・モジュール・ファイル *file* のシンボル情報を参照して，リンク処理を行います。
- *file* には，ブートーフラッシュ再リンク機能を使用して生成したブート領域側のロード・モジュール・ファイルを指定します。

  ここで指定するロード・モジュール・ファイルは，ROM 化処理前のもの（-Xno_romize，または -Xlink_output オプションを指定して生成したもの）である必要があります。
- *file* に存在しないファイルを指定した場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，-Xflash_ext_table オプションと同時に指定する必要があります。

[使用例]

- フラッシュ領域側の分岐テーブル先頭アドレスが 0x200 番地にあるものとして，ブート領域側のロード・モジュール・ファイル boot.lmf を生成します。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xlink_output=boot.lmf boot1.c boot2.c -
Xlink_directive=boot.dir
```

0x200 番地に分岐テーブルを生成して，フラッシュ領域側のロード・モジュール・ファイル flash.lmf を生成します。

その際，ブート領域側のロード・モジュール・ファイル boot.lmf のシンボル情報を参照して，リンク処理を行います。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xflash=boot.lmf -oflash.lmf flash1.c flash2.c  -
Xlink_directive=flash.dir
```

-Xflash_ext_table

ブートーフラッシュ再リンク機能用の分岐テーブル先頭アドレス値を指定します。

[指定形式]

```
-Xflash_ext_table=address
```

- 省略時解釈

  ブートーフラッシュ再リンク機能用のロード・モジュール・ファイルを生成せず，通常のリンク処理を行います。

[詳細説明]

- アドレス *address* を分岐テーブル先頭アドレス値として，ブートーフラッシュ再リンク機能用のロード・モジュール・ファイルを生成します。

  ブートーフラッシュ再リンク機能についての詳細は，「B. 1. 6　ブートーフラッシュ再リンク機能」を参照してください。
- -Xflash オプションの指定の有無により，フラッシュ領域側／ブート領域側のロード・モジュール・ファイルの生成を指定します。
  - ブート領域側のロード・モジュール・ファイルの生成を指定した場合（-Xflash オプションの指定なし）

    フラッシュ領域側への分岐処理となります。

    その際，本オプションで指定したアドレスに生成される分岐テーブルへの分岐として処理します。
  - フラッシュ領域側のロード・モジュール・ファイルの生成を指定した場合（-Xflash オプションの指定あり）

    本オプションで指定したアドレスに，本来の分岐先への分岐命令を持つ分岐テーブルを生成します。
- *address* に指定可能な値の範囲は，0x00000000 ～ 0xFFFFFFFF です。

  この範囲外の値を指定した場合は，エラーとなります。
- 指定した値が 8 桁に満たない場合は，上位の桁を 0 で補てんします。
- *address* に奇数を指定した場合は，偶数に補正したのち，警告を出力して，処理を続行します。
- *address* を省略した場合は，エラーとなります。
- *address* は，ブート領域側／フラッシュ領域側のロード・モジュール・ファイルの生成の際に，同じ値とする必要があります。

  異なる値を指定した場合は，正しく動作しません。

  また，エラー・チェックも行っていません。
- *address* は，フラッシュ領域側 ROM 内である必要があります。

  指定したアドレスがどちらの領域であるかの判断ができないため，エラー・チェックは行っていません。
- 本オプションを指定することにより，フラッシュ領域側のロード・モジュール・ファイルの生成時には，*address* を先頭とする，サイズ（（ID 値[注]の最大＋1）×分岐テーブルのエントリ・サイズ）バイトのセクション .ext_table を自動生成します。

  このセクションは，リンク・ディレクティブ・ファイルで配置指定を行う必要はありませんが，配置するため領域を空けておく必要があります。

注 アセンブラ・ソース・ファイルに .ext_func 疑似命令で指定された値

- ブート領域側のロード・モジュール・ファイルを生成する場合は，ROM 化処理前のロード・モジュール・ファイルを保存する必要があります。
  これは，フラッシュ領域側のロード・モジュール・ファイルを生成する際に，-Xflash オプションでそのファイルを指定するためです。
  したがって，本オプション指定時に -Xflash オプションを同時に指定しない場合は，-Xlink_output オプション（ROM 化を行う場合），または -Xno_romize（ROM 化を行わない場合）と同時に指定する必要があります。
- 本オプションは，-Xrelinkable_object オプションと同時に指定することはできません。
  また，-Xrelinkable_object オプションにより生成したロード・モジュール・ファイルを入力した場合は，正しく動作しません。

［使用例］

- 分岐テーブル先頭アドレス値を 0x200 として，ブート領域側のロード・モジュール・ファイル boot.lmf を生成します。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xlink_output=boot.lmf boot.c
```

ROM 化制御

ROM 化制御オプションには，次のものがあります。

- -Xno_romize
- -Xrompcrt
- -Xrompsec_start
- -Xrompsec_data
- -Xrompsec_text
- -Xrompsec_only
- -Xromize_check_off

-Xno_romize

ROM 化処理を抑止します。

[指定形式]

```
-Xno_romize
```

- 省略時解釈

  ROM 化処理を行います。

  また，以下をリンクします。

  - ROM 化用領域確保コード・ファイル（-Xno_stdlib オプション指定時を除く）
  - コピー関数 _rcopy の呼び出しを含むスタートアップ・ルーチン（cstart.obj）（-Xstartup，または -Xno_startup オプション指定時を除く）

[詳細説明]

- ROM 化処理を抑止します。

  また，ROM 化用領域確保コード・ファイルのリンクを行いません。

  コピー関数 _rcopy の呼び出しを含まないスタートアップ・ルーチン（cstartN.obj）をリンクします（-Xstartup，または -Xno_startup オプション指定時を除く）。
- ROM 化処理とは，RAM に展開するデータを ROM 上に持たせておき，アプリケーションの最初に，そのデータを ROM から RAM へコピーするルーチンを追加する処理のことです。
- 初期値ありデータなどがないアプリケーションにおいて，本オプションを指定すると，コードを削減することができます。

［使用例］

- ROM 化処理を抑止します。

```
>cx -CF3746 -Xno_romize main.c
```

-Xrompcrt

ROM 化用領域確保コード・ファイルを指定します。

[指定形式]

```
-Xrompcrt=file
```

- 省略時解釈

  -Xno_romize オプションを指定しない場合に，標準の ROM 化用領域確保コード・ファイル（rompcrt.obj）を入力ファイルの最後にリンクします。

[詳細説明]

- 標準の ROM 化用領域確保コード・ファイルの代わりに，*file* を入力ファイルの最後にリンクします。
- *file* には，オブジェクト・モジュール・ファイルを指定します。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，-Xno_romize オプションと同時に指定した場合は無効となります。

[使用例]

- 標準の ROM 化用領域確保コード・ファイルの代わりに，rompack.obj を入力ファイルの最後にリンクします。

```
>cx -CF3746 -Xrompcrt=rompack.obj main.c
```

-Xrompsec_start

rompsec セクションの先頭アドレスを指定します。

[指定形式]

```
-Xrompsec_start=label
```

- 省略時解釈

  ラベル __S_romp の値を，生成する rompsec セクションの先頭アドレスとします。

[詳細説明]

- 指定したセクションのみ，ヘキサ出力を行います。
- ラベル *label* の値を，生成する rompsec セクションの先頭アドレスとします。
- *label* がロード・モジュール・ファイル中に存在しない場合，または本オプションを複数指定した場合は，あとから指定したオプションが有効となり，先に指定したオプションは無視します。
- *label* を省略した場合は，エラーとなります。

[使用例]

- ラベル romp_start の値を，生成する rompsec セクションの先頭アドレスとします。

```
>cx -CF3746 -Xrompsec_start=romp_start main.c
```

-Xrompsec_data

rompsec セクションに含めるデータ・セクションを指定します。

[指定形式]

```
-Xrompsec_data=section[,section]...
```

- 省略時解釈

  data 属性，または sdata 属性を持つすべてのセクション，および内蔵命令 RAM に配置されるセクションを rompsec セクションに含めます。

[詳細説明]

- rompsec セクションに含めるデータ・セクションを指定します。
- セクション *section* の内容とそのアドレス，およびサイズの情報を rompsec セクションに含めます。
- 本オプションは，data 属性，または sdata 属性を持つセクションに関するオプションです。
- *section* がロード・モジュール・ファイル中に存在しない場合は，エラーを出力して，処理を中止します。
- *section* には，空白を使用することはできません。
- *section* を省略した場合は，エラーとなります。
- 本オプションを複数指定した場合は，指定した順に各データ・セクションを rompsec セクションに含めます。

[使用例]

- セクション data1，および data2 を rompsec セクションに含めます。

```
>cx -CF3746 -Xrompsec_data=data1,data2 main.c
```

-Xrompsec_text

rompsec セクションに含めるテキスト・セクションを指定します。

[指定形式]

```
-Xrompsec_text=section[,section]...
```

- 省略時解釈

  内蔵命令 RAM に配置される各セクションを rompsec セクションに含めます。

[詳細説明]

- rompsec セクションに含めるテキスト・セクションを指定します。
- セクション *section* の内容とそのアドレス，およびサイズの情報を rompsec セクションに含めます。
- 本オプションは，text 属性，または const 属性を持つセクションに関するオプションです。
- *section* に指定可能なセクションは，text 属性，または const 属性を持つセクションです。
  これ以外の属性のセクションを指定した場合は，警告を出力して，処理を中止します。
- *section* がロード・モジュール・ファイル中に存在しない場合は，エラーを出力して，処理を中止します。
- *section* には，空白を使用することはできません。
- *section* を省略した場合は，エラーとなります。
- 本オプションを複数指定した場合は，指定した順に rompsec セクションに含めます。
- 内蔵命令 RAM 搭載のデバイス・ファイルを指定してリンクされた入力ファイルに対して，本オプションで特定のセクションを指定した場合，指定しなかった内蔵命令 RAM に配置したセクションは，rompsec セクションに入らないだけでなく，出力ファイル中からも削除します。

[使用例]

- セクション text1，および text2 を rompsec セクションに含めます。

```
>cx -CF3746 -Xrompsec_text=text1,text2 main.c
```

-Xrompsec_only

rompsec セクションのみを持つロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xrompsec_only
```

- 省略時解釈

  生成するロード・モジュール・ファイル中に text 属性を持つセクションも含めます。

[詳細説明]

- 生成するロード・モジュール・ファイル中に text 属性を持つセクションを含めずに，rompsec セクションのみを持つロード・モジュール・ファイルを生成します。

[使用例]

- rompsec セクションのみを持つロード・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xrompsec_only main.c
```

-Xromize_check_off

ROM 化時のエラー・チェックを省略します。

[指定形式]

```
-Xromize_check_off=string[,string]
```

- 省略時解釈

  rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行います。

  入力ファイル，および出力ファイルのアドレスの重複チェックを行います。

[詳細説明]

- ROM 化時のエラー・チェックを省略します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

rom_less	rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行いません。 本項目は，ROM レス・モード使用時に指定します。 また，シングルチップ・モード 1 選択時の内蔵 ROM オーバフローのチェックには対応していません。 本項目を指定することにより，内蔵 ROM オーバフローのチェックを無効にしてください。
address	入力ファイル，および出力ファイルのアドレスの重複チェックを行いません。

- *string* を省略した場合は，エラーとなります。

[使用例]

- rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行いません。

```
>cx -CF3746 -Xromize_check_off=rom_less main.c
```

ヘキサ出力制御

ヘキサ出力制御オプションには，次のものがあります。

- -Xhex
- -Xhex_only
- -Xhex_format
- -Xhex_fill
- -Xhex_section
- -Xhex_block_size
- -Xhex_offset
- -Xhex_null
- -Xhex_symtab
- -Xhex_rom_less
- -Xcrc
- -Xcrc_method

-Xhex

ヘキサ・ファイル名を指定します。

[指定形式]

```
-Xhex=file
```

- 省略時解釈
  ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .hex に置き換えた名前（ロード・モジュール・ファイル名の拡張子が .lmf でない場合は，ロード・モジュール・ファイル名に .hex を追加した名前）でヘキサ・ファイルを出力します。

[詳細説明]

- リンク終了後，ヘキサ・ファイルを *file* という名前で出力します。
- *file* にファイル名のみを指定した場合は，ロード・モジュール・ファイルと同じフォルダに，指定したファイル名で出力します。
- *file* を省略した場合は，エラーとなります。

[使用例]

- リンク終了後，ヘキサ・ファイル sample.hex を出力します。

```
>cx -CF3746 -Xhex=sample.hex main.c
```

-Xhex_only

ヘキサ出力のみ実行します。

[指定形式]

```
-Xhex_only[=file]
```

- 省略時解釈

  通常どおり，リンク，ROM 化，ヘキサ出力の順で処理を行います。

[詳細説明]

- 入力ファイルとして指定したロード・モジュール・ファイルから，ヘキサ・ファイルを生成して，*file* という名前で出力します。
- *file* にファイル名のみを指定した場合は，ロード・モジュール・ファイルと同じフォルダに，指定したファイル名で出力します。
- *file* を省略した場合は，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .hex に置き換えた名前で，ヘキサ・ファイルを出力します。
- 本オプションは，ロード・モジュール・ファイルの生成後，ヘキサ出力のみを行いたい場合に使用します。
  入力ファイルとして，ロード・モジュール・ファイルを指定してしない場合は，エラーとなります。

[使用例]

- ロード・モジュール・ファイル a.lmf から，ヘキサ・ファイル sample.hex を生成します。

```
>cx -CF3746 -Xhex_only=sample.hex a.lmf
```

-Xhex_format

出力するヘキサ・ファイルのフォーマットを指定します。

[指定形式]

```
-Xhex_format=format
```

- 省略時解釈

  出力するヘキサ・ファイルのフォーマットをインテル拡張ヘキサ・フォーマット（32 ビット・アドレス）とします（-Xhex_format=i の指定と同じです）。

[詳細説明]

- 出力するヘキサ・ファイルのフォーマットを指定します。
- *format* に指定可能なものを以下に示します。

I	インテル拡張ヘキサ・フォーマット（1M バイトまで）
i	インテル拡張ヘキサ・フォーマット（32 ビット・アドレス）（4G バイトまで）
S	モトローラ S タイプ・ヘキサ・フォーマット（スタンダード・アドレス）（16M バイトまで）
s	モトローラ S タイプ・ヘキサ・フォーマット（32 ビット・アドレス）（4G バイトまで）
T	拡張テクトロニクス・ヘキサ・フォーマット（4G バイトまで）

- *format* を省略した場合は，エラーとなります。
- -Xhex_format=T オプションを指定した場合，-Xhex_fill，-Xcrc，-Xcrc_method オプションは無効となります。

[使用例]

- 出力するヘキサ・ファイルのフォーマットをモトローラ S タイプ・ヘキサ・フォーマット（スタンダード・アドレス）とします。

```
>cx -CF3746 -Xhex_format=S main.c
```

-Xhex_fill

ヘキサ・ファイルの充てん処理を指定します。

[指定形式]

```
-Xhex_fill
-Xhex_fill=value
-Xhex_fill=value,start,size
-Xhex_fill=start,size
```

- 省略時解釈

  充てん処理を行いません。

[詳細説明]

- アドレス *start* からサイズ *size* で指定した領域のすべてのコードをヘキサ変換して，出力します。
  指定した領域のうち，未使用領域は *value* で充てんします。
- *value* に指定可能な値の範囲は，0x00 ～ 0xFFFF です。
- *value* は，16 進数で指定します。
- *value* は，1 バイト，または 2 バイト指定が可能です。
  指定した値が 2 桁，または 4 桁に満たない場合は，上位ビットを 0 で補てんします。
- *value* を省略した場合は，0xFF を指定したものとみなします。
- *start* に指定可能な値の範囲は，0x00 ～ 0xFFFFFFFF です。
  start が上記の範囲内であっても，セクションが存在しない場合は，エラーとなります。
- *size* に指定可能な値の範囲は，0x01 ～ 0x100000000 です。
- *start*，*size* は，16 進数で指定します。
- *start*，*size* の指定を省略した場合，デバイス・ファイルで定義された内蔵 ROM 領域のすべてのコードをヘキサ変換して，出力します。
- 本オプションは，-Xhex_format=T オプションと同時に指定することはできません。
- デバイス・ファイルの内蔵 ROM 領域情報を使用しない場合は，-Xhex_rom_less オプションと同時に指定します。
  その場合は，本オプションのパラメータ *start*，*size* を指定する必要があります。
- -Xcrc オプションを指定した場合，-Xcrc オプションのパラメータ *dst*，*start*，*end* を考慮して，最小アドレスを *start*，最大アドレスを *end* とし，最大アドレス＋1 ー最小アドレスを *size* とします。

［使用例］

- アドレス 0x1000 からサイズ 0x2000 分の領域のすべてのコードをヘキサ変換して，出力します。
  その領域のうち，未使用領域は 0x55 で充てんします。

```
>cx -CF3746 -Xhex_fill=0x55,0x1000,0x2000 main.c
```

-Xhex_section

指定したセクションのコードをヘキサ変換して，出力します。

[指定形式]

```
-Xhex_section=section[,section]...
```

- 省略時解釈

  NOBITS 以外のセクション・タイプとセクション属性 A を持つすべてのセクションをヘキサ変換して，出力します。

[詳細説明]

- セクション *section* のコードをヘキサ変換して，出力します。
- *section* が存在しない場合は，エラーとなります。
- *section* を省略した場合は，エラーとなります。
- 本オプションは，-Xhex_fill オプションと同時に指定することはできません。

[使用例]

- セクション sec のコードをヘキサ変換して，出力します。

```
>cx -CF3746 -Xhex_section=sec main.c
```

-Xhex_block_size

ブロック長の最大値を指定します。

[指定形式]

```
-Xhex_block_size=num
```

- 省略時解釈

  ヘキサ・フォーマットごとに定められたデフォルト値をブロック長の最大値とします。

[詳細説明]

- *num* に指定した数値をブロック長（インテル拡張ヘキサ・フォーマット，およびモトローラ S タイプ・ヘキサ・フォーマットの場合，1 データ・レコードで示されるコードのバイト数）の最大値とします。
- *num* に指定可能な値の範囲は，ヘキサ・フォーマットごとに異なります。

  各ヘキサ・フォーマットについて，*num* に指定可能な値の範囲を以下に示します。

  指定値が最小値に満たない場合は，警告を出力して，デフォルト値で補正します。

  指定値が最大値を越えている場合は，警告を出力して，最大値で補正します。

  0 を指定した場合は，エラーを出力します。

ヘキサ・フォーマット	*num* の範囲	デフォルト値
インテル拡張	1 ～ 255（0x01 ～ 0xFF）	32（0x20）
インテル拡張（32 ビット・アドレス）	1 ～ 255（0x01 ～ 0xFF）	32（0x20）
モトローラ S タイプ（スタンダード・アドレス）	1 ～ 251（0x01 ～ 0xFB）	80（0x50）
モトローラ S タイプ（32 ビット・アドレス）	1 ～ 250（0x01 ～ 0xFA）	80（0x50）
拡張テクトロニクス	16 ～ 255（0x10 ～ 0xFF）	255（0xFF）

- *num* を省略した場合は，エラーとなります。

[使用例]

- ブロック長の最大値を 255 とします。

```
>cx -CF3746 -Xhex_block_size=255 main.c
```

-Xhex_offset

出力するアドレスのオフセットを指定します。

[指定形式]

```
-Xhex_offset=num
```

- 省略時解釈

  出力するアドレスにオフセットがないものとみなします。

  0 番地から出力します。

[詳細説明]

- 出力するアドレスを，本来のアドレスにオフセットとして *num* を加えたアドレスとします。
- *num* に指定可能な値の範囲は，0x0 ～ 0xFFFFFFFE です。

  この範囲外の値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。

[使用例]

- 出力するアドレスのオフセットを 0x10000 とします。

```
>cx -CF3746 -Xhex_offset=0x10000 main.c
```

-Xhex_null

初期値なしデータのセクションに対して，セクションのサイズ分だけ null 文字を生成します。

[指定形式]

```
-Xhex_null
```

- 省略時解釈

  すべてのコードをヘキサ変換して，未使用領域は 0xFFFF で充てんして出力します。

[詳細説明]

- セクション・タイプ NOBITS とセクション属性 A を持つセクション（初期値の指定されていないデータに対するセクション，たとえば .bss セクション，および .sbss セクション）に対して，セクションのサイズ分だけ null 文字（¥0）を生成します。
- 本オプションは，-Xhex_fill オプションと同時に指定することはできません。

[使用例]

- セクション・タイプ NOBITS とセクション属性 A を持つセクションに対して，セクションのサイズ分だけ null 文字を生成します。

```
>cx -CF3746 -Xhex_null main.c
```

-Xhex_symtab

シンボル・テーブルを変換して，出力します。

[指定形式]

```
-Xhex_symtab=string
```

- 省略時解釈

  シンボル・テーブルを出力しません。

[詳細説明]

- シンボル・テーブルを変換して，出力します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

global	グローバル・シンボルのみ変換します。
all	ローカル・シンボルも含めて変換します。

- *string* を省略した場合は，エラーとなります。
- 本オプションは，-Xhex_format=T オプション指定時のみ有効です。
- 本オプションは，-Xhex_fill オプションと同時に指定することはできません。

[使用例]

- シンボル・テーブルを変換して，出力します。

```
>cx -CF3746 -Xhex_symtab -Xhex_format=T main.c
```

-Xhex_rom_less

ヘキサ・ファイルの充てん時に，内蔵 ROM 領域情報を使用しません。

[指定形式]

```
-Xhex_rom_less
```

- 省略時解釈

  本オプションを省略，かつ -Xhex_fill オプション指定時に，パラメータ *start*，*size* を省略した場合，デバイス・ファイルの内蔵 ROM 領域を充てん対象とします。

[詳細説明]

- ヘキサ・ファイルの充てん時に，デバイス・ファイルの内蔵 ROM 領域情報を使用しません。
- 本オプションは，-Xhex_fill オプション指定時に，デバイス・ファイルの内蔵 ROM 領域情報を使用しない場合に指定します。
- 本オプションは，-Xhex_fill オプションと同時に指定する必要があります。

  また，-Xhex_fill オプションのパラメータ *start*，*size* を指定する必要があります。

  -Xhex_fill オプション，およびパラメータ *start*，*size* を省略した場合は，エラーとなります。
- -Xhex_fill オプションのパラメータ *start*，*size* が内蔵 ROM 領域を越えても，警告を出力しません。

[使用例]

- ヘキサ・ファイルの充てん時に，デバイス・ファイルの内蔵 ROM 領域情報を使用しません。

```
>cx -CF3746 -Xhex_rom_less -Xhex_fill=0xff,0x00,1000 main.c
```

-Xcrc

CRC（Cyclic Redundancy Check）演算結果を出力します。

[指定形式]

```
-Xcrc=dst,start,end[,start,end]...
```

- 省略時解釈

  CRC 演算，および CRC 演算結果の出力を行いません。

[詳細説明]

- 開始アドレス *start* から終了アドレス *end* までの領域を対象に CRC 演算を行い，その結果を出力アドレス *dst* に出力します。

  dst に出力する演算結果は，0x0 ～ 0xFFFF の 16 ビット・データとなります。
- *start*，*end* は，1 組ずつ対にして複数指定することができます。
- *dst*，*start*，*end* に指定可能な値の範囲は，0x0 ～ 0xFFFFFFFF です。

  10 進数，または 16 進数で指定します。

  範囲外の値を指定した場合，および省略した場合は，エラーとなります。
- CRC 演算対象領域にある空き領域は充てんして演算を行うため，本オプションは -Xhex_fill オプションと同時に指定する必要があります。

  -Xhex_fill オプションを指定していない場合は，エラーとなります。

  本オプションと -Xhex_fill オプションの関係を以下に示します。

		-Xhex_fill=*f_value*,*f_start*,*f_size*	
		指定あり	指定なし
-Xcrc=*crc_dst*, *crc_start*, *crc_end*	指定あり	- CRC 演算時の充てん開始アドレスは *crc_start*，*f_start* のうち小さい方とし，充てん終了アドレスは *crc_end*，*f_start* ＋ *f_size* － 1 のうち大きい方とします。 充てん値は，-Xhex_fill オプションで指定した領域の範囲内は *f_value*，それ以外の CRC 演算範囲内は 0xFF とします。 - CRC 演算時の充てん値 0xFF は，ヘキサ・ファイルに出力しません。	- CRC 演算を行いません。 -Xhex_fill オプションを指定していないため，エラーとなります。
	指定なし	- CRC 演算を行いません。 - 充てんを行う領域，および充てん値は，-Xhex_fill オプションのパラメータに従います。	－

- *dst* が *start* から *end* までの領域内にある場合は，エラーとなります。
- *dst* が -Xhex_fill オプションで指定した領域の範囲外にある場合は，エラーとなります。

- *start* に指定した値が *end* に指定した値よりも大きい場合は，エラーとなります。
- *start*，*end* で指定した領域が他の *start*，*end* で指定した領域と重なる場合は，エラーとなります。

［使用例］

- 開始アドレス 0x100 ～ 0x200，0x300 ～ 0x400 を対象に CRC 演算を行い，その結果をアドレス 0x12345678 に出力します。

```
>cx -CF3746 -Xcrc=0x12345678,0x100,0x0200,0x300,0x400 main.c
```

-Xcrc_method

CRC 演算方法を指定します。

[指定形式]

```
-Xcrc_method=method[,value]
```

- 省略時解釈

  -Xcrc オプションを指定している場合は，演算方法 HIGH に従って CRC 演算を行います（-Xcrc_method=HIGH オプションの指定と同じです）。

  -Xcrc オプションを指定していない場合は，警告を出力して，本オプションを無視します。

[詳細説明]

- 演算方法 *method* に従って CRC 演算を行います。
- 本オプションは，-Xcrc オプション指定時のみ有効です。
- *method* には HIGH，または GENERAL を指定します。

  HIGH を指定すると高速 CRC（high-speed CRC）用の CRC-16-CCITT による演算結果を，GENERAL を指定すると汎用 CRC（general-purpose CRC）用の演算結果をそれぞれ得ることができます（「高速 CRC」，「汎用 CRC」の詳細については，デバイスのユーザーズ・マニュアルを参照してください）。

  method を省略した場合は，エラーとなります。
- *method* に GENERAL を指定した場合のみ，演算用の初期値 *value* を指定することができます。

  value を指定した場合は，初期値を *value* として CRC 演算を行います。

  value を省略した場合は，0 を指定したものとして演算を行います。
- *value* に指定可能な値の範囲は，0x0 ～ 0xFFFF です。

  10 進数，または 16 進数で指定します。

  範囲外の数値を指定した場合は，エラーとなります。
- *method* に HIGH を指定した場合は，CRC 演算用の初期値は 0 として演算を行います。

  method に HIGH を指定した場合に *value* を指定した場合は，エラーとなります。

[使用例]

- 演算方法を GENERAL，初期値を 0x1234 として CRC 演算を行います。

```
>cx -CF3746 -Xcrc=0x12345678,0x100,0x0200,0x300,0x400 -Xcrc_method=GENERAL,0x1234 main.c
```

マルチコア対応指定

マルチコア対応指定オプションには，次のものがあります。

- -Xmulti
- -Xmulti_link

-Xmulti

マルチコア用プログラムのサブプログラムの生成を指定します。

[指定形式]

```
-Xmulti=type
```

- 省略時解釈

  シングルコア用プログラムを生成します。

  なお，-C オプションでマルチコア CPU のデバイスを指定し，かつ，-Xmulti_link オプションを指定していない場合は，本オプションを省略することはできません。省略した場合は，エラーとなります。

[詳細説明]

- マルチコア用プログラムのサブプログラムを生成する際に，サブプログラムの種別を指定します。
- マルチコア CPU でないターゲットに対して本オプションを指定した場合，警告を出力して無視します。
- *type* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

pe*n*	コア *n* 用プログラムを生成します。 *n* に指定可能な値は，1 ～ターゲット CPU が持つコアの数までの整数です。 これ以外の値を指定した場合は，エラーとなります。
cmn	共通部を生成します。 変数アクセス，および関数アドレスの取得に対して r0 相対命令を生成します（ファイルの先頭に #pragma nopic 指令が記述しているのと同様に扱います）。#pragma pic 指令を記述している場合は，エラーとなります。 データに対して data，const，sconst 以外の再配置属性（sdata など）を持つセクションへの配置が指定されている場合は，エラーとなります。 また，-Xsdata=0 オプションを指定したものとみなし，-Xsdata オプションを指定している場合は，警告を出力して無視します。

- 本オプションを指定した場合，コンパイル時に *type* に応じたコードを生成し，サブプログラム内で解決可能な範囲でシンボル参照を解決し，リンク処理を行って再リンク可能なロード・モジュール・ファイルを生成します。

  このとき，デフォルトのスタートアップ・ルーチン，および標準ライブラリはリンクしません。

  また，ROM 化処理，およびヘキサ・ファイルの生成も行いません。

- 本オプションを指定した場合，以下のプリプロセッサ・マクロを自動的に設定します。

type	定義するプリプロセッサ・マクロ（値は 1）
pe*n*	__MULTI_CORE__，MULTI_PE*n*__
cmn	__MULTI_CORE__，MULTI_CMN__

- 本オプションを指定した場合に影響するオプションを以下に示します。
  - 暗黙に有効になるオプション

```
-Xno_startup，-Xno_stdlib，-Xno_romize，-Xrelinkable_object
```

  - 無効にするオプション

```
-l，-L，-Xstartup，-Xno_romize，-Xrompcrt，-Xrompsec_start，-Xrompsec_data，
-Xrompsec_text，-Xrompsec_only，-Xromize_check_off，-Xhex，-Xhex_only，-Xhex_format，
-Xhex_fill，-Xhex_section，-Xhex_block_size，-Xhex_offset，-Xhex_null，-Xhex_symtab，
-Xhex_rom_less，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，
-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata
```

- 各サブプログラム用のライブラリ・ファイルを作成する場合は，本オプションを -c オプションと同時に指定して，オブジェクト・モジュール・ファイルを生成します。

注意　異なる -Xmulti オプションを指定して生成したオブジェクト・モジュール・ファイルをまとめてライブラリ・ファイルを作成しないでください。

- 本オプションと -Xmuitl_link オプションを同時に指定した場合はエラーになります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ビルド設定］タブの［マルチコア］カテゴリの［対象コア番号］

［使用例］

- コア 1 用プログラムを生成します。

```
>cx -CF3515 -Xmulti=pe1 file_pe1_1.c file_pe1_2.c -ope1.lmf
```

-Xmulti_link

マルチコア用サブプログラムのリンクを指定します。

[指定形式]

```
-Xmulti_link
```

- 省略時解釈

  マルチコア用サブプログラムのリンクとみなしません。

  なお，-C オプションでマルチコア CPU のデバイスを指定し，かつ，-Xmulti オプションを指定していない場合は，本オプションを省略することはできません。省略した場合は，エラーとなります。

[詳細説明]

- マルチコア用プログラムを生成する際に，生成済みの各サブプログラムをリンクします。
- 本オプションを指定した場合，各サブプログラム，およびライブラリ・ファイル，オブジェクト・モジュール・ファイルを入力して，サブプログラム間のアドレス解決を行い，マルチコア用プログラムを生成します。

  このとき，マルチコア対応スタートアップ・ルーチン，およびライブラリをリンクします。

  また，ROM 化処理，およびヘキサ・ファイルの生成も同時に行います。
- 入力として，C ソース・ファイル，またはアセンブラ・ソース・ファイルを指定した場合は，エラーとなります。

  それ以外の種別のファイルを指定した場合，必要なファイルを指定していない場合，未解決の gp，ep，tp 相対シンボル参照が残っている場合は，リンク時にエラーとなります。
- 本オプションと -Xmuitl オプションを同時に指定した場合はエラーになります。
- 本オプションを指定した場合は，警告を出力して，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata オプションを無視します。

[使用例]

- マルチコア用サブプログラム pe1.lmf，pe2.lmf，cmn.lmf をリンクします。

```
>cx -CF3515 -Xmulti_link pe1.lmf pe2.lmf cmn.lmf -otarget.lmf -lmulti_lib
```

情報ファイル出力制御

情報ファイル出力制御オプションには，次のものがあります。

- -Xcref
- -Xno_cref
- -Xsfg
- -Xsfg_opt
- -Xsfg_size_tidata
- -Xsfg_size_tidata_byte
- -Xsfg_size_sidata
- -Xsfg_size_sedata
- -Xsfg_size_sdata

-Xcref

静的解析情報ファイルを出力します。

[指定形式]

```
-Xcref=file
```

- 省略時解釈

  静的解析情報ファイルを出力しません。

[詳細説明]

- 入力ファイルを静的に解析して，本製品の IDE，エディタなどが利用する情報を *file* に出力します。
- *file* の推奨拡張子は，.cref です。
- *file* がすでに存在する場合は，その内容を更新します。
  - 入力が C ソース・ファイルである場合

    file 中にその C ソース・ファイルに関する情報が存在する場合は，その情報を削除し，新しい内容に置き換えます。

    情報が存在しない場合は，*file* 中に追加します。
  - 入力がアセンブラ・ソース・ファイルである場合

    file 中にそのアセンブラ・ソース・ファイルに関する情報が存在する場合は，その情報を削除し，新しい内容に置き換えます。

    情報が存在しない場合は，*file* 中に追加します。

    file 中にそのアセンブラ・ソース・ファイルの元の C ソース・ファイルに関する情報が存在する場合は，その情報を削除します。

    元の C ソース・ファイルは，アセンブラ・ソース・ファイル中の .file 疑似命令から取得します。

- 入力がオブジェクト・モジュール・ファイルである場合

  file 中にそのオブジェクト・モジュール・ファイルのソース・ファイルに関する情報が存在する場合は，そのソース・ファイルに関する情報の中のリンク時に決定する情報のみを削除し，新しい内容に置き換えます。

  情報が存在しない場合は，何も出力しません。

- *file* を省略した場合は，エラーとなります。
- 入力ファイルが複数の場合，およびＣソース・ファイル，アセンブラ・ソース・ファイル，オブジェクト・モジュール・ファイルが混在している場合も，個々の入力ファイルについて上記のように更新し，*file* に出力します。
- ライブラリ・ファイルについての情報は出力しません。

  また，呼び出されない static 関数の情報（-Odelete_static_func=off 指定時を除く），および未使用のファイル内 static 変数の情報も出力しません。
- 本オプションは，同一アプリケーションのすべてのソース・ファイルに対して同じ指定を行うことを推奨します。

［使用例］

- 静的解析情報ファイルをファイル名 info.cref で出力します。

```
>cx -CF3746 -Xcref=info.cref main.c
```

-Xno_cref

静的解析情報ファイルの出力を抑止します。

[指定形式]

```
-Xno_cref
```

- 省略時解釈

  -Xcref オプションを指定している場合は，静的解析情報ファイルを出力します。

  -Xcref オプションを指定していない場合は，静的解析情報ファイルを出力しません。

[詳細説明]

- -Xcref オプションと同時に指定した場合に，指定順序によらず，-Xcref オプションを無効にします。

  また，-Xcref オプションで指定したファイルが存在する場合は，そのファイルを削除します。
- IDE からの起動時には常に -Xcref オプションが付加されますが，ビルド／リビルド時間短縮などの目的でそれを抑止したい場合に，本オプションを指定します。
- 本オプションを指定した場合には，本製品のシンボル・ファイル・ジェネレータ等，一部の機能が使えなくなります。

[使用例]

- 静的解析情報ファイルの出力を抑止します。

```
>cx -CF3746 -Xcref=info.cref main.c -Xno_cref
```

-Xsfg

シンボル情報ファイルを生成します。

[指定形式]

```
-Xsfg[=file]
```

- 省略時解釈

  シンボル情報ファイルを生成しません。

[詳細説明]

- C ソース・ファイルの静的解析結果に基づいて，シンボル情報ファイル *file* を生成します。
- *file* の推奨拡張子は，.sfg です。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* を省略した場合は，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .sfg に置き換えた名前で出力します。
- CubeSuite+ 用情報ファイルの情報を利用するため，本オプションは，-Xcref オプションと同時に指定する必要があります。
- 本オプションは，同一アプリケーションのすべてのソース・ファイルに対して同じ指定を行うことを推奨します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［シンボル情報ファイルを出力する］，［シンボル情報ファイル出力フォルダ］，［シンボル情報ファイル名］

[使用例]

- シンボル情報ファイルをファイル名 symbol.sfg で出力します。

```
>cx -CF3746 -Xsfg=symbol.sfg -Xcref=info.cref main.c
```

-Xsfg_opt

最適な配置情報を出力します。

[指定形式]

```
-Xsfg_opt
```

- 省略時解釈

  変数を利用頻度の高い順にソートして，配置情報を出力します。

[詳細説明]

- .tidata.byte，.tidata.word，.sidata，.sedata，.sdata セクションのサイズ内に配置できるよう，セクション単位で変数の利用頻度の高い順に，最適な配置情報を出力します。
- 各セクションのサイズは，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata オプションで指定することができます。

  これらのオプションを指定しない場合，各セクションのサイズは以下のようになります。

セクション名	サイズ（バイト）
.tidata.byte	128
.tidata.word	128
.sidata	32512
.sedata	32768
.sdata	65536

- 本オプションは，-Xsfg オプション指定時のみ有効です。

  -Xsfg オプションを指定しない場合は，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［最適な配置情報を出力する］

[使用例]

- .tidata.byte，.tidata.word，.sidata，.sedata，.sdata セクションのサイズ内に配置できるよう，セクション単位で変数の利用頻度の高い順に，最適な配置情報を出力します。

```
>cx -CF3746 -Xsfg_opt -Xsfg main.c
```

-Xsfg_size_tidata

.tidata セクションのサイズを指定します。

[指定形式]

```
-Xsfg_size_tidata=num
```

- 省略時解釈

  .tidata セクションに変数を配置するサイズの上限を 128 バイトに制限して，変数の利用頻度を算出します。

[詳細説明]

- .tidata セクションに変数を配置するサイズの上限を *num* バイトに制限して，変数の利用頻度を算出します。
- *num* には，0 ～ 256 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- .tidata セクションは，デフォルトで 256 バイトであり，内部的に .tidata.byte セクション（デフォルトで最大 128 バイト）と .tidata.word セクションに分かれています。

  .tidata セクション＝ .tidata.byte セクション＋ .tidata.word セクション

  .tidata.byte セクションと tidata.word セクションの合計サイズが 256 バイトに達するまで変数を選択して，セクションに割り当てます。

  ただし，.tidata.byte セクションのサイズが指定サイズ，または 256 バイトに達したところで，選択を終了します。
- .tidata.byte セクションのサイズより小さい値を指定した場合は，.tidata セクションは .tidata.byte セクションと同じサイズであるとみなします。

  したがって，この場合の .tidata.word セクションのサイズは 0 となります。
- 本オプションは，-Xsfg_opt オプション指定時のみ有効です。

  -Xsfg_opt オプションを指定しない場合は，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［.tidata セクションのサイズ］

[使用例]

- .tidata セクションに変数を配置するサイズの上限を 128 バイトに制限して，変数の利用頻度を算出します。

```
>cx -CF3746 -Xsfg_size_tidata=128 -Xsfg_opt main.c
```

-Xsfg_size_tidata_byte

.tidata.byte セクションのサイズを指定します。

[指定形式]

```
-Xsfg_size_tidata_byte=num
```

- 省略時解釈

  .tidata.byte セクションに変数を配置するサイズの上限を 128 バイトに制限して，変数の利用頻度を算出します。

[詳細説明]

- .tidata.byte セクションに変数を配置するサイズの上限を *num* バイトに制限して，変数の利用頻度を算出します。
- *num* には，0 ～ 128 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- 本オプションは，-Xsfg_opt オプション指定時のみ有効です。

  -Xsfg_opt オプションを指定しない場合は，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［.tidata.byte セクションのサイズ］

[使用例]

- .tidata.byte セクションに変数を配置するサイズの上限を 64 バイトに制限して，変数の利用頻度を算出します。

```
>cx -CF3746 -Xsfg_size_tidata_byte=64 -Xsfg_opt main.c
```

-Xsfg_size_sidata

.sidata セクションのサイズを指定します。

［指定形式］

```
-Xsfg_size_sidata=num
```

- 省略時解釈

  .sidata セクションに変数を配置するサイズの上限を 32512 バイトに制限して，変数の利用頻度を算出します。

［詳細説明］

- .sidata セクションに変数を配置するサイズの上限を *num* バイトに制限して，変数の利用頻度を算出します。
- *num* には，0 ～ 32512 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- 本オプションは，-Xsfg_opt オプション指定時のみ有効です。

  -Xsfg_opt オプションを指定しない場合は，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［.sidata セクションのサイズ］

［使用例］

- .sidata セクションに変数を配置するサイズの上限を 32000 バイトに制限して，変数の利用頻度を算出します。

```
>cx -CF3746 -Xsfg_size_sidata=32000 -Xsfg_opt main.c
```

-Xsfg_size_sedata

.sedata セクションのサイズを指定します。

[指定形式]

```
-Xsfg_size_sedata=num
```

- 省略時解釈

  .sedata セクションに変数を配置するサイズの上限を 32768 バイトに制限して，変数の利用頻度を算出します。

[詳細説明]

- .sedata セクションに変数を配置するサイズの上限を *num* バイトに制限して，変数の利用頻度を算出します。
- *num* には，0 ～ 32768 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- 本オプションは，-Xsfg_opt オプション指定時のみ有効です。

  -Xsfg_opt オプションを指定しない場合は，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［.sedata セクションのサイズ］

[使用例]

- .sedata セクションに変数を配置するサイズの上限を 16384 バイトに制限して，変数の利用頻度を算出します。

```
>cx -CF3746 -Xsfg_size_sedata=16384 -Xsfg_opt main.c
```

-Xsfg_size_sdata

.sdata セクションのサイズを指定します。

[指定形式]

```
-Xsfg_size_sdata=num
```

- 省略時解釈

  .sdata セクションに変数を配置するサイズの上限を 65536 バイトに制限して，変数の利用頻度を算出します。

[詳細説明]

- .sdata セクションに変数を配置するサイズの上限を *num* バイトに制限して，変数の利用頻度を算出します。
- *num* には，0 ～ 65536 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- 本オプションは，-Xsfg_opt オプション指定時のみ有効です。

  -Xsfg_opt オプションを指定しない場合は，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［シンボル情報］カテゴリの［.sdata セクションのサイズ］

[使用例]

- .sdata セクションに変数を配置するサイズの上限を 32768 バイトに制限して，変数の利用頻度を算出します。

```
>cx -CF3746 -Xsfg_size_sdata=32768 -Xsfg_opt main.c
```

エラー出力制御

エラー出力制御オプションには，次のものがあります。

- -Xerror_file

-Xerror_file

エラー・メッセージをファイルに出力します。

[指定形式]

```
-Xerror_file=file
```

- 省略時解釈

  エラー・メッセージを標準エラー出力のみに出力します。

[詳細説明]

- エラー・メッセージを標準エラー出力，およびファイル *file* に出力します。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［エラー出力］カテゴリの［エラー・メッセージ・ファイルを出力する］，［エラー・メッセージ・ファイル出力フォルダ］，［エラー・メッセージ・ファイル名］
  - ［個別コンパイル・オプション］タブの［エラー出力］カテゴリの［エラー・メッセージ・ファイルを出力する］，［エラー・メッセージ・ファイル出力フォルダ］，［エラー・メッセージ・ファイル名］

[使用例]

- エラー・メッセージを標準エラー出力，およびファイル err に出力します。

```
>cx -CF3746 -Xerror_file=err main.c
```

警告メッセージ出力制御

警告メッセージ出力制御オプションには，次のものがあります。

- -Xwarning
- -Xno_warning

-Xwarning

指定した警告メッセージを出力します。

[指定形式]

```
-Xwarning=num[,num]...
-Xwarning=num1-num2
```

- 省略時解釈

  重大な警告メッセージを出力します。

[詳細説明]

- 指定した警告メッセージを出力します。
- *num*，*num1*，*num2* には，エラー番号を指定します。

  存在しないエラー番号を指定した場合は，無視します。
- *num*，または *num1*，および *num2* を省略した場合は，エラーとなります。
- *num1-num2* の形式で指定すると，その範囲に含まれるエラー番号を指定したものとみなします。
- 本オプションで指定するエラー番号は，W に続く 7 桁の数字のうち，下位 5 桁です。

  エラー番号については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル メッセージ編」を参照してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［警告メッセージ］カテゴリの［必ず表示させる警告メッセージ］
  - ［個別コンパイル・オプション］タブの［警告メッセージ］カテゴリの［必ず表示させる警告メッセージ］

[使用例]

- 警告メッセージ W0566002 を出力します。

```
>cx -CF3746 -Xwarning=66002 main.c
```

-Xno_warning

指定した警告メッセージの出力を抑止します。

[指定形式]

```
-Xno_warning=num[,num]...
-Xno_warning=num1-num2
```

- 省略時解釈

  重大な警告メッセージを出力します。

[詳細説明]

- 指定した警告メッセージの出力を抑止します。
- *num*，*num1*，*num2* には，エラー番号を指定します。

  存在しないエラー番号を指定した場合は，無視します。
- *num*，または *num1*，および *num2* を省略した場合は，エラーとなります。
- *num1-num2* の形式で指定すると，その範囲に含まれるエラー番号を指定したものとみなします。
- 本オプションで指定するエラー番号は，W に続く 7 桁の数字のうち，下位 5 桁です。

  エラー番号については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル メッセージ編」を参照してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［警告メッセージ］カテゴリの［表示させない警告メッセージ］
  - ［個別コンパイル・オプション］タブの［警告メッセージ］カテゴリの［表示させない警告メッセージ］

[使用例]

- 警告メッセージ W0566002 の出力を抑止します。

```
>cx -CF3746 -Xno_warning=66002 main.c
```

フェーズ個別オプション指定

フェーズ個別オプション指定オプションには，次のものがあります。

- -Xasm_option
- -Xlk_option
- -Xopt_option

-Xasm_option

アセンブル対象ファイルを指定します。

[指定形式]

```
-Xasm_option=file[,file]...
```

- 省略時解釈

  .asm ／ .s のみをアセンブル対象として認識します。

[詳細説明]

- アセンブラ・ソース・ファイルとして認識しないファイル（.asm ／ .s 以外）がアセンブル対象となるよう，cx に指示します。
- *file* には，アセンブルを行いたいファイルを指定します。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。

[使用例]

- ファイル assemble.test のアセンブルを行います。

```
>cx -CF3746 -Xasm_option=assemble.test
```

-Xlk_option

リンク対象ファイルを指定します。

[指定形式]

```
-Xlk_option=file[,file]...
```

- 省略時解釈

  .obj ／ .lib ／ .lmf のみをリンク対象として認識します。

[詳細説明]

- リンク対象として認識しないファイル（.obj ／ .lib ／ .lmf 以外）がリンク対象となるよう，cx に指示します。
- *file* には，リンクを行いたいファイルを指定します。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。

[使用例]

- ファイル link.test のリンクを行います。

```
>cx -CF3746 -Xlk_option=link.test
```

-Xopt_option

共通最適化部オプションを指定します。

[指定形式]

```
-Xopt_option=arg[,arg]...
```

- 省略時解釈

  コマンド・ラインで指定したオプションは，すべて cx のドライバが解釈します。

[詳細説明]

- *arg* を共通最適化部オプションとして，共通最適化部に渡します。
- *arg* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

-Ogc	最適化レベルがデフォルトの場合に，スタック・サイズの増加を抑止します。

- *arg* を省略した場合は，エラーとなります。

[使用例]

- 最適化レベルがデフォルトの場合に，スタック・サイズの増加を抑止します。

```
>cx -CF3746 -Xopt_option=-Ogc main.c
```

コマンド・ファイル指定

コマンド・ファイル指定オプションには，次のものがあります。

- @

@

コマンド・ファイルを指定します。

[指定形式]

```
@file
```

- 省略時解釈

  コマンド・ラインで指定したオプション，およびファイル名のみを認識します。

[詳細説明]

- *file* をコマンド・ファイルとして扱います。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- コマンド・ファイルについての詳細は，「(2) コマンド・ファイルによる操作方法」を参照してください。

[使用例]

- command をコマンド・ファイルとして扱います。

```
>cx @command
```

(2) アセンブル・オプション

アセンブル・フェーズのオプションの分類と説明を以下に示します。

表 B—4 アセンブル・オプション

分類	オプション	説明
バージョン／ヘルプ表示指定	-V	cx のバージョン情報を表示します。
	-h	cx のオプションの説明を表示します。
出力ファイル指定	-o	出力ファイル名を指定します。
	-Xobj_path	アセンブル途中に生成されるオブジェクト・モジュール・ファイルの保存先を指定します。
	-Xprn_path	アセンブル・リスト・ファイルの保存先を指定します。
ソース・デバッグ制御	-g	ソース・デバッグ用の情報を出力します。
デバイス指定	-C	ターゲット・デバイスを指定します。
	-Xcommon	デバイス共通のオブジェクト・モジュール・ファイルを生成することを指定します。
	-Xdev_path	デバイス・ファイルを検索するフォルダを指定します。
	-Xprogrammable_io	プログラマブル周辺 I/O レジスタの開始アドレスを指定します。
プリプロセッサ制御	-D	アセンブラ・シンボルを定義します。
	-U	-D オプションによるアセンブラ・シンボルの定義を解除します。
	-I	インクルード・ファイルを検索するフォルダを指定します。
日本語／中国語文字列制御	-Xcharacter_set	日本語／中国語の文字コードを指定します。
生成コード制御	-Xsdata	.sdata セクション，または .sbss セクションに配置するデータの最大サイズを指定します。
アセンブラ制御指定	-Xasm_far_jump【V850E2】【V850E2V3】	アセンブラ・ソース・ファイルに対して，far jump の出力を制御します。
リンク制御	-Xflash	フラッシュ領域側のロード・モジュール・ファイルを生成します。
マルチコア対応指定	-Xmulti	マルチコア用プログラムのサブプログラムの生成を指定します。
	-Xmulti_link	マルチコア用サブプログラムのリンクを指定します。
エラー出力制御	-Xerror_file	エラー・メッセージをファイルに出力します。
警告メッセージ出力制御	-Xwarning	指定した警告メッセージを出力します。
	-Xno_warning	指定した警告メッセージの出力を抑止します。
コマンド・ファイル指定	@	コマンド・ファイルを指定します。

表 B―5　オプション説明でのマーク

【V850E2】	V850E2 コアで命令セット・アーキテクチャが V850E2 であるデバイス専用のオプション
【V850E2V3】	V850E2 コアで命令セット・アーキテクチャが V850E2V3 であるデバイス専用のオプション

バージョン／ヘルプ表示指定

バージョン／ヘルプ表示指定オプションには，次のものがあります。

- -V
- -h

-V

cx のバージョン情報を表示します。

[指定形式]

```
-V
```

- 省略時解釈

  cx のバージョン情報を表示せずに，アセンブルを行います。

[詳細説明]

- cx のバージョン情報を標準エラー出力に出力します。

  アセンブルは行いません。

[使用例]

- cx のバージョン情報を標準エラー出力に出力します。

```
>cx -CF3746 -V
```

-h

cx のオプションの説明を表示します。

[指定形式]

```
-h
```

- 省略時解釈

  cx のオプションの説明を表示しません。

[詳細説明]

- cx のオプションの説明を標準エラー出力に出力します。

  コンパイルは行いません。

- cx のオプションの説明を標準エラー出力に出力します。

  アセンブルは行いません。

[使用例]

- cx のオプションの説明を標準エラー出力に出力します。

```
>cx -CF3746 -h
```

出力ファイル指定

出力ファイル指定オプションには，次のものがあります。

- -o
- -Xobj_path
- -Xprn_path

-o

出力ファイル名を指定します。

［指定形式］

```
-ofile
```

- 省略時解釈

  カレント・フォルダにファイルを出力します。

  - -c オプションと同時に指定した場合

    出力オブジェクト・モジュール・ファイル名は，ソース・ファイル名の拡張子を .obj に置き換えたものとなります。

  - 上記以外の場合

    出力ロード・モジュール・ファイル名は，a.lmf となります。

［詳細説明］

- 出力ファイル名を *file* に指定します。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- -c オプションと同時に指定することにより処理を中断した場合にも，本オプションは有効となります。
  - -c オプションと同時に指定した場合

    file には，オブジェクト・モジュール・ファイル名を指定したものとみなします。

  - 上記以外の場合

    file には，ロード・モジュール・ファイル名を指定したものとみなします。
- 出力ファイルが複数の場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［個別アセンブル・オプション］タブの［出力ファイル］カテゴリの［オブジェクト・モジュール・ファイル名］

［使用例］

- ロード・モジュール・ファイルをファイル名 sample.lmf で出力します。

```
>cx -CF3746 -osample.lmf main.asm
```

-Xobj_path

アセンブル途中に生成されるオブジェクト・モジュール・ファイルの保存先を指定します。

[指定形式]

```
-Xobj_path[=path]
```

- 省略時解釈

カレント・フォルダに，ソース・ファイル名の拡張子を .obj で置き換えたファイル名でオブジェクト・モジュール・ファイルを保存します。

ただし，入力として，1 つのソース・ファイルを指定し，かつ -c オプションを指定しない場合は，オブジェクト・モジュール・ファイルは保存しません。

[詳細説明]

- アセンブル途中に生成されるオブジェクト・モジュール・ファイルを *path* に保存します。
  - *path* に存在するファイル名を指定した場合

    出力するオブジェクト・モジュール・ファイルが 1 つの場合は，*path* という名前で保存します。

    出力するオブジェクト・モジュール・ファイルが複数の場合は，エラーとなります。
  - *path* に存在するフォルダ名を指定した場合

    path に，ソース・ファイル名の拡張子を .obj で置き換えたファイル名でオブジェクト・モジュール・ファイルを保存します。
  - *path* に指定した名前のフォルダ，およびファイルが存在しない場合

    エラーとなります。
  - =*path* を省略した場合

    カレント・フォルダに，ソース・ファイル名の拡張子を .obj で置き換えたファイル名でオブジェクト・モジュール・ファイルを保存します。
- ソース・ファイルとして同じ名前のファイル（異なるフォルダにある場合を含む）を複数指定した場合は，警告を出力して，最後に指定したソース・ファイルに対するオブジェクト・モジュール・ファイルのみを保存します。

[使用例]

- アセンブル途中に生成されるオブジェクト・モジュール・ファイルをファイル名 sample.obj で保存します。

```
>cx -CF3746 -Xobj_path=sample.obj main.asm
```

-Xprn_path

アセンブル・リスト・ファイルの保存先を指定します。

[指定形式]

```
-Xprn_path[=path]
```

- 省略時解釈

  アセンブル・リスト・ファイルを出力しません。

[詳細説明]

- アセンブル時にアセンブル・リスト・ファイルを出力して，*path* に保存します。
  - *path* に存在するファイル名を指定した場合

    出力するアセンブル・リスト・ファイルが 1 つの場合は，*path* という名前で保存します。

    出力するアセンブル・リスト・ファイルが複数の場合は，エラーとなります。
  - *path* に存在するフォルダ名を指定した場合

    path に，ソース・ファイル名の拡張子を .prn で置き換えたファイル名でアセンブル・リスト・ファイルを保存します。
  - *path* に指定した名前のフォルダ，およびファイルが存在しない場合

    エラーとなります。
  - =*path* を省略した場合

    カレント・フォルダに，ソース・ファイル名の拡張子を .prn で置き換えたファイル名でアセンブル・リスト・ファイルを保存します。
- ソース・ファイルとして同じ名前のファイル（異なるフォルダにある場合を含む）を複数指定した場合は，警告を出力して，最後に指定したソース・ファイルに対するアセンブル・リスト・ファイルのみを保存します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［アセンブル・オプション］タブの［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイルを出力する］，［アセンブル・リスト・ファイル出力フォルダ］
  - ［個別アセンブル・オプション］タブの［アセンブル・リスト］カテゴリの［アセンブル・リスト・ファイルを出力する］，［アセンブル・リスト・ファイル出力フォルダ］

[使用例]

- アセンブル時に出力されるアセンブル・リスト・ファイルをファイル名 sample.prn で保存します。

```
>cx -CF3746 -Xprn_path=sample.prn main.asm
```

ソース・デバッグ制御

ソース・デバッグ制御オプションには，次のものがあります。

- -g

-g

ソース・デバッグ用の情報を出力します。

[指定形式]

```
-g
```

- 省略時解釈

  ソース・デバッグ用の情報を出力しません。

[詳細説明]

- ソース・デバッグ用の情報を出力ファイル中に出力します。
- 本オプションを指定することにより，ソース・デバッグが可能となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［アセンブル・オプション］タブの［デバッグ情報］カテゴリの［デバッグ情報を生成する］
  - ［個別アセンブル・オプション］タブの［デバッグ情報］カテゴリの［デバッグ情報を生成する］

[使用例]

- ソース・デバッグ用の情報を出力ファイル中に出力します。

```
>cx -CF3746 -g main.asm
```

デバイス指定

デバイス指定オプションには，次のものがあります。

- -C
- -Xcommon
- -Xdev_path
- -Xprogrammable_io

-C

ターゲット・デバイスを指定します。

[指定形式]

```
-Cdevice
```

- 省略時解釈

  -Xcommon オプションを指定している場合は，その指定内容に従います。

  それ以外の場合は，エラーとなります（-V ／ -h ／ -P オプション指定時を除く）。

  なお，リンクを行う場合は，エラーとなります。

[詳細説明]

- ターゲット・デバイスを指定します。
- *device* に指定可能なデバイス品種については，各デバイス・ファイルのユーザーズ・マニュアルを参照してください。
- *device* が存在しない（対応するデバイス・ファイルがない）場合は，エラーとなります。
- *device* を省略した場合は，エラーとなります。
- リンクを行う場合は，本オプションを省略することはできません。

[使用例]

- ターゲット・デバイスとして，μPD70F3746 を指定します。

```
>cx -CF3746 main.asm
```

-Xcommon

デバイス共通のオブジェクト・モジュール・ファイルを生成することを指定します。

[指定形式]

```
-Xcommon=series
```

- 省略時解釈

  -C オプションを指定している場合は，その指定内容に従います。

  それ以外の場合は，エラーとなります。

[詳細説明]

- デバイス共通のオブジェクト・モジュール・ファイルを生成することを指定します。
- 本オプションを指定した場合，ターゲットの命令セット・アーキテクチャの命令のみを使用し，また，命令セット・アーキテクチャに対応した共通のマジック・ナンバ *series* をオブジェクト・モジュール・ファイルに埋め込みます。
- *series* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

v850e	命令セット・アーキテクチャが V850E より上位（V850E，V850E2，V850E2V3）の品種をターゲット・デバイスとして指定したものとのリンクが可能です。
v850e2	命令セット・アーキテクチャが V850E2 より上位（V850E2，および V850E2V3）の品種をターゲット・デバイスとして指定したものとのリンクが可能です。
v850e2v3	命令セット・アーキテクチャが V850E2V3 である品種をターゲット・デバイスとして指定したものとのリンクが可能です。 ターゲット・デバイスの命令セット・アーキテクチャが V850E2V3 である場合は，その性能を引き出すために，本項目を指定することを推奨します。

- *series* を省略した場合は，エラーとなります。
- -C オプションと同時に指定した場合の処理は，以下のようになります。

-Xcommon の *series* パラメータ	-C の指定デバイス		
	V850E	V850E2	V850E2V3
v850e	通常処理	-Xcommon=v850e2 に置換（警告を出力）	-Xcommon=v850e2v3 に置換（警告を出力）
v850e2	通常処理（警告出力）	通常処理	-Xcommon=v850e2v3 に置換（警告を出力）
v850e2v3	通常処理（警告出力）	通常処理（警告出力）	通常処理

-C オプションで指定したデバイスの命令セット・アーキテクチャが本オプションで指定したものである場合，両方のオプションを処理します。
-C オプションで指定したデバイスの命令セット・アーキテクチャが本オプションで指定したものより下位（V850E2V3 ＞ V850E2 ＞ V850E/ES）の場合，警告を出力して，両方のオプションを処理します。
-C オプションで指定したデバイスの命令セット・アーキテクチャが本オプションで指定したものより上位の場合，警告を出力して，本オプションの *series* パラメータを -C オプションで指定したデバイスの命令セット・アーキテクチャに置換して処理します。

［使用例］

- 生成するオブジェクト・モジュール・ファイルに，命令セット・アーキテクチャが V850E より上位の品種に共通のマジック・ナンバを埋め込みます。

```
>cx -Xcommon=v850e -c main.asm
```

-Xdev_path

デバイス・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Xdev_path=path
```

- 省略時解釈

  デバイス・ファイルを標準のデバイス・ファイル・フォルダから検索します。

[詳細説明]

- デバイス・ファイルを *path* で指定したフォルダから検索します。
- *path* で指定したフォルダが存在しない場合，または -C オプションで指定したデバイス・ファイルが *path* で指定したフォルダに見つからない場合は，警告を出力して，標準のデバイス・ファイル・フォルダ[注]を検索します。それでも見つからない場合は，エラーとなります。

  注　以下の順に検索します。

  【V850E2V3】

  1. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850E2 ¥ Devicefile
  2. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850 ¥ Devicefile
  3. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device_Custom ¥ Devicefile

  【V850E/V850E2】

  1. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850 ¥ Devicefile
  2. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device_Custom ¥ Devicefile

- *path* を省略した場合は，エラーとなります。

[使用例]

- デバイス・ファイルをフォルダ D:¥dev から検索します。

```
>cx -CF3746 -Xdev_path=D:¥dev main.asm
```

-Xprogrammable_io

プログラマブル周辺 I/O レジスタの開始アドレスを指定します。

[指定形式]

```
-Xprogrammable_io=num
```

- 省略時解釈

  プログラマブル周辺 I/O レジスタの開始アドレスは，各デバイスで既定されているアドレスとなります。

[詳細説明]

- アドレス変更可能なプログラマブル周辺 I/O レジスタの開始アドレス *num* を指定します。
- ターゲット・デバイスがアドレス変更可能なプログラマブル周辺 I/O レジスタを持つ場合（V850E/IA1 など），コンパイル（アセンブル）時に，その開始アドレスを確定させる必要があります。

  そこで，本オプションにより，プログラマブル周辺 I/O レジスタの開始アドレスを指定します。
- プログラマブル周辺 I/O レジスタの開始アドレスの下位ビット（ビット数はデバイスにより異なります）は固定であるため，*num* は下位ビットが 0 となるように指定します。

  下位ビットに 0 以外を指定した場合は，警告を出力して，下位ビットを切り捨てます。
- *num* に不正な値，または各デバイスで指定可能な範囲を越える値を指定した場合は，警告を出力して，本オプションを無視します。
- *num* に指定する値は，同一アプリケーションのすべてのファイルで同じにする必要があります。

  ただし，プログラマブル周辺 I/O レジスタを使用しないファイルを個別にコンパイルする場合は，本オプションを指定する必要はありません。

  また，プログラマブル周辺 I/O レジスタ機能を持たないデバイスの場合，および V850E/V850E2/V850E2V3 共通としてアセンブルする場合に本オプションを指定すると，警告を出力して，本オプションを無視します。
- *num* を省略した場合は，エラーとなります。
- プログラマブル周辺 I/O レジスタを使用するためには，BPC レジスタに値を指定する必要がありますが，本オプションはプログラマブル周辺 I/O レジスタのアドレスをコンパイル（アセンブル）時に確定するためのものであり，BPC レジスタに実際に値を反映させるものではありません。

  動作のためには，別途，スタートアップ・ルーチンなどで BPC レジスタに値を指定する必要があります。

  例えば，V850E/IA1 の場合は，以下の例のように指定コードを追加してください。

```
USEBPC  set 0x8000                          ; プログラマブル周辺 I/O レジスタのアクセス許可（固定）
PIOADDR set 0x38d000                        ; -Xprogrammable_io オプションで渡したのと同じ値
mov     (USEBPC or (PIOADDR shr 14)), r13   ; PIOADDR（=num）は自動的に渡されます
st.h    r13, BPC
```

［使用例］

- ターゲット・デバイスが V850E/IA1 の場合，プログラマブル周辺 I/O レジスタの開始アドレスを 0x38d0000 に指定します。

```
>cx -CF3116 -Xprogrammable_io=0x38d0000 main.asm
```

プリプロセッサ制御

プリプロセッサ制御オプションには，次のものがあります。

- -D
- -U
- -I

-D

アセンブラ・シンボルを定義します。

[指定形式]

```
-Dname[=def][name[=def]]...
```

- 省略時解釈

  なし

[詳細説明]

- アセンブラ・シンボルとして *name* を定義します。
- アセンブラ・ソース・プログラムの前に，*name* .SET *def* を記述するのと同等です。
- *name* が，アセンブラ・シンボルには使用可能であるがプリプロセッサ・マクロには使用できない文字（@，.，~）を含む場合，警告を出力してアセンブラ・シンボルとしてのみ定義します。
- *name* を省略した場合は，エラーとなります。
- =*def* を省略した場合，*def* は 1 とみなします。
- 本オプションは，複数指定が可能です。
- 同じアセンブラ・シンボルに対して，本オプションと -U オプションを同時に指定した場合は，あとから指定したものが有効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（アセンブル）］カテゴリの［定義マクロ］
  - ［アセンブル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］
  - ［個別アセンブル・オプション］タブの［プリプロセス］カテゴリの［定義マクロ］

[使用例]

- アセンブラ・シンボルとして sample=256 を定義します。

```
>cx -CF3746 -Dsample=256 main.asm
```

-U

-D オプションによるアセンブラ・シンボルの定義を解除します。

[指定形式]

```
-Uname[,name]...
```

- 省略時解釈

  なし

[詳細説明]

- -D オプションによるアセンブラ・シンボル *name* の定義を解除します。
- *name* を省略した場合は，エラーとなります。
- 本オプションでは，*name* .SET *def* の記述による定義は解除できません。
- 本オプションは，複数指定が可能です。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［アセンブル・オプション］タブの［プリプロセス］カテゴリの［定義解除マクロ］
  - ［個別アセンブル・オプション］タブの［プリプロセス］カテゴリの［定義解除マクロ］

[使用例]

- -D オプションによるアセンブラ・シンボル test の定義を解除します。

```
>cx -CF3746 -Utest main.asm
```

-I

インクルード・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Ipath[,path]...
```

- 省略時解釈
  インクルード・ファイルを 標準インクルード・ファイル・フォルダからのみ検索します。

[詳細説明]

- アセンブラ制御命令 $INCLUDE ／ $BINCLUDE で読み込むインクルード・ファイルを検索するフォルダを *path* に指定します。
  インクルード・ファイルの検索は，以下の順番で行います。

(1) -I オプションで指定したフォルダ

(2) ソース・ファイルのあるフォルダ

(3) 元の C ソース・ファイルのあるフォルダ

(4) カレント・フォルダ

- *path* が存在しない場合は，警告を出力します。
- *path* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（アセンブル）］カテゴリの［追加のインクルード・パス］，［システム・インクルード・パス］
  - ［アセンブル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］，［システム・インクルード・パス］
  - ［個別アセンブル・オプション］タブの［プリプロセス］カテゴリの［追加のインクルード・パス］，［ビルド・ツールに指定した全体インクルード・パスも使用する］

[使用例]

- インクルード・ファイルをカレント・フォルダ，フォルダ D:¥include，標準フォルダの順で検索します。

```
>cx -CF3746 -ID:¥include main.asm
```

日本語／中国語文字列制御

日本語／中国語文字列制御オプションには，次のものがあります。

- -Xcharacter_set

-Xcharacter_set

日本語／中国語の文字コードを指定します。

[指定形式]

```
-Xcharacter_set=code
```

- 省略時解釈

  日本語の文字コードを SJIS として扱います。

[詳細説明]

- ソース・ファイル中の日本語／中国語のコメント，文字列に対して，使用する文字コードを指定します。
- *code* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

  なお，ソース・ファイル中で使用している文字コードと異なるものを指定した場合，動作は保証されません。

none	日本語／中国語の文字コードを処理しません
euc_jp	EUC（日本語）
sjis	SJIS
utf8	UTF-8
big5	繁体字中国語
gb2312	簡体字中国語

- *code* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［アセンブル・オプション］タブの［文字コード］カテゴリの［文字コード］
  - ［個別アセンブル・オプション］タブの［文字コード］カテゴリの［文字コード］

[使用例]

- ソース・ファイル中の日本語のコメント，文字列に対して，使用する文字コードに EUC を指定します。

```
>cx -CF3746 -Xcharacter_set=euc_jp main.asm
```

生成コード制御

生成コード制御オプションには，次のものがあります。

- -Xsdata

-Xsdata

.sdata セクション，または .sbss セクションに配置するデータの最大サイズを指定します。

[指定形式]

```
-Xsdata=num
```

- 省略時解釈

  すべてのデータを .sdata ／ .sbss セクションに配置します。

  ただし，const 修飾された静的変数は，.const セクションに割り付けます。

[詳細説明]

- *num* バイト以下のデータを .sdata セクション，または .sbss セクションに配置します。
- #pragma section 指令で .sdata ／ .sbss セクションを指定したデータは，そのサイズに関係なく .sdata ／ .sbss セクションに配置します。
- 不完全型の配列（ファイル内でサイズが不明な配列）に対しては，本オプションは適用しません。
- *num* には，0 ～ 65535 を指定します。

  この範囲を越える値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- *num* に指定する値の目安は，-Xsdata_info オプションで出力することができます。
- ソース・ファイルごとに異なるオプションを指定すると，変数の配置，および参照方法が異なるコードを生成して，リンク時にエラー，または警告を出力することがあります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［アセンブル・オプション］タブの［出力コード］カテゴリの［sdata/sbss セクションに配置するデータ長の上限値 (バイト)］

[使用例]

- 16 バイト以下のデータを .sdata セクション，または .sbss セクションに配置します。

```
>cx -CF3746 -Xsdata=16 main.asm
```

アセンブラ制御指定

アセンブラ制御指定オプションには，次のものがあります。

- -Xasm_far_jump【V850E2】【V850E2V3】

-Xasm_far_jump【V850E2】【V850E2V3】

アセンブラ・ソース・ファイルに対して，far jump の出力を制御します。

[指定形式]

```
-Xasm_far_jump
```

- 省略時解釈

  jarl，または jr 命令としてアセンブルを行います。

[詳細説明]

- アセンブラ・ソース・ファイルに対して，ソース中に記述されたすべての jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。
- 個別の命令ごとに制御したい場合は，ソース中で jarl22 ／ jarl32，jr22 ／ jarl32 のように明記します。
- 本オプションは，jump 命令には影響しません。
- C ソース・ファイルに対して本オプションを指定した場合，警告を出力せずに無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［アセンブル・オプション］タブの［出力コード］カテゴリの［32 ビット分岐命令を使用する］
  - ［個別アセンブル・オプション］タブの［出力コード］カテゴリの［32 ビット分岐命令を使用する］

[使用例]

- アセンブラ・ソース中に記述されたすべての jarl，および jr 命令を jarl32，および jr32 命令とみなしてアセンブルを行います。

```
>cx -CF3746 -Xasm_far_jump main.asm
```

リンク制御

リンク制御オプションには，次のものがあります。

- -Xflash

-Xflash

フラッシュ領域側のロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xflash=file
```

- 省略時解釈

  ブートーフラッシュ再リンク機能を使用する場合は，ブート領域側のロード・モジュール・ファイルを生成します。

  ブートーフラッシュ再リンク機能を使用しない場合は，通常のリンク処理を行います。

[詳細説明]

- ブートーフラッシュ再リンク機能を使用する場合に，フラッシュ領域側のロード・モジュール・ファイルを生成します。

  その際，ブート領域側のロード・モジュール・ファイル *file* のシンボル情報を参照して，リンク処理を行います。
- *file* には，ブートーフラッシュ再リンク機能を使用して生成したブート領域側のロード・モジュール・ファイルを指定します。

  ここで指定するロード・モジュール・ファイルは，ROM 化処理前のもの（-Xno_romize，または -Xlink_output オプションを指定して生成したもの）である必要があります。
- *file* に存在しないファイルを指定した場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，-Xflash_ext_table オプションと同時に指定する必要があります。

[使用例]

- フラッシュ領域側の分岐テーブル先頭アドレスが 0x200 番地にあるものとして，ブート領域側のロード・モジュール・ファイル boot.lmf を生成します。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xlink_output=boot.lmf boot1.asm boot2.asm -
Xlink_directive=boot.dir
```

0x200 番地に分岐テーブルを生成して，フラッシュ領域側のロード・モジュール・ファイル flash.lmf を生成します。

その際，ブート領域側のロード・モジュール・ファイル boot.lmf のシンボル情報を参照して，リンク処理を行います。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xflash=boot.lmf -oflash.lmf flash1.asm flash2.asm  -
Xlink_directive=flash.dir
```

マルチコア対応指定

マルチコア対応指定オプションには，次のものがあります。

- -Xmulti
- -Xmulti_link

-Xmulti

マルチコア用プログラムのサブプログラムの生成を指定します。

[指定形式]

```
-Xmulti=type
```

- 省略時解釈

  シングルコア用プログラムを生成します。

  なお，-C オプションでマルチコア CPU のデバイスを指定し，かつ，-Xmulti_link オプションを指定していない場合は，本オプションを省略することはできません。省略した場合は，エラーとなります。

[詳細説明]

- マルチコア用プログラムのサブプログラムを生成する際に，サブプログラムの種別を指定します。
- マルチコア CPU でないターゲットに対して本オプションを指定した場合，警告を出力して無視します。
- *type* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

pe*n*	コア *n* 用プログラムを生成します。 *n* に指定可能な値は，1 ～ターゲット CPU が持つコアの数までの整数です。 これ以外の値を指定した場合は，エラーとなります。
cmn	共通部を生成します。 変数アクセス，および関数アドレスの取得に対して r0 相対命令を生成します（ファイルの先頭に #pragma nopic 指令が記述しているのと同様に扱います）。#pragma pic 指令を記述している場合は，エラーとなります。 データに対して data，const，sconst 以外の再配置属性（sdata など）を持つセクションへの配置が指定されている場合は，エラーとなります。 また，-Xsdata=0 オプションを指定したものとみなし，-Xsdata オプションを指定している場合は，警告を出力して無視します。

- 本オプションを指定した場合，以下のプリプロセッサ・マクロを自動的に設定します。

type	定義するプリプロセッサ・マクロ（値は 1）
pe*n*	__MULTI_CORE__，MULTI_PE*n*__

type	定義するプリプロセッサ・マクロ（値は 1）
cmn	__MULTI_CORE__，MULTI_CMN__

- 本オプションを指定した場合，アセンブル時にセクション名を自動的に変換します（セクション名がソース中で明示的に指定されている場合を除く）。
  すなわち，デフォルトのセクション名の後に type に応じた接尾子（.pen，または .cmn）を追加します。
  また，各オブジェクト・モジュール・ファイル中に，そのファイルで定義されているセクションと *type* の対応を示す情報のセクションを出力します。
- 本オプションを指定した場合に影響するオプションを以下に示します。
  - 暗黙に有効になるオプション

```
-Xno_startup，-Xno_stdlib，-Xno_romize，-Xrelinkable_object
```

  - 無効にするオプション

```
-l，-L，-Xstartup，-Xno_romize，-Xrompcrt，-Xrompsec_start，-Xrompsec_data，
-Xrompsec_text，-Xrompsec_only，-Xromize_check_off，-Xhex，-Xhex_only，-Xhex_format，
-Xhex_fill，-Xhex_section，-Xhex_block_size，-Xhex_offset，-Xhex_null，-Xhex_symtab，
-Xhex_rom_less，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，
-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata
```

- 各サブプログラム用のライブラリ・ファイルを作成する場合は，本オプションを -c オプションと同時に指定して，オブジェクト・モジュール・ファイルを生成します。

注意　異なる -Xmulti オプションを指定して生成したオブジェクト・モジュール・ファイルをまとめてライブラリ・ファイルを作成しないでください。

- 本オプションと -Xmulti_link オプションを同時に指定した場合はエラーになります（エラー処理は，上位層のドライバで行います）。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ビルド設定］タブの［マルチコア］カテゴリの［対象コア番号］

［使用例］

- コア 1 用プログラムを生成します。

```
>cx -CF3515 -Xmulti=pe1 file_pe1_1.asm file_pe1_2.asm -ope1.lmf
```

-Xmulti_link

マルチコア用サブプログラムのリンクを指定します。

[指定形式]

```
-Xmulti_link
```

- 省略時解釈

  マルチコア用サブプログラムのリンクとみなしません。

  なお，-C オプションでマルチコア CPU のデバイスを指定し，かつ，-Xmulti オプションを指定していない場合は，本オプションを省略することはできません。省略した場合は，エラーとなります。

[詳細説明]

- マルチコア用プログラムを生成する際に，生成済みの各サブプログラムをリンクします。
- 本オプションを指定した場合，各サブプログラム，およびライブラリ・ファイル，オブジェクト・モジュール・ファイルを入力して，サブプログラム間のアドレス解決を行い，マルチコア用プログラムを生成します。

  このとき，マルチコア対応スタートアップ・ルーチン，およびライブラリをリンクします。

  また，ROM 化処理，およびヘキサ・ファイルの生成も同時に行います。
- 入力として，C ソース・ファイル，またはアセンブラ・ソース・ファイルを指定した場合は，エラーとなります。

  それ以外の種別のファイルを指定した場合，必要なファイルを指定していない場合，未解決の gp，ep，tp 相対シンボル参照が残っている場合は，リンク時にエラーとなります。
- 本オプションと -Xmulti オプションを同時に指定した場合はエラーになります（エラー処理は，上位層のドライバで行います）。
- 本オプションを指定した場合は，警告を出力して，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata オプションを無視します。

[使用例]

- マルチコア用サブプログラム pe1.lmf，pe2.lmf，cmn.lmf をリンクします。

```
>cx -CF3515 -Xmulti_link pe1.lmf pe2.lmf cmn.lmf -otarget.lmf -lmulti_lib
```

エラー出力制御

エラー出力制御オプションには，次のものがあります。

- -Xerror_file

-Xerror_file

エラー・メッセージをファイルに出力します。

[指定形式]

```
-Xerror_file=file
```

- 省略時解釈

  エラー・メッセージを標準エラー出力のみに出力します。

[詳細説明]

- エラー・メッセージを標準エラー出力，およびファイル *file* に出力します。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［個別アセンブル・オプション］タブの［エラー出力］カテゴリの［エラー・メッセージ・ファイルを出力する］，［エラー・メッセージ・ファイル出力フォルダ］，［エラー・メッセージ・ファイル名］

[使用例]

- エラー・メッセージを標準エラー出力，およびファイル err に出力します。

```
>cx -CF3746 -Xerror_file=err main.asm
```

警告メッセージ出力制御

警告メッセージ出力制御オプションには，次のものがあります。

- -Xwarning
- -Xno_warning

-Xwarning

指定した警告メッセージを出力します。

[指定形式]

```
-Xwarning=num[,num]...
-Xwarning=num1-num2
```

- 省略時解釈

  重大な警告メッセージを出力します。

[詳細説明]

- 指定した警告メッセージを出力します。
- *num*，*num1*，*num2* には，エラー番号を指定します。

  存在しないエラー番号を指定した場合は，無視します。
- *num*，または *num1*，および *num2* を省略した場合は，エラーとなります。
- *num1-num2* の形式で指定すると，その範囲に含まれるエラー番号を指定したものとみなします。
- 本オプションで指定するエラー番号は，W に続く 7 桁の数字のうち，下位 5 桁です。

  エラー番号については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル メッセージ編」を参照してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［個別アセンブル・オプション］タブの［警告メッセージ］カテゴリの［必ず表示させる警告メッセージ］

[使用例]

- 警告メッセージ W0566002 を出力します。

```
>cx -CF3746 -Xwarning=66002 main.asm
```

-Xno_warning

指定した警告メッセージの出力を抑止します。

[指定形式]

```
-Xno_warning=num[,num]...
-Xno_warning=num1-num2
```

- 省略時解釈

  重大な警告メッセージを出力します。

[詳細説明]

- 指定した警告メッセージの出力を抑止します。
- *num*，*num1*，*num2* には，エラー番号を指定します。

  存在しないエラー番号を指定した場合は，無視します。
- *num*，または *num1*，および *num2* を省略した場合は，エラーとなります。
- *num1-num2* の形式で指定すると，その範囲に含まれるエラー番号を指定したものとみなします。
- 本オプションで指定するエラー番号は，W に続く 7 桁の数字のうち，下位 5 桁です。

  エラー番号については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル メッセージ編」を参照してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［個別アセンブル・オプション］タブの［警告メッセージ］カテゴリの［表示させない警告メッセージ］

[使用例]

- 警告メッセージ W0566002 の出力を抑止します。

```
>cx -CF3746 -Xno_warning=66002 main.asm
```

コマンド・ファイル指定

コマンド・ファイル指定オプションには，次のものがあります。

- @

@

コマンド・ファイルを指定します。

[指定形式]

```
@file
```

- 省略時解釈

  コマンド・ラインで指定したオプション，およびファイル名のみを認識します。

[詳細説明]

- *file* をコマンド・ファイルとして扱います。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- コマンド・ファイルについての詳細は，「(2) コマンド・ファイルによる操作方法」を参照してください。

[使用例]

- command をコマンド・ファイルとして扱います。

```
>cx @command
```

(3) リンク・オプション

リンク・フェーズのオプションの分類と説明を以下に示します。

表 B—6 リンク・オプション

分類	オプション	説明
バージョン／ヘルプ表示指定	-V	cx のバージョン情報を表示します。
	-h	cx のオプションの説明を表示します。
出力ファイル指定	-o	出力ファイル名を指定します。
	-Xtemp_path	作業用フォルダを指定します。
ソース・デバッグ制御	-g	ソース・デバッグ用の情報を出力します。
デバイス指定	-C	ターゲット・デバイスを指定します。
	-Xdev_path	デバイス・ファイルを検索するフォルダを指定します。
ライブラリ・リンク制御	-l	リンク時に使用するライブラリ・ファイルを指定します。
	-L	ライブラリ・ファイルを検索するフォルダを指定します。
	-Xno_stdlib	標準ライブラリのリンクを抑止します。
リンク・ディレクティブ・ファイル指定	-Xlink_directive	リンク・ディレクティブ・ファイルを指定します。
セキュリティ ID 制御	-Xsecurity_id	セキュリティ ID を設定します。
ユーザ・オプション・バイト制御	-Xoption_byte	ユーザ・オプション・バイトを設定します。
リンク処理強制続行指定	-Xforce_link	内部 ROM/RAM のオーバフロー時に，リンク処理を続行します。
エントリ・ポイント・アドレス指定	-Xentry_address	エントリ・ポイント・アドレスを指定します。
リンク・マップ・ファイル出力指定	-Xmap	リンク・マップ・ファイルを出力します。
シンボル情報出力指定	-Xsymbol_dump	リンク・マップ・ファイルにシンボル情報を出力します。
生成オブジェクト・モジュール・ファイル制御	-Xrelinkable_object	再配置可能なオブジェクト・モジュール・ファイルを生成します。
レジスタ・モード混在チェック制御	-Xreg_mode	指定したレジスタ・モードに対して，混在チェックを行います。
	-Xregmode_info	異なるレジスタ・モードが混在している場合，詳細情報を出力します。

分類	オプション	説明
デバイス共通オブジェクト混在チェック指定	-Xcommon	生成するデバイス共通オブジェクト・モジュール・ファイルと -C オプションで指定したデバイスの混在チェックを行います。
sdata/sbss 情報出力指定	-Xsdata_info	-Xsdata オプションのパラメータに対して，目安として用いることのできる情報を標準出力に出力します。
2 パス・モード・リンク指定	-Xtwo_pass_link	2 パス・モードでリンクを行います。
リロケーション解決エラー処理制御	-Xignore_address_error	リンク時のリロケーション処理において，不正箇所があった場合，リンク処理を続行します。
シンボル多重定義エラー出力指定	-Xmultiple_symbol	多重定義されたすべての外部シンボルに対してエラー・メッセージを出力します。
リンク時チェック抑止指定	-Xlink_check_off	リンク時のチェックを抑止します。
充てん値指定	-Xalign_fill	アライン・ホールの充てん値を指定します。
ライブラリ・ファイル再スキャン指定	-Xrescan	-I オプションで指定したライブラリ・ファイルの再スキャンを行います。
デバッグ情報セクション出力抑止指定	-Xstrip	デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。
ブート＿フラッシュ再リンク機能制御	-Xflash_ext_table	ブート＿フラッシュ再リンク機能用の分岐テーブル先頭アドレス値を指定します。
	-Xflash	フラッシュ領域側のロード・モジュール・ファイルを生成します。
ROM 化処理前ロード・モジュール・ファイル保存指定	-Xlink_output	ROM 化処理前のロード・モジュール・ファイルを保存します。
マルチコア対応指定	-Xmulti	マルチコア用プログラムのサブプログラムの生成を指定します。
	-Xmulti_link	マルチコア用サブプログラムのリンクを指定します。
エラー出力制御	-Xerror_file	エラー・メッセージをファイルに出力します。
警告メッセージ出力制御	-Xwarning	指定した警告メッセージを出力します。
	-Xno_warning	指定した警告メッセージの出力を抑止します。
フェーズ個別オプション指定	-Xlk_option	リンク対象ファイルを指定します。
コマンド・ファイル指定	@	コマンド・ファイルを指定します。

バージョン／ヘルプ表示指定

バージョン／ヘルプ表示指定オプションには，次のものがあります。

- -V
- -h

-V

cx のバージョン情報を表示します。

[指定形式]

```
-V
```

- 省略時解釈

  cx のバージョン情報を表示せずに，リンクを行います。

[詳細説明]

- cx のバージョン情報を標準エラー出力に出力します。

  リンクは行いません。

[使用例]

- cx のバージョン情報を標準エラー出力に出力します。

```
>cx -CF3746 -V
```

-h

cx のオプションの説明を表示します。

[指定形式]

```
-help
```

- 省略時解釈

  cx のオプションの説明を表示しません。

[詳細説明]

- cx のオプションの説明を標準エラー出力に出力します。

  リンクは行いません。

[使用例]

- cx のオプションの説明を標準エラー出力に出力します。

```
>cx -CF3746 -h
```

出力ファイル指定

出力ファイル指定オプションには，次のものがあります。

- -o
- -Xtemp_path

-o

出力ファイル名を指定します。

[指定形式]

```
-ofile
```

- 省略時解釈

  カレント・フォルダにファイルを出力します。

  出力ロード・モジュール・ファイル名は，a.lmf となります。

[詳細説明]

- 出力ファイル名を *file* に指定します。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* には，ロード・モジュール・ファイル名を指定したものとみなします。
- 出力ファイルが複数の場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（リンク）］カテゴリの［出力フォルダ］，［出力ファイル名］
  - ［リンク・オプション］タブの［出力ファイル］カテゴリの［出力フォルダ］，［出力ファイル名］

[使用例]

- ロード・モジュール・ファイルをファイル名 sample.lmf で出力します。

```
>cx -CF3746 -osample.lmf main.obj
```

-Xtemp_path

作業用フォルダを指定します。

[指定形式]

```
-Xtemp_path=path
```

- 省略時解釈

以下の順番で作業用フォルダを決定します。

（1）環境変数 TEMP で指定したフォルダ

（2）環境変数 TMP で指定したフォルダ

（3）カレント・フォルダ

[詳細説明]

- 内部的に用いるテンポラリ・ファイルを生成する作業用フォルダを *path* に指定します。
- *path* が存在しない場合は，警告を出力して，以下の順番で作業用フォルダを決定します。

（1）環境変数 TEMP で指定したフォルダ

（2）環境変数 TMP で指定したフォルダ

（3）カレント・フォルダ

- *path* を省略した場合は，エラーとなります。

[使用例]

- 作業用フォルダをフォルダ D:￥tmp に指定します。

```
>cx -CF3746 -Xtemp_path=D:￥tmp main.obj
```

ソース・デバッグ制御

ソース・デバッグ制御オプションには，次のものがあります。

- -g

-g

ソース・デバッグ用の情報を出力します。

[指定形式]

```
-g
```

- 省略時解釈

  ソース・デバッグ用の情報を出力しません。

[詳細説明]

- ソース・デバッグ用の情報を出力ファイル中に出力します。
- 本オプションを指定することにより，ソース・レベルでのデバッグが可能となります。

[使用例]

- ソース・デバッグ用の情報を出力ファイル中に出力します。

```
>cx -CF3746 -g main.obj
```

デバイス指定

デバイス指定オプションには，次のものがあります。

- -C
- -Xdev_path

-C

ターゲット・デバイスを指定します。

[指定形式]

```
-Cdevice
```

- 省略時解釈

  -Xcommon オプションを指定している場合は，その指定内容に従います。

  それ以外の場合は，エラーとなります（-V ／ -h ／ -P オプション指定時を除く）。

  なお，リンクを行う場合は，エラーとなります。

[詳細説明]

- ターゲット・デバイスを指定します。
- *device* に指定可能なデバイス品種については，各デバイス・ファイルのユーザーズ・マニュアルを参照してください。
- *device* が存在しない（対応するデバイス・ファイルがない）場合は，エラーとなります。
- *device* を省略した場合は，エラーとなります。
- リンクを行う場合は，本オプションを省略することはできません。

[使用例]

- ターゲット・デバイスとして，μPD70F3746 を指定します。

```
>cx -CF3746 main.obj
```

-Xdev_path

デバイス・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Xdev_path=path
```

- 省略時解釈

  デバイス・ファイルを標準のデバイス・ファイル・フォルダから検索します。

[詳細説明]

- デバイス・ファイルを *path* で指定したフォルダから検索します。
- *path* で指定したフォルダが存在しない場合，または -C オプションで指定したデバイス・ファイルが *path* で指定したフォルダに見つからない場合は，警告を出力して，標準のデバイス・ファイル・フォルダ[注]を検索します。それでも見つからない場合は，エラーとなります。

  注　以下の順に検索します。

  【V850E2V3】

  1. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850E2 ¥ Devicefile
  2. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850 ¥ Devicefile
  3. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device_Custom ¥ Devicefile

  【V850E/V850E2】

  1. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device ¥ V850 ¥ Devicefile
  2. *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ .. ¥ .. ¥ Device_Custom ¥ Devicefile

- *path* を省略した場合は，エラーとなります。

[使用例]

- デバイス・ファイルをフォルダ D:¥dev から検索します。

```
>cx -CF3746 -Xdev_path=D:¥dev main.obj
```

ライブラリ・リンク制御

ライブラリ・リンク制御オプションには，次のものがあります。

- -l
- -L
- -Xno_stdlib

-l

リンク時に使用するライブラリ・ファイルを指定します。

[指定形式]

```
-lstring[,string]...
```

- 省略時解釈

  標準ライブラリ，数学ライブラリ，および標準スタートアップ・ルーチンのみをリンクします。

  -Xno_romize オプションを指定しない場合は，ROM 化用領域確保コード・ファイルもリンクします。

[詳細説明]

- リンク時に使用するライブラリ・ファイル lib*string*.lib を指定します。
- cx は，すべてのオブジェクト・モジュール・ファイルのリンク後，未解決な外部シンボルの参照を解決する際に，ライブラリ・ファイル（lib*string*.lib）を参照します。
- *string* を省略した場合は，エラーとなります。
- -L オプションと同時に指定した場合は，-L オプションで指定したフォルダからライブラリ・ファイルを検索します。

  -L オプションを指定しない場合は，標準フォルダ[注]から検索します。
- 本オプションで指定したライブラリ・ファイルが見つからない場合は，メッセージを出力せずに，正常にリンク処理を続行します。
- 複数のライブラリ・ファイルを指定した場合は，指定した順番に検索します。
- cx は，本オプションで指定したライブラリ以外にも，標準ライブラリ（libc.lib），および標準スタートアップ・ルーチン（cstart.obj）を自動的にリンクします。

  -Xno_romize オプションを指定しない場合は，ROM 化用領域確保コード・ファイル（rompcrt.obj）も自動的にリンクします。

  これを抑止するには，-Xno_stdlib オプション，および -Xno_startup オプションを使用します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（リンク）］カテゴリの［使用するライブラリ・ファイル］
  - ［リンク・オプション］タブの［ライブラリ］カテゴリの［使用するライブラリ・ファイル］，［システム・ライブラリ・ファイル］

注 *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ lib ¥ 850e

［使用例］

- リンク時に使用するライブラリ・ファイル libsmp.lib を指定します。

```
>cx -CF3746 -lsmp main.obj
```

-L

ライブラリ・ファイルを検索するフォルダを指定します。

[指定形式]

```
-Lpath[,path]...
```

- 省略時解釈

  ライブラリを標準ライブラリ・フォルダからのみ検索します。

[詳細説明]

- ライブラリをフォルダ *path*，標準ライブラリ・フォルダ[注]の順に検索します。
- *path* が存在しない場合は，警告を出力します。
- *path* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［よく使うオプション（リンク）］カテゴリの［追加のライブラリ・パス］
  - ［リンク・オプション］タブの［ライブラリ］カテゴリの［追加のライブラリ・パス］［システム・ライブラリ・パス］

注 *本製品のインストール・フォルダ*¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ lib ¥ 850e

なお，V850E2V3 で FPU 対応数学ライブラリをリンクする場合は，以下も標準ライブラリ・フォルダとします。

本製品のインストール・フォルダ¥ CubeSuite+ ¥ CX ¥ V*x.xx* ¥ lib ¥ 850e2v3f

[使用例]

- ライブラリをフォルダ lib，標準フォルダの順に検索します。

```
>cx -CF3746 -Llib main.obj
```

-Xno_stdlib

標準ライブラリのリンクを抑止します。

[指定形式]

```
-Xno_stdlib
```

- 省略時解釈

  標準ライブラリをリンクします。

[詳細説明]

- 標準ライブラリをリンクしません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［ライブラリ］カテゴリの［標準ライブラリをリンクする］

[使用例]

- 標準ライブラリをリンクしません。

```
>cx -CF3746 -Xno_stdlib main.obj
```

リンク・ディレクティブ・ファイル指定

リンク・ディレクティブ・ファイル指定オプションには，次のものがあります。

- -Xlink_directive

-Xlink_directive

リンク・ディレクティブ・ファイルを指定します。

[指定形式]

```
-Xlink_directive=file
```

- 省略時解釈

  デフォルトのリンク・ディレクティブ・ファイルを使用します。

[詳細説明]

- リンク・ディレクティブ・ファイル *file* 内のリンク・ディレクティブに従ってリンクを行います。
- *file* の推奨拡張子は，.dir です。
- *file* を省略した場合は，エラーとなります。
- 本オプションを複数指定した場合は，あとから指定したオプションが有効となり，先に指定したオプションは無視します。
- リンク・ディレクティブ・ファイルについての詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）」を参照してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［入力ファイル］カテゴリの［使用するリンク・ディレクティブ・ファイル］

[使用例]

- リンク・ディレクティブ・ファイル link.dir 内のリンク・ディレクティブに従ってリンクを行います。

```
>cx -CF3746 -Xlink_directive=link.dir main.obj
```

セキュリティ ID 制御

セキュリティ ID 制御オプションには，次のものがあります。

- -Xsecurity_id

-Xsecurity_id

セキュリティ ID を設定します。

[指定形式]

```
-Xsecurity_id=id
-Xsecurity_id=none
```

- 省略時解釈

  セキュリティ ID として 0xFFFFFFFFFFFFFFFFFFFF を自動生成します。

  セキュリティ ID 機能を持たないデバイスを指定した場合は，何も行いません。

[詳細説明]

- セキュリティ ID を 70H ～ 79H 番地に設定します。
- *id* には，10 バイト以内の 16 進数を指定します。

  指定した値が 10 バイトに満たない場合は，上位ビットを 0 で補てんします。
- *id* にデバイスで対応している桁数を越えた値を指定した場合は，エラーとなります。

  ただし，配置アドレス，および *id* のデータサイズは，デバイスにより異なります。
- none を指定した場合は，セキュリティ ID の自動生成を抑止します。
- *id* ，または none を省略した場合は，エラーとなります。
- セキュリティ ID の設定は，アセンブラ・ソース中で以下のように記述することでも可能です。

```
.CSEG   SECUR_ID
.DB     0x11
    :
.DB     0xAA
```

- 本オプションとアセンブラ・ソースでの設定が重なった場合は，警告を出力して，本オプションを優先します。
- セキュリティ ID 機能を持たないデバイスに対して，本オプションを指定した場合は，エラーとなります。
- -Xmulti オプションと同時に指定した場合は，警告を出力して，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［デバイス］カテゴリの［セキュリティ ID を設定する］，［セキュリティ ID］

［使用例］

- セキュリティ ID として 0x112233445566778899AA（0x70 番地に 0x11，0x71 番地に 0x22，0x72 番地に 0x33，0x73 番地に 0x44，0x74 番地に 0x55，0x75 番地に 0x66，0x76 番地に 0x77，0x77 番地に 0x88，0x78 番地に 0x99，0x79 番地に 0xAA）を設定します。

```
>cx -CF3746 -Xsecurity_id=0x112233445566778899AA file1.obj file2.obj
```

ユーザ・オプション・バイト制御

ユーザ・オプション・バイト制御オプションには，次のものがあります。

- -Xoption_byte

-Xoption_byte

ユーザ・オプション・バイトを設定します。

[指定形式]

```
-Xoption_byte=none
```

- 省略時解釈

  デバイス・ファイルの初期値をユーザ・オプション・バイト領域に埋め込みます。

  オプション・バイト機能をサポートしていないデバイスに対しては，何も行いません。

[詳細説明]

- ユーザ・オプション・バイト領域の生成を抑止します。
- ユーザ・オプション・バイトは，7AH ～ 7FH 番地に設定されます。

  7AH ～ 7FH 番地のユーザ・オプション・バイトは，ソース中に記述することも可能です。

  ただし，配置アドレスは，デバイスにより異なります。
- none を省略した場合は，エラーとなります。
- ユーザ・オプション・バイトの設定は，アセンブラ・ソース中で以下のように記述することにより，任意に設定することも可能です。

```
.CSEG OPT_BYTE
.DB 0x11
.DB 0x22
.DB 0x33
.DB 0x44
.DB 0x55
.DB 0x66
```

- オプション・バイト機能をサポートしていないデバイスに対して，本オプションを指定した場合は，本オプションを無視します。
- ユーザ・オプション・バイトをユーザ空間に設定することができないデバイスに対して，本オプションを指定した場合は，エラーとなります。
- -Xmulti オプションと同時に指定した場合は，警告を出力して，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［ユーザ・オプション・バイトを設定する］

［使用例］

- ユーザ・オプション・バイト領域の生成を抑止します。

```
>cx -CF3746 -Xoption_byte=none file1.c file2.c
```

リンク処理強制続行指定

リンク処理強制続行指定オプションには，次のものがあります。

- -Xforce_link

-Xforce_link

内部 ROM/RAM のオーバフロー時に，リンク処理を続行します。

[指定形式]

```
-Xforce_link
```

- 省略時解釈

  内部 ROM/RAM のオーバフロー時に，エラーを出力して，リンク処理を終了します。

[詳細説明]

- 内部 ROM/RAM のオーバフロー時に，警告を出力して，リンク処理を続行します。
- 出力ファイルの生成は行いません。
- 超過したサイズを標準エラー出力に出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［強制リンクを行う］

[使用例]

- 内部 ROM/RAM のオーバフロー時に，警告を出力して，リンク処理を続行します。

```
>cx -CF3746 -Xforce_link main.obj
```

エントリ・ポイント・アドレス指定

エントリ・ポイント・アドレス指定オプションには，次のものがあります。

- -Xentry_address

-Xentry_address

エントリ・ポイント・アドレスを指定します。

[指定形式]

```
-Xentry_address=symbol
```

- 省略時解釈

  生成するロード・モジュール・ファイルのエントリ・ポイント・アドレス値を，以下の規則で決定します。

  - シンボル __start が存在する場合は，その値
  - __start が存在しない場合は，生成するロード・モジュール・ファイル内の最下位に割り付けられた text 属性のセクションの先頭アドレス
  - text 属性セクションが存在しない場合は，0

[詳細説明]

- シンボル *symbol* の値を生成するロード・モジュール・ファイルのエントリ・ポイント・アドレス値（ヘキサ変換時に使用します）とします。
- *symbol* が存在しない場合は，エラーとなります。
- *symbol* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［エントリ・シンボル］

[使用例]

- シンボル __my_start の値を生成するロード・モジュール・ファイルのエントリ・ポイント・アドレス値とします。

```
>cx -CF3746 -Xentry_address=__my_start main.obj
```

リンク・マップ・ファイル出力指定

リンク・マップ・ファイル出力指定オプションには，次のものがあります。

- -Xmap

-Xmap

リンク・マップ・ファイルを出力します。

[指定形式]

```
-Xmap[=file]
```

- 省略時解釈

  リンク・マップ・ファイルを出力しません。

[詳細説明]

- リンク・マップ・ファイル *file* を出力します。
- リンク・マップ・ファイルの内容を以下に示します。
  - 指定したオブジェクト・モジュール・ファイルに含まれる入力セクションのメモリ空間への割り付け
  - 入力セクションを結合して生成するロード・モジュール・ファイルを構成する出力セクションのメモリ空間への割り付け
  - シンボルのアドレス情報
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* にファイル名のみを指定した場合は，ロード・モジュール・ファイルと同じフォルダに，指定したファイル名で出力します。
- *file* を省略した場合は，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .map に置き換えた名前で出力します。
- -Xno_romize オプションの有無により，出力するリンク・マップ・ファイルは以下のようになります。
  - -Xno_romize オプションの指定あり

    リンク処理後までのリンク・マップ・ファイルを出力します。
  - -Xno_romize オプションの指定なし

    ROM 化処理後までのリンク・マップ・ファイルを出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［リンク・マップ］カテゴリの［リンク・マップ・ファイルを出力する］，［リンク・マップ・ファイル出力フォルダ］，［リンク・マップ・ファイル名］

［使用例］

- リンク処理後のリンク・マップ・ファイル smp1.map を出力します。

```
>cx -CF3746 -Xmap=smp.map main.obj -Xno_romize
```

- ROM 化処理後のリンク・マップ・ファイル smp2.map を出力します。

```
>cx -CF3746 -Xmap=smp.map main.obj
```

シンボル情報出力指定

シンボル情報出力指定オプションには，次のものがあります。

- -Xsymbol_dump

-Xsymbol_dump

リンク・マップ・ファイルにシンボル情報を出力します。

[指定形式]

```
-Xsymbol_dump
```

- 省略時解釈

  リンク・マップ・ファイルにシンボル情報を出力しません。

[詳細説明]

- -Xmap オプションで指定したリンク・マップ・ファイルにシンボル情報を出力します。
- 本オプションは，-Xmap オプション指定時のみ有効です。

  -Xmap オプションを指定しない場合は，警告を出力して，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［リンク・マップ］カテゴリの［リンク・マップ・ファイルへシンボル情報を出力する］

[使用例]

- リンク・マップ・ファイル smp.map にシンボル情報を出力します。

```
>cx -CF3746 -Xmap=smp.map -Xsymbol_dump main.obj
```

生成オブジェクト・モジュール・ファイル制御

生成オブジェクト・モジュール・ファイル制御オプションには，次のものがあります。

- -Xrelinkable_object

-Xrelinkable_object

再配置可能なオブジェクト・モジュール・ファイルを生成します。

[指定形式]

```
-Xrelinkable_object
```

- 省略時解釈

  実行可能なオブジェクト・モジュール・ファイルを生成しようとします。

  リンク終了後に未解決な外部参照が残されていた場合，エラーを出力して，リンクを中止します。

  その際，ロード・モジュール・ファイルは生成しません。

[詳細説明]

- 再配置可能なオブジェクト・モジュール・ファイルを生成します。
- 本オプションを指定すると，リンク終了後に未解決な外部参照が残されていた場合，メッセージを出力せず，リンクを正常終了します。
- cx によって生成したオブジェクト・モジュール・ファイルを再配置の対象として指定する場合，本オプションを使用して，再配置の対象となるオブジェクト・モジュール・ファイルを生成します。
- 本オプションを指定した場合，リンク・ディレクティブはセグメント・ディレクティブ部分のタイプ／属性のみが有効になり，その他は無視します。
- 本オプションを指定した場合，予約シンボルの作成は行いません。
- 本オプションを指定した場合，-Xno_startup，および -Xno_romize オプションが暗黙に指定されたものとみなします。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［出力ファイル］カテゴリの［再配置可能なオブジェクト・モジュール・ファイルを生成する］

[使用例]

- 再配置可能なオブジェクト・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xrelinkable_object main.obj -Xno_startup -Xno_romize
```

レジスタ・モード混在チェック制御

レジスタ・モード混在チェック制御オプションには，次のものがあります。

- -Xreg_mode
- -Xregmode_info

-Xreg_mode

指定したレジスタ・モードに対して，混在チェックを行います。

[指定形式]

```
-Xreg_mode=mode
```

- 省略時解釈

  32 レジスタ・モードに対して，混在チェックを行います。

[詳細説明]

- コンパイル・オプション，またはアセンブル・オプションとして指定したレジスタ・モード *mode* に対して，混在チェックを行います。
- *mode* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

レジスタ・モード（*mode*）	作業用レジスタ	レジスタ変数用レジスタ
common	r10 ～ r14	r25 ～ r29
22	r10 ～ r14	r25 ～ r29
26	r10 ～ r16	r23 ～ r29
32	r10 ～ r19	r20 ～ r29

- *mode* に common を指定した場合は，混在チェックの対象外とします。
- *mode* を省略した場合は，エラーとなります。
- レジスタ・モードが異なるオブジェクト・モジュール・ファイルが混在している場合は，警告を出力して，リンク処理を続行します。

[使用例]

- 22 レジスタ・モードの混在チェックを行います。

```
>cx -CF3746 -Xreg_mode=22 main.obj
```

-Xregmode_info

異なるレジスタ・モードが混在している場合，詳細情報を出力します。

[指定形式]

```
-Xregmode_info
```

- 省略時解釈

  異なるレジスタ・モードが混在している場合，詳細情報を出力しません。

[詳細説明]

- すべての入力オブジェクト・モジュール・ファイルに対して，異なるレジスタ・モードが混在している場合，詳細情報を出力して，警告の原因となっている入力オブジェクト・モジュール・ファイルを特定します。
- レジスタ・モードが統一されている場合は，情報は出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［レジスタ・モードをチェックする］

[使用例]

- すべての入力オブジェクト・モジュール・ファイルに対して，異なるレジスタ・モードが混在している場合，詳細情報を出力します。

```
>cx -CF3746 -Xregmode_info file1.obj file2.obj
```

デバイス共通オブジェクト混在チェック指定

デバイス共通オブジェクト混在チェック指定オプションには，次のものがあります。

- -Xcommon

-Xcommon

生成するデバイス共通オブジェクト・モジュール・ファイルと -C オプションで指定したデバイスの混在チェックを行います。

[指定形式]

```
-Xcommon=series
```

- 省略時解釈

  -C オプションの指定内容に従って，リンク処理を行います。

[詳細説明]

- コンパイル・オプション，またはアセンブル・オプションとして指定した共通マジック・ナンバ *series* と -C オプションで指定したデバイスのシリーズ番号に対して，混在チェックを行います。
- *series* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

v850e	命令セット・アーキテクチャが V850E より上位（V850E，V850E2，V850E2V3）の品種をターゲット・デバイスとして指定したものとのリンクが可能です。
v850e2	命令セット・アーキテクチャが V850E2 より上位（V850E2，および V850E2V3）の品種をターゲット・デバイスとして指定したものとのリンクが可能です。
v850e2v3	V850E2V3 共通 命令セット・アーキテクチャが V850E2V3 である品種をターゲット・デバイスとして指定したものとのリンクが可能です。 ターゲット・デバイスの命令セット・アーキテクチャが V850E2V3 である場合は，その性能を引き出すために，本項目を指定することを推奨します。

- *series* を省略した場合は，エラーとなります。

[使用例]

- 共通マジック・ナンバと v850e と -C オプションで指定したデバイス μPD70F3746 に対して，混在チェックを行います。

```
>cx -CF3746 -Xcommon=v850e main.obj
```

sdata/sbss 情報出力指定

sdata/sbss 情報出力指定オプションには，次のものがあります。

- -Xsdata_info

-Xsdata_info

-Xsdata オプションのパラメータに対して，目安として用いることのできる情報を標準出力に出力します。

[指定形式]

```
-Xsdata_info
```

- 省略時解釈

  -Xsdata オプションのパラメータに対して，目安として用いることのできる情報を出力しません。

[詳細説明]

- -Xsdata オプション（sdata，sbss セクションに配置するデータの最大サイズの指定）のパラメータに対して，目安として用いることのできる情報を標準出力に出力します。
- *OK* と表示された数値を用いると，sdata，sbss の領域へは，その数値以下のサイズを持つデータが割り当てられます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［GP 情報を表示する］

[使用例]

- -Xsdata オプションのパラメータに対して，目安として用いることのできる情報を標準出力に出力します。

```
>cx -CF3746 -Xsdata_info main.obj
```

2 パス・モード・リンク指定

2 パス・モード・リンク・オプションには，次のものがあります。

- -Xtwo_pass_link

-Xtwo_pass_link

2 パス・モードでリンクを行います。

[指定形式]

```
-Xtwo_path_link
```

- 省略時解釈

  1 パス・モードでリンクを行います。

[詳細説明]

- 2 パス・モードでリンクを行います。
- 2 パス・モードは，1 パス・モードよりも処理が遅いですが，より大きなサイズのファイルを処理することができます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［2 パス・モードでリンクする］

[使用例]

- 2 パス・モードでリンクを行います。

```
>cx -CF3746 -Xtwo_path_link main.obj
```

リロケーション解決エラー処理制御

リロケーション解決エラー処理制御オプションには，次のものがあります。

- -Xignore_address_error

-Xignore_address_error

リンク時のリロケーション処理において，不正箇所があった場合，リンク処理を続行します。

[指定形式]

```
-Xignore_address_error
```

- 省略時解釈

  リンク時のリロケーション処理において，不正箇所があった場合，エラーとします。

[詳細説明]

- リンク時のリロケーション処理において，以下の不正箇所があった場合，エラーとはせず，警告を出力して，リンク処理を続行します。
  - 未解決な外部参照のアドレス計算結果が不正な場合
  - 配置されるセクションとの関係が不正な場合

  具体的には，以下のような場合を指します。
  - GP 相対リロケーションのための GP シンボルが存在しない場合（LOCAL ／ GLOBAL ／ EXTERN）
  - PC22/26 のリロケーション結果が奇数アドレスへの分岐である場合
  - PC 相対リロケーション結果が許容範囲を越えた場合（シンボルあり／シンボルなし）
  - PC 相対以外のリロケーション結果が許容範囲を越えた場合（シンボルあり／シンボルなし）
- 誤りとされた未解決な外部参照に対しては，不正と判断されたアドレスの計算結果の値は入れず，元の値を残します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［リロケーション不正を無視する］

[使用例]

- リンク時のリロケーション処理において，未解決な外部参照のアドレス計算結果が不正な場合，警告を出力して，リンク処理を続行します。

```
>cx -CF3746 -Xignore_address_error main.obj
```

シンボル多重定義エラー出力指定

シンボル多重定義エラー出力指定オプションには，次のものがあります。

- -Xmultiple_symbol

-Xmultiple_symbol

多重定義されたすべての外部シンボルに対してエラー・メッセージを出力します。

[指定形式]

```
-Xmultiple_symbol
```

- 省略時解釈

  多重定義された最初の外部シンボルに対してエラー・メッセージを出力して，リンク処理を中止します。

[詳細説明]

- 多重定義されたすべての外部シンボルとファイル名に対してエラー・メッセージを出力して，リンク処理を中止します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［すべての多重定義シンボルを検出する］

[使用例]

- 多重定義されたすべての外部シンボルに対してエラー・メッセージを出力して，リンク処理を中止します。

```
>cx -CF3746 -Xmultiple_symbol main.obj
```

リンク時チェック抑止指定

リンク時チェック抑止指定オプションには，次のものがあります。

- -Xlink_check_off

-Xlink_check_off

リンク時のチェックを抑止します。

[指定形式]

```
-Xlink_check_off=string[,string]
```

- 省略時解釈

外部シンボルをリンクする際，シンボルのサイズのチェックを行い，サイズの違いを検出した場合は，警告を出力して，リンク処理を続行します。
その際，実際にシンボルが定義されているファイルのシンボル・サイズが有効となります。
未定義シンボルをリンクする際，シンボルのサイズ，および整列条件のチェックを行い，違いを検出した場合は，警告を出力して，リンク処理を続行します。
内蔵 ROM 領域となっているアドレスに，アプリケーションの配置がオーバラップしている場合は，警告を出力します。

[詳細説明]

- リンク時のチェックを抑止します。
- *string* に指定可能なものを以下に示します。
  これ以外のものを指定した場合は，エラーとなります。

symbol	外部シンボルをリンクする際，シンボルのサイズ，および整列条件のチェックを行いません。
undefined	未定義外部シンボルをリンクする際，シンボルのサイズ，および整列条件のチェックを行いません。
irom	内蔵 ROM 領域への配置に対して，チェックを行いません。 つまり，内蔵 ROM 領域となっているアドレスに，アプリケーションの配置がオーバラップしていても，警告を出力しません。

- *string* を省略した場合は，エラーとなります。
- アプリケーションを ROM レス・モードで作成する場合は，irom を指定したものとみなします。

注意　シングルチップ・モード選択時の内蔵 ROM オーバのチェックには対応していません。
-Xlink_check_off=irom を指定して内蔵 ROM オーバフローのチェックを無効とし，リンク・マップで確認してください。

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［外部シンボル不正をチェックする］，［未定義外部シンボル不正をチェックする］，［内蔵 ROM 領域への配置をチェックする］

［使用例］

- 外部シンボルをリンクする際，サイズ，および整列条件のチェックを行いません。

```
>cx -CF3746 -Xlink_check_off=symbol main.obj
```

充てん値指定

充てん値指定オプションには，次のものがあります。

- -Xalign_fill

-Xalign_fill

アライン・ホールの充てん値を指定します。

[指定形式]

```
-Xalign_fill=value
```

- 省略時解釈

  生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値を 0x0000 とします。

[詳細説明]

- 生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値 *value* を指定します。
- *value* に指定可能な値の範囲は，0x0000 ～ 0xFFFF です。

  この範囲外の値を指定した場合は，エラーとなります。
- 指定した値が 4 桁に満たない場合は，上位の桁を 0 で補てんします。
- *value* を省略した場合は，エラーとなります。
- 本オプションを指定する場合は，-Xtwo_path_link オプションを指定して 2 パス・モードでリンクを行う必要があります。

  -Xtwo_path_link オプションを指定しない場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［ホールの充てん値］

[使用例]

- 生成するロード・モジュール・ファイル内のセクション間のアライン・ホールの充てん値を 0xFFFF とします。

```
>cx -CF3746 -Xalign_fill=0xFFFF -Xtwo_path_link main.obj
```

ライブラリ・ファイル再スキャン指定

ライブラリ・ファイル再スキャン指定オプションには，次のものがあります。

- -Xrescan

-Xrescan

-l オプションで指定したライブラリ・ファイルの再スキャンを行います。

[指定形式]

```
-Xrescan
```

- 省略時解釈

  -l オプションで指定したライブラリ・ファイルの再スキャンを行いません。

[詳細説明]

- -l オプションで指定したライブラリ・ファイルの再スキャンを行います。
- 本オプションを指定すると，ライブラリのリンク順によるシンボル未解決を防ぐことができます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［その他］カテゴリの［ライブラリ・ファイルを再スキャンする］

[使用例]

- ライブラリ・ファイル libtest1.lib，libtest2.lib の再スキャンを行います。

```
>cx -CF3746 -Xrescan main.obj -ltest1 -ltest2
```

デバッグ情報セクション出力抑止指定

デバッグ情報セクション出力抑止指定オプションには，次のものがあります。

- -Xstrip

-Xstrip

デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xstrip
```

- 省略時解釈
- ロード・モジュール・ファイルの生成において，デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルが存在する場合，これらの削除を行いません。

[詳細説明]

- ロード・モジュール・ファイルの生成において，デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［リンク・オプション］タブの［デバッグ情報］カテゴリの［デバッグ情報を削除する］

[使用例]

- ロード・モジュール・ファイルの生成において，デバッグ情報，ライン・ナンバ情報，およびグローバル・ポインタ・テーブルを削除したロード・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xstrip main.obj
```

ブートーフラッシュ再リンク機能制御

ブートーフラッシュ再リンク機能制御オプションには，次のものがあります。

- -Xflash_ext_table
- -Xflash

-Xflash_ext_table

ブートーフラッシュ再リンク機能用の分岐テーブル先頭アドレス値を指定します。

[指定形式]

```
-Xflash_ext_table=address
```

- 省略時解釈

  ブートーフラッシュ再リンク機能用のロード・モジュール・ファイルを生成せず，通常のリンク処理を行います。

[詳細説明]

- アドレス *address* を分岐テーブル先頭アドレス値として，ブートーフラッシュ再リンク機能用のロード・モジュール・ファイルを生成します。

  ブートーフラッシュ再リンク機能についての詳細は，「B. 1. 6　ブートーフラッシュ再リンク機能」を参照してください。
- -Xflash オプションの指定の有無により，フラッシュ領域側／ブート領域側のロード・モジュール・ファイルの生成を指定します。
  - ブート領域側のロード・モジュール・ファイルの生成を指定した場合（-Xflash オプションの指定なし）

    フラッシュ領域側への分岐処理となります。

    その際，本オプションで指定したアドレスに生成される分岐テーブルへの分岐として処理します。
  - フラッシュ領域側のロード・モジュール・ファイルの生成を指定した場合（-Xflash オプションの指定あり）

    本オプションで指定したアドレスに，本来の分岐先への分岐命令を持つ分岐テーブルを生成します。
- *address* に指定可能な値の範囲は，0x00000000 ～ 0xFFFFFFFF です。

  この範囲外の値を指定した場合は，エラーとなります。
- 指定した値が 8 桁に満たない場合は，上位の桁を 0 で補てんします。
- *address* に奇数を指定した場合は，偶数に補正したのち，警告を出力して，処理を続行します。
- *address* を省略した場合は，エラーとなります。
- *address* は，ブート領域側／フラッシュ領域側のロード・モジュール・ファイルの生成の際に，同じ値とする必要があります。

  異なる値を指定した場合は，正しく動作しません。

  また，エラー・チェックも行っていません。
- *address* は，フラッシュ領域側 ROM 内である必要があります。

  指定したアドレスがどちらの領域であるかの判断ができないため，エラー・チェックは行っていません。

- 本オプションを指定することにより，フラッシュ領域側のロード・モジュール・ファイルの生成時には，*address* を先頭とする，サイズ（(ID 値[注]の最大＋1）×分岐テーブルのエントリ・サイズ）バイトのセクション .ext_table を自動生成します。
  このセクションは，リンク・ディレクティブ・ファイルで配置指定を行う必要はありませんが，配置するため領域を空けておく必要があります。

  注　アセンブラ・ソース・ファイルに .ext_func 疑似命令で指定された値

- ブート領域側のロード・モジュール・ファイルを生成する場合は，ROM 化処理前のロード・モジュール・ファイルを保存する必要があります。
  これは，フラッシュ領域側のロード・モジュール・ファイルを生成する際に，-Xflash オプションでそのファイルを指定するためです。
  したがって，本オプション指定時に -Xflash オプションを同時に指定しない場合は，-Xlink_output オプション（ROM 化を行う場合），または -Xno_romize（ROM 化を行わない場合）と同時に指定する必要があります。
- 本オプションは，-Xrelinkable_object オプションと同時に指定することはできません。
  また，-Xrelinkable_object オプションにより生成したロード・モジュール・ファイルを入力した場合は，正しく動作しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［フラッシュ対応］カテゴリの［分岐テーブルのアドレス］

［使用例］

- 分岐テーブル先頭アドレス値を 0x200 として，ブート領域側のロード・モジュール・ファイル boot.lmf を生成します。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xlink_output=boot.lmf boot.obj
```

-Xflash

フラッシュ領域側のロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xflash=file
```

- 省略時解釈

  ブートーフラッシュ再リンク機能を使用する場合は，ブート領域側のロード・モジュール・ファイルを生成します。

  ブートーフラッシュ再リンク機能を使用しない場合は，通常のリンク処理を行います。

[詳細説明]

- ブートーフラッシュ再リンク機能を使用する場合に，フラッシュ領域側のロード・モジュール・ファイルを生成します。

  その際，ブート領域側のロード・モジュール・ファイル *file* のシンボル情報を参照して，リンク処理を行います。
- *file* には，ブートーフラッシュ再リンク機能を使用して生成したブート領域側のロード・モジュール・ファイルを指定します。

  ここで指定するロード・モジュール・ファイルは，ROM 化処理前のもの（-Xno_romize，または -Xlink_output オプションを指定して生成したもの）である必要があります。
- *file* に存在しないファイルを指定した場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，-Xflash_ext_table オプションと同時に指定する必要があります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［共通オプション］タブの［フラッシュ対応］カテゴリの［生成するロード・モジュール・ファイルの種類］，［ブート領域用ロード・モジュール・ファイル名］

[使用例]

- フラッシュ領域側の分岐テーブル先頭アドレスが 0x200 番地にあるものとして，ブート領域側のロード・モジュール・ファイル boot.lmf を生成します。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xlink_output=boot.lmf boot1.obj boot2.obj -
Xlink_directive=boot.dir
```

0x200 番地に分岐テーブルを生成して，フラッシュ領域側のロード・モジュール・ファイル flash.lmf を生成します。

その際，ブート領域側のロード・モジュール・ファイル boot.lmf のシンボル情報を参照して，リンク処理を行います。

```
>cx -CF3746 -Xflash_ext_table=0x200 -Xflash=boot.lmf -oflash.lmf flash1.obj flash2.obj  -
Xlink_directive=flash.dir
```

ROM 化処理前ロード・モジュール・ファイル保存指定

ROM 化処理前ロード・モジュール・ファイル保存指定オプションには，次のものがあります。

- -Xlink_output

-Xlink_output

ROM 化処理前のロード・モジュール・ファイルを保存します。

[指定形式]

```
-Xlink_output=file
```

- 省略時解釈

  ROM 化処理前のロード・モジュール・ファイルを保存しません。

[詳細説明]

- ROM 化処理前のロード・モジュール・ファイルをファイル名 *file* で保存します。
- ROM 化処理前のロード・モジュール・ファイルは，通常，削除しますが，これを保存したい場合に，本オプションを指定します。
- *file* の推奨拡張子は，.lmf です。
- *file* を省略した場合は，エラーとなります。
- -Xno_romize オプションと同時に指定した場合は，警告を出力して，本オプションを無視します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［出力ファイル］カテゴリの［ROM 化処理前のロード・モジュール・ファイルを出力する］

[使用例]

- ROM 化処理前のロード・モジュール・ファイルをファイル名 sample.lmf で保存します。

```
>cx -CF3746 -Xlink_output=sample.lmf main.c
```

マルチコア対応指定

マルチコア対応指定オプションには，次のものがあります。

- -Xmulti
- -Xmulti_link

-Xmulti

マルチコア用プログラムのサブプログラムの生成を指定します。

[指定形式]

```
-Xmulti=type
```

- 省略時解釈

  シングルコア用プログラムを生成します。

  なお，-C オプションでマルチコア CPU のデバイスを指定し，かつ，-Xmulti_link オプションを指定していない場合は，本オプションを省略することはできません。省略した場合は，エラーとなります。

[詳細説明]

- マルチコア用プログラムのサブプログラムを生成する際に，サブプログラムの種別を指定します。
- マルチコア CPU でないターゲットに対して本オプションを指定した場合，警告を出力して無視します。
- *type* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

pe*n*	コア *n* 用プログラムを生成します。 *n* に指定可能な値は，1 ～ターゲット CPU が持つコアの数までの整数です。 これ以外の値を指定した場合は，エラーとなります。
cmn	共通部を生成します。 入力ファイルが C ソース・ファイルである場合，変数アクセス，および関数アドレスの取得に対して r0 相対命令を生成します（ファイルの先頭に #pragma nopic 指令が記述しているのと同様に扱います）。 #pragma pic 指令を記述している場合は，エラーとなります。 C ソース・ファイル，アセンブラ・ソース・ファイルともに，データに対して data，const，sconst 以外の再配置属性（sdata など）を持つセクションへの配置が指定されている場合は，エラーとなります。 また，-Xsdata=0 オプションを指定したものとみなし，-Xsdata オプションを指定している場合は，警告を出力して無視します。

- 本オプションを指定した場合，コンパイル時，およびアセンブル時に *type* に応じたコードを生成し，サブプログラム内で解決可能な範囲でシンボル参照を解決し，リンク処理を行って再リンク可能なロード・モジュール・ファイルを生成します。

このとき，デフォルトのスタートアップ・ルーチン，および標準ライブラリはリンクしません。
また，ROM 化処理，およびヘキサ・ファイルの生成も行いません。

- 本オプションを指定した場合，以下のプリプロセッサ・マクロを自動的に設定します。

type	定義するプリプロセッサ・マクロ（値は 1）
pe*n*	__MULTI_CORE__，MULTI_PE*n*__
cmn	__MULTI_CORE__，MULTI_CMN__

- 本オプションを指定した場合，アセンブル時にセクション名を自動的に変換します（セクション名がソース中で明示的に指定されている場合を除く）。
すなわち，デフォルトのセクション名の後に type に応じた接尾子（.pen，または .cmn）を追加します。
また，各オブジェクト・モジュール・ファイル中に，そのファイルで定義されているセクションと *type* の対応を示す情報のセクションを出力します。
- 本オプションを指定した場合に影響するオプションを以下に示します。
  - 暗黙に有効になるオプション

```
-Xno_startup，-Xno_stdlib，-Xno_romize，-Xrelinkable_object
```

  - 無効にするオプション

```
-l，-L，-Xstartup，-Xno_romize，-Xrompcrt，-Xrompsec_start，-Xrompsec_data，
-Xrompsec_text，-Xrompsec_only，-Xromize_check_off，-Xhex，-Xhex_only，-Xhex_format，
-Xhex_fill，-Xhex_section，-Xhex_block_size，-Xhex_offset，-Xhex_null，-Xhex_symtab，
-Xhex_rom_less，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，
-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata
```

- 各サブプログラム用のライブラリ・ファイルを作成する場合は，本オプションを -c オプションと同時に指定して，オブジェクト・モジュール・ファイルを生成します。

注意　異なる -Xmulti オプションを指定して生成したオブジェクト・モジュール・ファイルをまとめてライブラリ・ファイルを作成しないでください。

- 本オプションと -Xmuitl_link オプションを同時に指定した場合はエラーになります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ビルド設定］タブの［マルチコア］カテゴリの［対象コア番号］

［使用例］

- コア 1 用プログラムを生成します。

```
>cx -CF3515 -Xmulti=pe1 file_pe1_1.c file_pe1_2.c -ope1.lmf
```

-Xmulti_link

マルチコア用サブプログラムのリンクを指定します。

[指定形式]

```
-Xmulti_link
```

- 省略時解釈

  マルチコア用サブプログラムのリンクとみなしません。

  なお，-C オプションでマルチコア CPU のデバイスを指定し，かつ，-Xmulti オプションを指定していない場合は，本オプションを省略することはできません。省略した場合は，エラーとなります。

[詳細説明]

- マルチコア用プログラムを生成する際に，生成済みの各サブプログラムをリンクします。
- 本オプションを指定した場合，各サブプログラム，およびライブラリ・ファイル，オブジェクト・モジュール・ファイルを入力して，サブプログラム間のアドレス解決を行い，マルチコア用プログラムを生成します。

  このとき，マルチコア対応スタートアップ・ルーチン，およびライブラリをリンクします。

  また，ROM 化処理，およびヘキサ・ファイルの生成も同時に行います。
- 入力として，C ソース・ファイル，またはアセンブラ・ソース・ファイルを指定した場合は，エラーとなります。

  それ以外の種別のファイルを指定した場合，必要なファイルを指定していない場合，未解決の gp，ep，tp 相対シンボル参照が残っている場合は，リンク時にエラーとなります。
- 本オプションと -Xmuitl オプションを同時に指定した場合はエラーになります。
- 本オプションを指定した場合は，警告を出力して，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata オプションを無視します。

[使用例]

- マルチコア用サブプログラム pe1.lmf，pe2.lmf，cmn.lmf をリンクします。

```
>cx -CF3515 -Xmulti_link pe1.lmf pe2.lmf cmn.lmf -otarget.lmf -lmulti_lib
```

エラー出力制御

エラー出力制御オプションには，次のものがあります。

- -Xerror_file

-Xerror_file

エラー・メッセージをファイルに出力します。

[指定形式]

```
-Xerror_file=file
```

- 省略時解釈

  エラー・メッセージを標準エラー出力のみに出力します。

[詳細説明]

- エラー・メッセージを標準エラー出力，およびファイル *file* に出力します。
- *file* がすでに存在する場合は，そのファイルを上書きします。
- *file* を省略した場合は，エラーとなります。

[使用例]

- エラー・メッセージを標準エラー出力，およびファイル err に出力します。

```
>cx -CF3746 -Xerror_file=err main.obj
```

警告メッセージ出力制御

警告メッセージ出力制御オプションには，次のものがあります。

- -Xwarning
- -Xno_warning

-Xwarning

指定した警告メッセージを出力します。

[指定形式]

```
-Xwarning=num[,num]...
-Xwarning=num1-num2
```

- 省略時解釈

  重大な警告メッセージを出力します。

[詳細説明]

- 指定した警告メッセージを出力します。
- *num*，*num1*，*num2* には，エラー番号を指定します。

  存在しないエラー番号を指定した場合は，無視します。
- *num*，または *num1*，および *num2* を省略した場合は，エラーとなります。
- *num1-num2* の形式で指定すると，その範囲に含まれるエラー番号を指定したものとみなします。
- 本オプションで指定するエラー番号は，W に続く 7 桁の数字のうち，下位 5 桁です。

  エラー番号については，「CubeSuite+ メッセージ編」を参照してください。

[使用例]

- 警告メッセージ W0566002 を出力します。

```
>cx -CF3746 -Xwarning=66002 main.obj
```

-Xno_warning

指定した警告メッセージの出力を抑止します。

[指定形式]

```
-Xno_warning=num[,num]...
-Xno_warning=num1-num2
```

- 省略時解釈

  重大な警告メッセージを出力します。

[詳細説明]

- 指定した警告メッセージの出力を抑止します。
- *num*，*num1*，*num2* には，エラー番号を指定します。

  存在しないエラー番号を指定した場合は，無視します。
- *num*，または *num1*，および *num2* を省略した場合は，エラーとなります。
- *num1-num2* の形式で指定すると，その範囲に含まれるエラー番号を指定したものとみなします。
- 本オプションで指定するエラー番号は，W に続く 7 桁の数字のうち，下位 5 桁です。

  エラー番号については，「CubeSuite+ メッセージ編」を参照してください。

[使用例]

- 警告メッセージ W0566002 の出力を抑止します。

```
>cx -CF3746 -Xno_warning=66002 main.obj
```

フェーズ個別オプション指定

フェーズ個別オプション指定オプションには，次のものがあります。

- -Xlk_option

-Xlk_option

リンク対象ファイルを指定します。

[指定形式]

```
-Xlk_option=file1[,file2]...
```

- 省略時解釈

  .obj ／ .lib ／ .lmf のみをリンク対象として認識します。

[詳細説明]

- リンク対象として認識しないファイル（.obj ／ .lib ／ .lmf 以外）がリンク対象となるよう，cx に指示します。
- *file* には，リンクを行いたいファイルを指定します。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。

[使用例]

- ファイル link.test のリンクを行います。

```
>cx -CF3746 -Xlk_option=link.test
```

コマンド・ファイル指定

コマンド・ファイル指定オプションには，次のものがあります。

- @

@

コマンド・ファイルを指定します。

[指定形式]

```
@file
```

- 省略時解釈

  コマンド・ラインで指定したオプション，およびファイル名のみを認識します。

[詳細説明]

- *file* をコマンド・ファイルとして扱います。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- コマンド・ファイルについての詳細は，「(2) コマンド・ファイルによる操作方法」を参照してください。

[使用例]

- command をコマンド・ファイルとして扱います。

```
>cx @command
```

（4）ROM 化オプション

ROM 化フェーズのオプションの分類と説明を以下に示します。

表 B―7　ROM 化オプション

分類	オプション	説明
ROM 化処理抑止指定	-Xno_romize	ROM 化処理を抑止します。
ROM 化用領域確保コード・ファイル指定	-Xrompcrt	ROM 化用領域確保コード・ファイルを指定します。
rompsec セクション制御	-Xrompsec_start	rompsec セクションの先頭アドレスを指定します。
	-Xrompsec_data	rompsec セクションに含めるデータ・セクションを指定します。
	-Xrompsec_text	rompsec セクションに含めるテキスト・セクションを指定します。
	-Xrompsec_only	rompsec セクションのみを持つロード・モジュール・ファイルを生成します。
ROM 化時エラー・チェック制御	-Xromize_check_off	ROM 化時のエラー・チェックを省略します。

ROM 化処理抑止指定

ROM 化処理抑止指定オプションには，次のものがあります。

- -Xno_romize

-Xno_romize

ROM 化処理を抑止します。

[指定形式]

```
-Xno_romize
```

- 省略時解釈

  ROM 化処理を行います。

  また，以下をリンクします。

  - ROM 化用領域確保コード・ファイル（-Xno_stdlib オプション指定時を除く）
  - コピー関数 _rcopy の呼び出しを含むスタートアップ・ルーチン（cstart.obj）（-Xstartup，または -Xno_startup オプション指定時を除く）

[詳細説明]

- ROM 化処理を抑止します。

  また，ROM 化用領域確保コード・ファイルのリンクを行いません。

  コピー関数 _rcopy の呼び出しを含まないスタートアップ・ルーチン（cstartN.obj）をリンクします（-Xstartup，または -Xno_startup オプション指定時を除く）。
- ROM 化処理とは，RAM に展開するデータを ROM 上に持たせておき，アプリケーションの最初に，そのデータを ROM から RAM へコピーするルーチンを追加する処理のことです。
- 初期値ありデータなどがないアプリケーションにおいて，本オプションを指定すると，コードを削減することができます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［出力ファイル］カテゴリの［ROM 化用ロード・モジュール・ファイルを出力する］

[使用例]

- ROM 化処理を抑止します。

```
>cx -CF3746 -Xno_romize main.c
```

ROM 化用領域確保コード・ファイル指定

ROM 化用領域確保コード・ファイル指定オプションには，次のものがあります。

- -Xrompcrt

-Xrompcrt

ROM 化用領域確保コード・ファイルを指定します。

[指定形式]

```
-Xrompcrt=file
```

- 省略時解釈

  -Xno_romize オプションを指定しない場合に，標準の ROM 化用領域確保コード・ファイル（rompcrt.obj）を入力ファイルの最後にリンクします。

[詳細説明]

- 標準の ROM 化用領域確保コード・ファイルの代わりに，*file* を入力ファイルの最後にリンクします。
- *file* には，オブジェクト・モジュール・ファイルを指定します。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，-Xno_romize オプションと同時に指定した場合は無効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［入力ファイル］カテゴリの［ROM 化用領域確保コード・ファイル名］

[使用例]

- 標準の ROM 化用領域確保コード・ファイルの代わりに，rompack.obj を入力ファイルの最後にリンクします。

```
>cx -CF3746 -Xrompcrt=rompack.obj main.c
```

rompsec セクション制御

rompsec セクション制御オプションには，次のものがあります。

- -Xrompsec_start
- -Xrompsec_data
- -Xrompsec_text
- -Xrompsec_only

-Xrompsec_start

rompsec セクションの先頭アドレスを指定します。

[指定形式]

```
-Xrompsec_start=label
```

- 省略時解釈

  ラベル __S_romp の値を，生成する rompsec セクションの先頭アドレスとします。

[詳細説明]

- 指定したセクションのみ，ヘキサ出力を行います。
- ラベル *label* の値を，生成する rompsec セクションの先頭アドレスとします。
- *label* がロード・モジュール・ファイル中に存在しない場合，または本オプションを複数指定した場合は，あとから指定したオプションが有効となり，先に指定したオプションは無視します。
- *label* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［セクション］カテゴリの［rompsec セクションの開始シンボル］

[使用例]

- ラベル romp_start の値を，生成する rompsec セクションの先頭アドレスとします。

```
>cx -CF3746 -Xrompsec_start=romp_start main.c
```

-Xrompsec_data

rompsec セクションに含めるデータ・セクションを指定します。

[指定形式]

```
-Xrompsec_data=section[,section]...
```

- 省略時解釈

  data 属性，または sdata 属性を持つすべてのセクション，および内蔵命令 RAM に配置されるセクションを rompsec セクションに含めます。

[詳細説明]

- rompsec セクションに含めるデータ・セクションを指定します。
- セクション *section* の内容とそのアドレス，およびサイズの情報を rompsec セクションに含めます。
- 本オプションは，data 属性，または sdata 属性を持つセクションに関するオプションです。
- *section* がロード・モジュール・ファイル中に存在しない場合は，エラーを出力して，処理を中止します。
- *section* には，空白を使用することはできません。
- *section* を省略した場合は，エラーとなります。
- 本オプションを複数指定した場合は，指定した順に各データ・セクションを rompsec セクションに含めます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［セクション］カテゴリの［rompsec セクションに含めるデータ・セクション］

[使用例]

- セクション data1，および data2 を rompsec セクションに含めます。

```
>cx -CF3746 -Xrompsec_data=data1,data2 main.c
```

-Xrompsec_text

rompsec セクションに含めるテキスト・セクションを指定します。

[指定形式]

```
-Xrompsec_text=section[,section]...
```

- 省略時解釈

  内蔵命令 RAM に配置される各セクションを rompsec セクションに含めます。

[詳細説明]

- rompsec セクションに含めるテキスト・セクションを指定します。
- セクション *section* の内容とそのアドレス，およびサイズの情報を rompsec セクションに含めます。
- 本オプションは，text 属性，または const 属性を持つセクションに関するオプションです。
- *section* に指定可能なセクションは，text 属性，または const 属性を持つセクションです。
  これ以外の属性のセクションを指定した場合は，警告を出力して，処理を中止します。
- *section* がロード・モジュール・ファイル中に存在しない場合は，エラーを出力して，処理を中止します。
- *section* には，空白を使用することはできません。
- *section* を省略した場合は，エラーとなります。
- 本オプションを複数指定した場合は，指定した順に rompsec セクションに含めます。
- 内蔵命令 RAM 搭載のデバイス・ファイルを指定してリンクされた入力ファイルに対して，本オプションで特定のセクションを指定した場合，指定しなかった内蔵命令 RAM に配置されたセクションは，rompsec セクションに入らないだけでなく，出力ファイル中からも削除します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［セクション］カテゴリの［rompsec セクションに含めるテキスト・セクション］

[使用例]

- セクション text1，および text2 を rompsec セクションに含めます。

```
>cx -CF3746 -Xrompsec_text=text1,text2 main.c
```

-Xrompsec_only

rompsec セクションのみを持つロード・モジュール・ファイルを生成します。

[指定形式]

```
-Xrompsec_only
```

- 省略時解釈

  生成するロード・モジュール・ファイル中に text 属性を持つセクションも含めます。

[詳細説明]

- 生成するロード・モジュール・ファイル中に text 属性を持つセクションを含めずに，rompsec セクションのみを持つロード・モジュール・ファイルを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［セクション］カテゴリの［rompsec セクションのみのロード・モジュール・ファイルを生成する］

[使用例]

- rompsec セクションのみを持つロード・モジュール・ファイルを生成します。

```
>cx -CF3746 -Xrompsec_only main.c
```

ROM 化時エラー・チェック制御

ROM 化時エラー・チェック制御オプションには，次のものがあります。

- -Xromize_check_off

-Xromize_check_off

ROM 化時のエラー・チェックを省略します。

[指定形式]

```
-Xromize_check_off=string[,string]
```

- 省略時解釈

  rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行います。

  入力ファイル，および出力ファイルのアドレスの重複チェックを行います。

[詳細説明]

- ROM 化時のエラー・チェックを省略します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

rom_less	rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行いません。 本項目は，ROM レス・モード使用時に指定します。 また，シングルチップ・モード 1 選択時の内蔵 ROM オーバフローのチェックには対応していません。 本項目を指定することにより，内蔵 ROM オーバフローのチェックを無効にしてください。
address	入力ファイル，および出力ファイルのアドレスの重複チェックを行いません。

- *string* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ROM 化オプション］タブの［その他］カテゴリの［ROM 化時のエラーを無視する］

[使用例]

- rompsec セクションに対して，内蔵 ROM 周辺の配置エラー・チェックを行いません。

```
>cx -CF3746 -Xromize_check_off=rom_less main.c
```

（5）ヘキサ出力オプション

ヘキサ出力フェーズのオプションの分類と説明を以下に示します。

表 B―8　ヘキサ出力オプション

分類	オプション	説明
ヘキサ・ファイル出力指定	-Xhex	ヘキサ・ファイル名を指定します。
	-Xhex_only	ヘキサ出力のみ実行します。
ヘキサ変換制御	-Xhex_format	出力するヘキサ・ファイルのフォーマットを指定します。
	-Xhex_fill	ヘキサ・ファイルの充てん処理を指定します。
	-Xhex_section	指定したセクションのコードをヘキサ変換して，出力します。
	-Xhex_block_size	ブロック長の最大値を指定します。
	-Xhex_offset	出力するアドレスのオフセットを指定します。
	-Xhex_null	初期値なしデータのセクションに対して，セクションのサイズ分だけ null 文字を生成します。
	-Xhex_symtab	シンボル・テーブルを変換して，出力します。
	-Xhex_rom_less	ヘキサ・ファイルの充てん時に，内蔵 ROM 領域情報を使用しません。
	-Xcrc	CRC 演算結果を出力します。
	-Xcrc_method	CRC 演算方法を指定します。

ヘキサ・ファイル出力指定

ヘキサ・ファイル出力指定オプションには，次のものがあります。

- -Xhex
- -Xhex_only

-Xhex

ヘキサ・ファイル名を指定します。

[指定形式]

```
-Xhex=file
```

- 省略時解釈

  ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .hex に置き換えた名前（ロード・モジュール・ファイル名の拡張子が .lmf でない場合は，ロード・モジュール・ファイル名に .hex を追加した名前）でヘキサ・ファイルを出力します。

[詳細説明]

- リンク終了後，ヘキサ・ファイルを *file* という名前で出力します。
- *file* にファイル名のみを指定した場合は，ロード・モジュール・ファイルと同じフォルダに，指定したファイル名で出力します。
- *file* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［出力ファイル］カテゴリの［ヘキサ・ファイル出力フォルダ］，［ヘキサ・ファイル名］

[使用例]

- リンク終了後，ヘキサ・ファイル sample.hex を出力します。

```
>cx -CF3746 -Xhex=sample.hex main.c
```

-Xhex_only

ヘキサ出力のみ実行します。

[指定形式]

```
-Xhex_only[=file]
```

- 省略時解釈

  通常どおり，リンク，ROM 化，ヘキサ出力の順で処理を行います。

[詳細説明]

- 入力ファイルとして指定したロード・モジュール・ファイルから，ヘキサ・ファイルを生成して，*file* という名前で出力します。
- *file* にファイル名のみを指定した場合は，ロード・モジュール・ファイルと同じフォルダに，指定したファイル名で出力します。
- *file* を省略した場合は，ロード・モジュール・ファイルと同じフォルダに，ロード・モジュール・ファイル名の拡張子を .hex に置き換えた名前で，ヘキサ・ファイルを出力します。
- 本オプションは，ロード・モジュール・ファイルの生成後，ヘキサ出力のみを行いたい場合に使用します。
  入力ファイルとして，ロード・モジュール・ファイルを指定してしない場合は，エラーとなります。

[使用例]

- ロード・モジュール・ファイル a.lmf から，ヘキサ・ファイル sample.hex を生成します。

```
>cx -CF3746 -Xhex_only=sample.hex a.lmf
```

ヘキサ変換制御

ヘキサ変換制御オプションには，次のものがあります。

- -Xhex_format
- -Xhex_fill
- -Xhex_section
- -Xhex_block_size
- -Xhex_offset
- -Xhex_null
- -Xhex_symtab
- -Xhex_rom_less
- -Xcrc
- -Xcrc_method

-Xhex_format

出力するヘキサ・ファイルのフォーマットを指定します。

[指定形式]

```
-Xhex_format=format
```

- 省略時解釈

  出力するヘキサ・ファイルのフォーマットをインテル拡張ヘキサ・フォーマット（32 ビット・アドレス）とします（-Xhex_format=i の指定と同じです）。

[詳細説明]

- 出力するヘキサ・ファイルのフォーマットを指定します。
- *format* に指定可能なものを以下に示します。

I	インテル拡張ヘキサ・フォーマット（1M バイトまで）
i	インテル拡張ヘキサ・フォーマット（32 ビット・アドレス）（4G バイトまで）
S	モトローラ S タイプ・ヘキサ・フォーマット（スタンダード・アドレス）（16M バイトまで）
s	モトローラ S タイプ・ヘキサ・フォーマット（32 ビット・アドレス）（4G バイトまで）
T	拡張テクトロニクス・ヘキサ・フォーマット（4G バイトまで）

- *format* を省略した場合は，エラーとなります。
- -Xhex_format=T オプションを指定した場合，-Xhex_fill，-Xcrc オプションは無効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。

- ［共通オプション］タブの［よく使うオプション（ヘキサ出力）］カテゴリの［ヘキサ・ファイル・フォーマット］
- ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイル・フォーマット］

［使用例］

- 出力するヘキサ・ファイルのフォーマットをモトローラ S タイプ・ヘキサ・フォーマット（スタンダード・アドレス）とします。

```
>cx -CF3746 -Xhex_format=S main.c
```

-Xhex_fill

ヘキサ・ファイルの充てん処理を指定します。

[指定形式]

```
-Xhex_fill
-Xhex_fill=value
-Xhex_fill=value,start,size
-Xhex_fill=start,size
```

- 省略時解釈

  充てん処理を行いません。

[詳細説明]

- アドレス *start* からサイズ *size* で指定した領域のすべてのコードをヘキサ変換して，出力します。
  指定した領域のうち，未使用領域は *value* で充てんします。
- *value* に指定可能な値の範囲は，0x00 ～ 0xFFFF です。
- *value* は，16 進数で指定します。
- *value* は，1 バイト，または 2 バイト指定が可能です。
  指定した値が 2 桁，または 4 桁に満たない場合は，上位ビットを 0 で補てんします。
- *value* を省略した場合は，0xFF を指定したものとみなします。
- *start* に指定可能な値の範囲は，0x00 ～ 0xFFFFFFFF です。
  start が上記の範囲内であっても，セクションが存在しない場合は，エラーとなります。
- *size* に指定可能な値の範囲は，0x01 ～ 0x100000000 です。
- *start*，*size* は，16 進数で指定します。
- *start*，*size* の指定を省略した場合，デバイス・ファイルで定義された内蔵 ROM 領域のすべてのコードをヘキサ変換して，出力します。
- 本オプションは，-Xhex_format=T オプションと同時に指定することはできません。
- デバイス・ファイルの内蔵 ROM 領域情報を使用しない場合は，-Xhex_rom_less オプションと同時に指定します。
  その場合は，本オプションのパラメータ *start*，*size* を指定する必要があります。
- -Xcrc オプションを指定した場合，-Xcrc オプションのパラメータ *dst*，*start*，*end* を考慮して，最小アドレスを *start*，最大アドレスを *end* とし，最大アドレス＋1 ー最小アドレスを *size* とします。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイルに変換するアドレス範囲を指定する］，［充てん値］，［開始アドレス］，［サイズ］

［使用例］

- アドレス 0x1000 からサイズ 0x2000 分の領域のすべてのコードをヘキサ変換して，出力します。
  その領域のうち，未使用領域は 0x55 で充てんします。

```
>cx -CF3746 -Xhex_fill=0x55,0x1000,0x2000 main.c
```

-Xhex_section

指定したセクションのコードをヘキサ変換して，出力します。

［指定形式］

```
-Xhex_section=section[,section]...
```

- 省略時解釈

  NOBITS 以外のセクション・タイプとセクション属性 A を持つすべてのセクションをヘキサ変換して，出力します。

［詳細説明］

- セクション *section* のコードをヘキサ変換して，出力します。
- *section* が存在しない場合は，エラーとなります。
- *section* を省略した場合は，エラーとなります。
- 本オプションは，-Xhex_fill オプションと同時に指定することはできません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［ヘキサ・ファイルに変換するセクション］

［使用例］

- セクション sec のコードをヘキサ変換して，出力します。

```
>cx -CF3746 -Xhex_section=sec main.c
```

-Xhex_block_size

ブロック長の最大値を指定します。

[指定形式]

```
-Xhex_block_size=num
```

- 省略時解釈

  ヘキサ・フォーマットごとに定められたデフォルト値をブロック長の最大値とします。

[詳細説明]

- *num* に指定した数値をブロック長（インテル拡張ヘキサ・フォーマット，およびモトローラ S タイプ・ヘキサ・フォーマットの場合，1 データ・レコードで示されるコードのバイト数）の最大値とします。
- *num* に指定可能な値の範囲は，ヘキサ・フォーマットごとに異なります。

  各ヘキサ・フォーマットについて，*num* に指定可能な値の範囲を以下に示します。

  指定値が最小値に満たない場合は，警告を出力して，デフォルト値で補正します。

  指定値が最大値を越えている場合は，警告を出力して，最大値で補正します。

  0 を指定した場合は，エラーを出力します。

ヘキサ・フォーマット	*num* の範囲	デフォルト値
インテル拡張	1 ～ 255（0x01 ～ 0xFF）	32（0x20）
インテル拡張（32 ビット・アドレス）	1 ～ 255（0x01 ～ 0xFF）	32（0x20）
モトローラ S タイプ（スタンダード・アドレス）	1 ～ 251（0x01 ～ 0xFB）	80（0x50）
モトローラ S タイプ（32 ビット・アドレス）	1 ～ 250（0x01 ～ 0xFA）	80（0x50）
拡張テクトロニクス	16 ～ 255（0x10 ～ 0xFF）	255（0xFF）

- *num* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［ブロック／レコードの最大長を指定する］，［ブロック／レコードの最大長］

[使用例]

- ブロック長の最大値を 255 とします。

```
>cx -CF3746 -Xhex_block_size=255 main.c
```

-Xhex_offset

出力するアドレスのオフセットを指定します。

[指定形式]

```
-Xhex_offset=num
```

- 省略時解釈

  出力するアドレスにオフセットがないものとみなします。

  0 番地から出力します。

[詳細説明]

- 出力するアドレスを，本来のアドレスにオフセットとして *num* を加えたアドレスとします。
- *num* に指定可能な値の範囲は，0x0 ～ 0xFFFFFFFE です。

  この範囲外の値を指定した場合は，エラーとなります。
- *num* を省略した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［出力アドレスのオフセットを指定する］，［出力アドレスのオフセット］

[使用例]

- 出力するアドレスのオフセットを 0x10000 とします。

```
>cx -CF3746 -Xhex_offset=0x10000 main.c
```

-Xhex_null

初期値なしデータのセクションに対して，セクションのサイズ分だけ null 文字を生成します。

[指定形式]

```
-Xhex_null
```

- 省略時解釈

  すべてのコードをヘキサ変換して，未使用領域は 0xFFFF で充てんして出力します。

[詳細説明]

- セクション・タイプ NOBITS とセクション属性 A を持つセクション（初期値の指定されていないデータに対するセクション，たとえば .bss セクション，および .sbss セクション）に対して，セクションのサイズ分だけ null 文字（¥0）を生成します。
- 本オプションは，-Xhex_fill オプションと同時に指定することはできません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［ヘキサ・フォーマット］カテゴリの［初期値なしデータのセクションをゼロ初期化する］

[使用例]

- セクション・タイプ NOBITS とセクション属性 A を持つセクションに対して，セクションのサイズ分だけ null 文字を生成します。

```
>cx -CF3746 -Xhex_null main.c
```

-Xhex_symtab

シンボル・テーブルを変換して，出力します。

［指定形式］

```
-Xhex_symtab=string
```

- 省略時解釈

  シンボル・テーブルを出力しません。

［詳細説明］

- シンボル・テーブルを変換して，出力します。
- *string* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，エラーとなります。

global	グローバル・シンボルのみ変換します。
all	ローカル・シンボルも含めて変換します。

- *string* を省略した場合は，エラーとなります。
- 本オプションは，-Xhex_format=T オプション指定時のみ有効です。
- 本オプションは，-Xhex_fill オプションと同時に指定することはできません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［シンボル・テーブル］カテゴリの［シンボル・テーブルを変換する］

［使用例］

- シンボル・テーブルを変換して，出力します。

```
>cx -CF3746 -Xhex_symtab -Xhex_format=T main.c
```

-Xhex_rom_less

ヘキサ・ファイルの充てん時に，内蔵 ROM 領域情報を使用しません。

[指定形式]

```
-Xhex_rom_less
```

- 省略時解釈

  本オプションを省略，かつ -Xhex_fill オプション指定時に，パラメータ *start*，*size* を省略した場合，デバイス・ファイルの内蔵 ROM 領域を充てん対象とします。

[詳細説明]

- ヘキサ・ファイルの充てん時に，デバイス・ファイルの内蔵 ROM 領域情報を使用しません。
- 本オプションは，-Xhex_fill オプション指定時に，デバイス・ファイルの内蔵 ROM 領域情報を使用しない場合に指定します。
- 本オプションは，-Xhex_fill オプションと同時に指定する必要があります。

  また，-Xhex_fill オプションのパラメータ *start*，*size* を指定する必要があります。

  -Xhex_fill オプション，およびパラメータ *start*，*size* を省略した場合は，エラーとなります。
- -Xhex_fill オプションのパラメータ *start*，*size* が内蔵 ROM 領域を越えても，警告を出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［その他］カテゴリの［内蔵 ROM 領域オーバフロー時に警告を表示する］

[使用例]

- ヘキサ・ファイルの充てん時に，デバイス・ファイルの内蔵 ROM 領域情報を使用しません。

```
>cx -CF3746 -Xhex_rom_less -Xhex_fill=0xff,0x00,1000 main.c
```

-Xcrc

CRC（Cyclic Redundancy Check）演算結果を出力します。

[指定形式]

```
-Xcrc=dst,start,end[,start,end]...
```

- 省略時解釈

  CRC 演算，および CRC 演算結果の出力を行いません。

[詳細説明]

- 開始アドレス *start* から終了アドレス *end* までの領域を対象に CRC 演算を行い，その結果を出力アドレス *dst* に出力します。

  dst に出力する演算結果は，0x0 ～ 0xFFFF の 16 ビット・データとなります。
- *start*，*end* は，1 組ずつ対にして複数指定することができます。
- *dst*，*start*，*end* に指定可能な値の範囲は，0x0 ～ 0xFFFFFFFF です。

  10 進数，または 16 進数で指定します。

  範囲外の値を指定した場合，および省略した場合は，エラーとなります。
- CRC 演算対象領域にある空き領域は充てんして演算を行うため，本オプションは -Xhex_fill オプションと同時に指定する必要があります。

  -Xhex_fill オプションを指定していない場合は，エラーとなります。

  本オプションと -Xhex_fill オプションの関係を以下に示します。

		-Xhex_fill=*f_value*,*f_start*,*f_size*	
		指定あり	指定なし
-Xcrc=*crc_dst*, *crc_start*, *crc_end*	指定あり	- CRC 演算時の充てん開始アドレスは *crc_start*，*f_start* のうち小さい方とし，充てん終了アドレスは *crc_end*，*f_start* ＋ *f_size* － 1 のうち大きい方とします。 充てん値は，-Xhex_fill オプションで指定した領域の範囲内は *f_value*，それ以外の CRC 演算範囲内は 0xFF とします。 - CRC 演算時の充てん値 0xFF は，ヘキサ・ファイルに出力しません。	- CRC 演算を行いません。 -Xhex_fill オプションを指定していないため，エラーとなります。
	指定なし	- CRC 演算を行いません。 - 充てんを行う領域，および充てん値は，-Xhex_fill オプションのパラメータに従います。	－

- *dst* が *start* から *end* までの領域内にある場合は，エラーとなります。
- *dst* が -Xhex_fill オプションで指定した領域の範囲外にある場合は，エラーとなります。

- *start* に指定した値が *end* に指定した値よりも大きい場合は，エラーとなります。
- *start*，*end* で指定した領域が他の *start*，*end* で指定した領域と重なる場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［CRC 演算］カテゴリの［CRC 演算を行う］，［CRC 結果出力アドレス］，［CRC 演算の範囲］

［使用例］

- 開始アドレス 0x100 ～ 0x200，0x300 ～ 0x400 を対象に CRC 演算を行い，その結果をアドレス 0x12345678 に出力します。

```
>cx -CF3746 -Xcrc=0x12345678,0x100,0x0200,0x300,0x400 main.c
```

-Xcrc_method

CRC 演算方法を指定します。

[指定形式]

```
-Xcrc_method=method[,value]
```

- 省略時解釈

  -Xcrc オプションを指定している場合は，演算方法 HIGH に従って CRC 演算を行います（-Xcrc_method=HIGH オプションの指定と同じです）。

  -Xcrc オプションを指定していない場合は，警告を出力して，本オプションを無視します。

[詳細説明]

- 演算方法 *method* に従って CRC 演算を行います。
- 本オプションは，-Xcrc オプション指定時のみ有効です。
- *method* には HIGH，または GENERAL を指定します。

  HIGH を指定すると高速 CRC（high-speed CRC）用の CRC-16-CCITT による演算結果を，GENERAL を指定すると汎用 CRC（general-purpose CRC）用の演算結果をそれぞれ得ることができます（「高速 CRC」，「汎用 CRC」の詳細については，デバイスのユーザーズ・マニュアルを参照してください）。

  method を省略した場合は，エラーとなります。
- *method* に GENERAL を指定した場合のみ，演算用の初期値 *value* を指定することができます。

  value を指定した場合は，初期値を *value* として CRC 演算を行います。

  value を省略した場合は，0 を指定したものとして演算を行います。
- *value* に指定可能な値の範囲は，0x0 ～ 0xFFFF です。

  10 進数，または 16 進数で指定します。

  範囲外の数値を指定した場合は，エラーとなります。
- *method* に HIGH を指定した場合は，CRC 演算用の初期値は 0 として演算を行います。

  method に HIGH を指定した場合に *value* を指定した場合は，エラーとなります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ヘキサ出力オプション］タブの［CRC 演算］カテゴリの［CRC 演算方法］，［CRC 演算の初期値］

[使用例]

- 演算方法を GENERAL，初期値を 0x1234 として CRC 演算を行います。

```
>cx -CF3746 -Xcrc=0x12345678,0x100,0x0200,0x300,0x400 -Xcrc_method=GENERAL,0x1234 main.c
```

（6）オプションの複数指定

ここでは，オプションを同時に 2 つ以上指定した場合について説明します。

（a）優先順序

- 以下のオプションは，ほかの特定のオプションを無効とします。

-V/-h	ほかのすべてのオプションは無効となります。 このとき，警告を出力しません。
-P	プリプロセス処理で終了するため，それ以降の処理に関するオプションは無効となります。 このとき，警告を出力します。
-S	コード生成処理で終了するため，それ以降の処理に関するオプションは無効となります。 このとき，警告を出力します。
-c	アセンブル処理で終了するため，それ以降の処理に関するオプションは無効となります。 このとき，警告を出力します。
-Xswitch	パラメータに table 以外を指定した場合，-Xword_case オプションは無効となります。 このとき，警告を出力します。
-Xrelinkable_object	-Xlink_output，-Xstartup，ROM 化オプション（-Xno_romize 以外），ヘキサ出力オプション（-Xhex_only 以外）は無効となります。 このとき，警告を出力します。
-Xno_romize	その他の ROM 化オプション，および -Xlink_output は無効となります。 このとき，警告を出力します。
-Xhex_only	ヘキサ出力オプション以外は無効となります。 また，ヘキサ出力オプションのうち，-Xhex も無効となります。 このとき，警告を出力します。
-Xhex_format	パラメータに T を指定した場合，-Xhex_fill は無効となります。 このとき，警告を出力します。
-Xhex_fill	-Xhex_section/-Xhex_symtab/-Xhex_null は無効となります。 このとき，警告を出力します。
-Xmulti	-I，-L，-Xstartup，-Xsecurity_id，-Xoption_byte，-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata，ヘキサ出力オプション（-Xhex_only 以外）は無効となります。 このとき，警告を出力します。
-Xmulti_link	-Xsfg，-Xsfg_opt，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata は無効となります。 このとき，警告を出力します。

- 以下の組み合わせでオプションを指定した場合は，警告を出力して，最後に指定したものが有効となります。
  - -P，-S，-c
  - -D，-U（シンボル名が同じ場合）
  - -Onothing，-Odefault，-Osize，-Ospeed
  - -Xinline_strcpy，-Xcall_lib
  - -Xstartup，-Xno_startup

なお，オプションの指定順序により，以下のオプションは無効となります。

- -Onothing，-Odefault，-Osize の前に指定した -O*item* 注
- -Ospeed の前に指定した -O*item* 注，-Xpro_epi_runtime

注 -O*item*：-Ounroll，-Oinline，-Odelete_static_func，-Opipeline

（b）依存関係

以下のオプションは，ほかの特定のオプションにより動作が変わります。

-C	-C オプションを指定しない場合，通常はエラーとなりますが，-Xcommon/-V/-h/-P オプションのいずれかを指定した場合はエラーとなりません。
-Xcommon	-C オプションと同時に指定した場合，振る舞いが変わることがあります。 詳細については，「-Xcommon」を参照してください。
-Xpreprocess	-P オプションと同時に指定しない場合，無効となります。 このとき，警告を出力しません。
-Xpass_source	-S，または -Xasm_path オプションと同時に指定しない場合，無効となります。 このとき，警告を出力しません。
-Xalign_fill	-Xtwo_pass_link オプションと同時に指定しない場合，エラーとなります。
-Xhex_fll	-Xhex_rom_less オプションと同時に指定する場合，パラメータ *start*，および *size* の指定が必要です。 これらのパラメータを指定しない場合，エラーとなります。
-Xflash_ext_table	-Xflash/-Xlink_output/-Xno_romize オプションのいずれか 1 個以上と同時に指定しない場合，エラーとなります。
-Xflash	-Xflash_ext_table と同時に指定しない場合，エラーとなります。
-Xsfg	-Xcref オプションと同時に指定しない場合，エラーとなります。
-Xsfg_opt	-Xsfg オプションと同時に指定しない場合，無効となります。 このとき，警告を出力します。
-Xsfg_size_tidata -Xsfg_size_tidata_byte -Xsfg_size_sidata -Xsfg_size_sedata -Xsfg_size_sdata	-Xsfg_opt オプションと同時に指定しない場合，無効となります。 このとき，警告を出力します。

-o	-P/-S/-c オプションと同時に指定した場合，生成ファイルの種別は，それぞれプリプロセス処理済みファイル／アセンブラ・ソース・ファイル／オブジェクト・モジュール・ファイルとなります。
-g	-O オプションと同時に指定した場合，デバッグ情報が正しくないことがあります。
-Oinline -Xdelete_func	-Xintermodule オプションと同時に指定した場合，それぞれの最適化の効果が変わります。
-Xcrc	-Xhex_fill オプションと同時に指定しない場合，エラーとなります。
-Xcrc_method	-Xcrc オプションと同時に指定しない場合，無効となります。 このとき，警告を出力します。

B. 1. 4　シンボル情報ファイル

シンボル情報ファイルとは，C ソース・ファイル内で定義した変数（グローバル変数，ファイル内 static 変数，関数内 static 変数）の配置情報，および変数，関数の参照回数などの情報を記述したテキスト形式のファイルです。

（1）シンボル情報ファイルに関する機能

cx は，シンボル情報ファイルに関して，以下の 2 つの機能を提供しています。

- シンボル情報ファイルの参照

  コンパイル時にシンボル情報ファイルを参照することにより，C ソース・ファイルを修正することなく，外部のファイルから変数の配置に関する指定（配置セクションの変更）を行うことができます。

  また，-Xdelete_func オプションを指定することにより，参照回数が 0 でない関数を削除しないようにすることもできます。

  参照するシンボル情報ファイルは，ユーザが直接編集して作成することができますが，cx を使用して生成することも可能です。

- シンボル情報ファイルの生成

  C ソース・ファイル内で定義した変数，および関数の最適な配置情報を記述したシンボル情報ファイルを自動的に生成することができます。

  生成したシンボル情報ファイルは，必要に応じてユーザが直接編集することもできます。

注意　cx は，コンパイル時にシンボル情報ファイルを参照する際，変数の配置情報と関数の参照回数のみ利用します。
cx を使用してシンボル情報ファイルを自動生成する場合，関数の配置情報も出力しますが，それを利用して関数の配置に関する指定を行うことはできません。

（2）シンボル情報ファイルへの出力情報

cx を使用してシンボル情報ファイルを自動生成する場合，C ソース・ファイル内で定義した変数，および関数の最適な配置情報を出力します。

- 出力対象となるのは，グローバル変数，ファイル内 static 変数，関数内 static 変数，および関数です。
- 変数，および関数の参照回数やサイズなどの情報を，それぞれ利用頻度の高い順に出力します。
- -Xsfg_opt オプションの指定により，セクション単位で変数の最適な配置情報を出力することができます。

  利用頻度の高い順に，.tidata.byte，.tidata.word，.sidata，.sedata，.sdata セクションのサイズ内に配置できるように，変数を振り分けます。

  なお，-Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata オプションの指定により，各セクションのサイズを指定することもできます。

備考 1.　変数の利用頻度は，アセンブラ・ソース・レベルでの参照回数とサイズ（バイト数）から 1 バイトあたりの参照回数を算出したものとなります。

また，関数の利用頻度は，C ソース・ファイル内での参照回数とサイズ（バイト数）から 1 バイトあたりの参照回数を算出したものとなります。

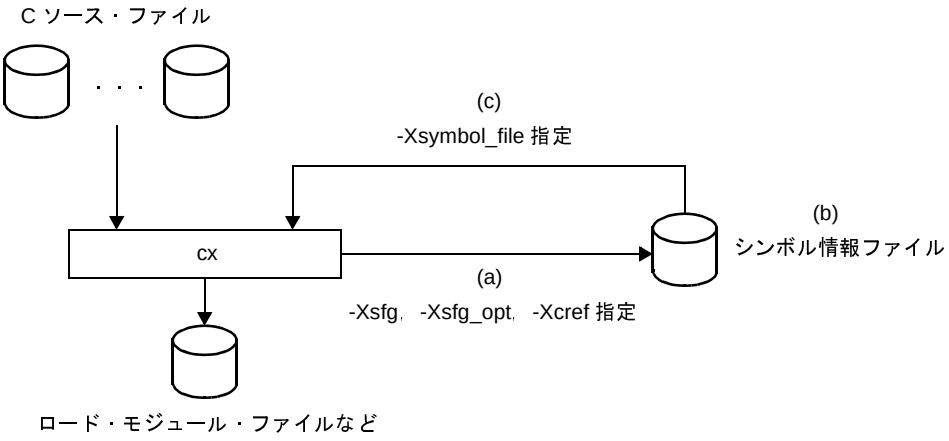
2. 各オプションについての詳細は，「情報ファイル出力制御」を参照してください。

（3）シンボル情報ファイルの利用方法

cx を使用してシンボル情報ファイルを生成し，それを参照するには，cx コマンドを 2 回実行する必要があります。

1 回目の実行でシンボル情報ファイルを生成して，2 回目の実行で生成したシンボル情報ファイルを参照します。

図 B―3　シンボル情報ファイルを利用した処理の流れ



コマンド・ラインでの操作方法を以下に示します。

備考　CubeSuite+ での操作方法については，「2. 13　変数を最適なセクションに配置する」を参照してください。

（a）シンボル情報ファイルの生成

-Xsfg オプションを指定して cx コマンドを実行することにより，シンボル情報ファイルを生成します。

例　シンボル情報ファイル symbol.sfg を生成します。

```
>cx -CF3746 -Xsfg=symbol.sfg -Xsfg_opt -Xcref=info.cref file.c
```

注意 1.　セクション単位で変数の最適な配置情報を出力するためには，-Xsfg_opt オプションも指定してください。
変数の配置情報は指定せず，未使用関数の削除のみを行う場合は，-Xsfg_opt オプションを指定する必要はありません。

2. **シンボル情報ファイルはC ソース・ファイルの静的解析結果に基づいて生成します。**
   そのため，-Xsfg オプションを指定する際は，-Xcref オプションも同時に指定する必要があります。
3. **シンボル情報ファイルはリンク時の情報を参照しているため，リンクが正常にできない場合は生成することはできません。**

備考 -Xsfg_size_tidata，-Xsfg_size_tidata_byte，-Xsfg_size_sidata，-Xsfg_size_sedata，-Xsfg_size_sdata オプションの指定により，各セクションのサイズを指定することもできます。
各オプションについての詳細は，「情報ファイル出力制御」を参照してください。

（b）シンボル情報ファイルの編集

シンボル情報ファイルはテキスト形式なので，エディタなどで編集することにより，変数の配置情報（配置セクションの変更）を変更することができます。
必要に応じてシンボル情報ファイルを修正してください。

備考 シンボル情報ファイルのフォーマットについては，「3. 3 シンボル情報ファイル」を参照してください。

（c）シンボル情報ファイルの参照

生成したシンボル情報ファイルを -Xsymbol_file オプションで指定して，cx コマンドを再度実行することにより，指定したシンボル情報ファイルの内容に従って，コンパイルを行います。

例 シンボル情報ファイル symbol.sfg の内容に従って，コンパイルを行います。

```
>cx -CF3746 -Xsymbol_file=symbol.sfg file.c
```

B. 1. 5　最適化機能

ここでは，cx が実行する最適化機能について説明します。
なお，最適化は，C ソース・ファイルを対象としています。

（1）概要

cx では，以下の 2 つを目的として，最適化を行います。

- 実行速度の短縮（生成コードの実行速度を速くする）
- オブジェクト・サイズの削減（生成コードの使用する ROM/RAM 容量を小さくする）

最適化項目の多くは上記の両方を改善しますが，項目によっては相反する場合もあります（実行速度は速くなるが ROM サイズが増加する場合など）。

また，最適化を行うことにより，C ソース行と機械語命令との対応が複雑になるためにブレーク・ポイントが設定できなくなったり，変数の値の参照／設定が C ソースに記述した位置で行われなくなる場合があり，デバッグのしやすさに影響が出ることがあります。

そのため，cx では，以下の 4 つの最適化レベルを用意しています。

表 B—9　最適化レベル

最適化レベル	説明
デバッグ優先	デバッグのしやすさを重視して，デフォルトで実行する最適化を含むすべての最適化を抑止します。
デフォルト	デバッグに影響しない範囲の最適化（式の最適化，およびレジスタ割り付けなど）を行います。
オブジェクト・サイズ優先	ROM/RAM 容量の削減を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。
実行速度優先	実行速度の短縮を重視して，一般的なプログラムに対して有効な最大限の最適化を行います。

cx で実行する最適化を以下に示します。

表 B—10　最適化項目

項目	説明
式の最適化	定数演算や式の変形を行います。
制御フロー最適化	分岐命令が減るように命令を並び替えます。
複写の伝播	変数が参照されている箇所を，その変数に格納されている値や別の変数参照で置き換えます。
共通部分式の認識	一度計算した値を保持し，同じ計算が複数行われる場合に再利用します。
不要な命令の削除	結果が利用されない演算や代入を削除します。
ループ展開	for/while などのループを指定された回数分展開します。

項目	説明
ループ最適化	実行回数 0 回，または 1 回のループの削除，ループ制御変数を含む式の最適化などを行います。
ループ不変式の移動	ループ内で値が変わらない計算をループの前に移動します。
関数のインライン展開	呼び出し箇所に関数を展開します。 呼び出す関数と呼び出される関数は，同じソース・ファイル内で定義する必要があります。 ただし，ファイル間最適化を同時に指定した場合は，異なるソース・ファイル間で定義することが可能です。
未使用ファイル内 static 変数の削除	使用されないファイル内 static 変数を削除します。 使用されない関数内 static 変数は削除しません。
未使用 static 関数の削除	呼び出されない static 関数を削除します。
標準ライブラリ関数のインライン展開	標準ライブラリ関数 strcpy，strcmp，memcpy，memset の呼び出しをインライン展開します。
関数のプロローグ／エピローグ処理のランタイム・ライブラリ呼び出し	関数のプロローグ／エピローグ処理を行う命令列を生成する代わりに，同等の処理を行うランタイム・ライブラリを呼び出す命令を生成します。 その結果，コード・サイズは小さくなりますが，実行速度はやや遅くなります。
パイプライン最適化【V850E2V3】	V850E2V3 のアーキテクチャの性能を引き出すため，パイプラインを有効に利用するよう，機械語レベルで命令の並べ替えを行います。
外部変数のソート	const/sconst セクション以外のセクションに配置されている外部変数を，アライメントの大きい順に並び替えます。 その結果，アライン・ホールが減って，必要な RAM 容量を削減します。

備考 最適化オプションについては，「最適化指定」を参照してください。

(2) 最適化によるデバッグへの影響

最適化を行うと，デバッグ時に以下のような影響があるので，注意してください。

- 最適化による式の変形（複写の伝播や共通部分式の認識）によって，C ソース・プログラム中での出現箇所で“変数参照”が起こらなくなり，変数の read ／ write イベントがユーザの意図どおり発生しない場合があります。
- 文の共通化や削除，並び替えが行われると，ステップ実行やブレークポイントがユーザの意図どおりに設定できないことがあります。
- 変数の生存範囲（プログラム中でその変数を参照可能な範囲），変数の位置（レジスタやメモリ上の位置）が変更される可能性があります。
- 文の削除が起こった場合，その文にはブレークポイントを設定することができません。
- 文の移動や分割／統合により，実行命令の順序が入れ替わった[注]場合，入れ替わった行とそれらの行の間にある行は 1 つのまとまりとして扱われ，途中のブレークポイントの設定やステップ実行ができなくなる場合があります。

注 あるソース行に対する実行命令のアドレスが，それ以前の行に対する実行命令のアドレスより小さくなった，あるいは，それ以後の行に対する実行命令のアドレスより大きくなったことをいいます。

- if-else の各場合で実行命令の順序が入れ替わったり，ループ展開で実行命令の順序が入れ替わった場合，ステップ実行などができなくなる場合があります。
- 自動変数はすべて有効範囲（スコープ）が関数全体とみなします。
  しかし，その変数がレジスタへ割り付けられた場合，スコープ内であっても最適化により削除され，見えなくなる可能性があります。
  これは，スコープ内でその変数が“局所的に”使用されている場合や，最適化の結果として局所化された場合に起こります。

例

```
void f(void)
{
    int     a;      /* 関数内で有効 */
        :
    /* アドレス1 */
        :           /* aはアドレス1～アドレス2の範囲でのみ使用 */
    /* アドレス2 */
        :
}
```

この例では，a のスコープは関数 f() 内全体です。
しかし，a はアドレス 1～アドレス 2 間に限定して使用されています。このとき，a がレジスタに割り付けられ，最適化によりスタック・フレームから削除されると，a はアドレス 1～アドレス 2 の区間外では見えなくなります。
この現象は，レジスタを効率的に使用するために，a の見える区間外では，a が割り付けられたレジスタにほかの変数を割り付ける結果として生じます。

- コンパイル時にデバッグ情報の処理でメモリを大量に消費するため，“out of memory”となる可能性があります。
- インライン展開が行われた部分は 1 つのまとまりとして扱われ，ステップ実行はできません。
- 変数の値を参照した際，正しい値ではなく，計算途中の一時的な値が得られることがあります。
- 配列の一部，構造体の要素，ユーザ定義型のポインタ変数がレジスタに割りつけられた場合，デバッグ・ツールのウォッチ パネル等にて変数の表示／変更が不正になることがあります。

(3) 最適化に関する注意事項

最適化に関する注意事項を以下に示します。

- C プログラム中のアセンブラ記述（__asm 宣言，および #pragma asm ～ #pragma endasm），および以下の組み込み関数をまたぐ最適化，およびレジスタ割り付けは行いません。

  __DI，__EI，__set_il，__nop，__halt，__ldsr，__stsr，__ldgr，__stgr

- 未使用の static 関数，およびファイル内 static 変数は，デフォルトで削除します。
  ただし，-Odelete_static_func オプションを指定することにより，未使用の static 関数の削除を抑止することができます。
- -Oinline=1，または 2 を指定した場合でも，#pragma inline 指定した関数が必ずしもインライン展開されるわけではありません。
  #pragma inline は，C 言語における必ず有効な指定ではなく，コンパイラに対する参考情報であり，register/__inline キーワードと同様，内容やコンパイル状況により展開されないこともあります。

B. 1. 6　ブートーフラッシュ再リンク機能

（1）概要

システムによっては，フラッシュ領域や，着脱可能な ROM を搭載していることがあります。

フラッシュ領域の場合は，書かれている内容を書き換えたり，着脱可能な ROM の場合は，新しく書き換えた ROM 自体を取り替えることによって，プログラムのバージョン・アップ等を行います。

プログラムの一部でも変更する場合，基本的にプロジェクトそのものを再構築，つまり，リビルドして生成し直すことになります。

しかし，バージョン・アップしたい箇所が，フラッシュ領域や外付け ROM だけに限られている場合は，再構築しないで済むと便利です。

また，ブート部分は内蔵 ROM などに固定され，書き換え対象のフラッシュ領域との間に関数呼び出しがある場合，フラッシュ領域内の関数を修正することにより，関数の先頭アドレスがずれてしまうと，関数呼び出しを正常に行うことができなくなってしまいます。

このような状況を防ぎ，正常な関数呼び出しの実現をするのが“ブートーフラッシュ再リンク機能”（以下“再リンク機能”）です。

実現方法の概略は，以下のようになります。

（a）フラッシュ領域に，フラッシュ領域内の関数群への分岐命令が書かれている“分岐テーブル”を用意する

（b）ブート領域から，フラッシュ領域内の関数をコールするとき，いったんフラッシュ領域の分岐テーブルへジャンプし，その後，目的の関数への分岐命令を実行してジャンプする

これらの仕組みは，ユーザが用意して実現することもできますが，この“再リンク機能”を用いると比較的簡単に実現することができます。

ただし，この機能を使用する上では，ブート領域側を作成した時点で，フラッシュ領域側の呼び出す関数が決定している必要があります。

あくまでも，フラッシュ領域側の関数に変更があっても，ブート領域側からその関数を問題なく呼び出すことができるようにする仕組みです。

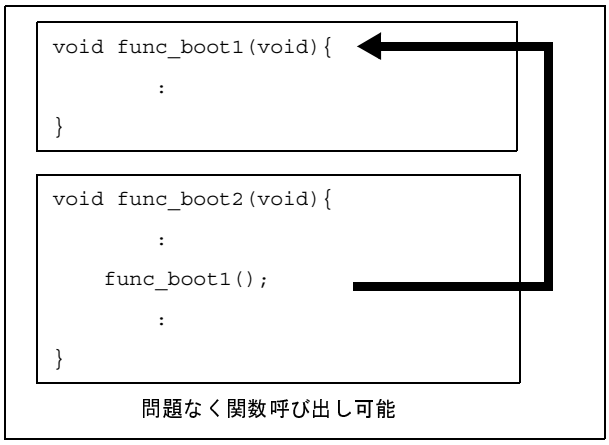
(2) 再リンク機能のイメージ

再リンク機能を利用したときの関数呼び出しのイメージは，以下のようになります。

(a) ブート領域内からブート領域内の関数を呼び出すとき

ブート領域に書き込む前に，すでにアドレス解決ができているので，問題なく関数呼び出しが可能です。

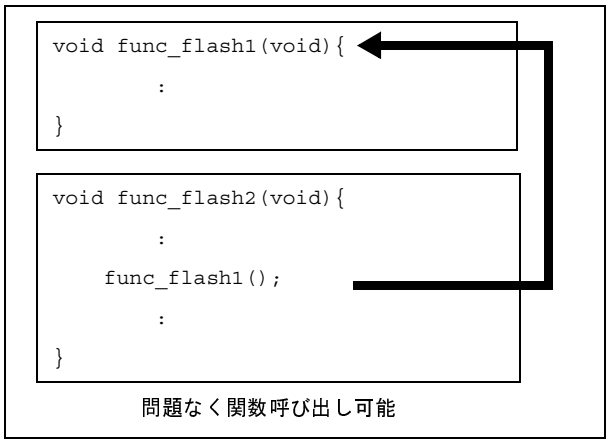
図 B—4 ブート領域内



(b) フラッシュ領域内からフラッシュ領域内の関数を呼び出すとき

フラッシュ領域内ではアドレス解決ができているので，問題なく関数呼び出しが可能です。

図 B—5 フラッシュ領域内



(c) ブート領域内からフラッシュ領域内の関数を呼び出すとき

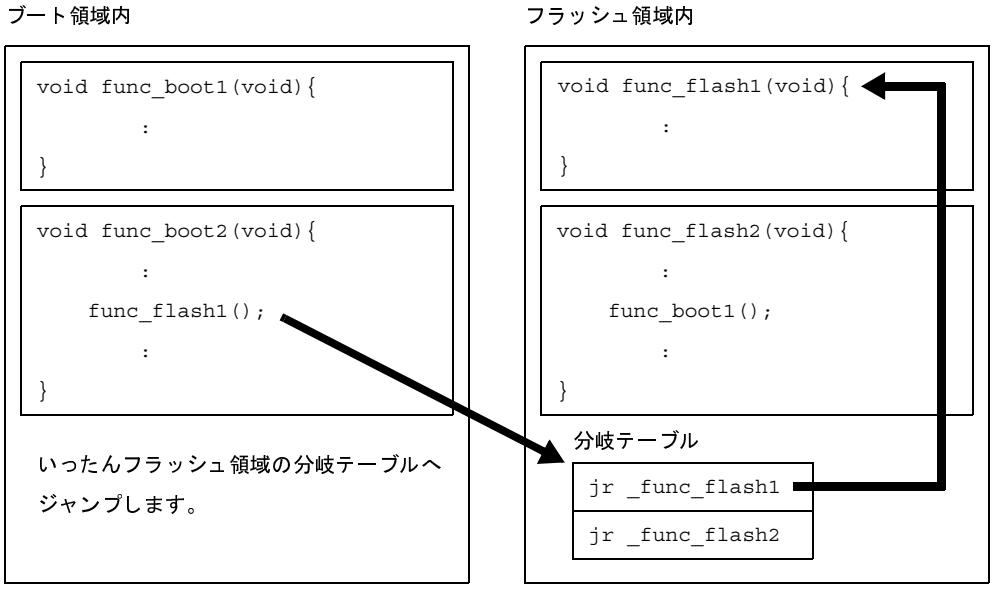
ブート領域内からフラッシュ領域内にある関数を呼び出すとき，フラッシュ領域内の関数サイズ等の変更により，ブート領域内からはアドレスを特定することができません。

つまり，フラッシュ領域内の関数を直接呼び出すことができません。

これを解決するため，いったんフラッシュ領域内の分岐テーブルへジャンプします。

次に，そのテーブルから該当する関数へのジャンプ命令を実行して，目的の関数へジャンプします。

図 B—6　ブート領域内からフラッシュ領域内



また，関数と同様に，外部変数の参照の可否にも関係します。

フラッシュ領域内に定義されているグローバル変数は，ブート領域内から参照することはできません。

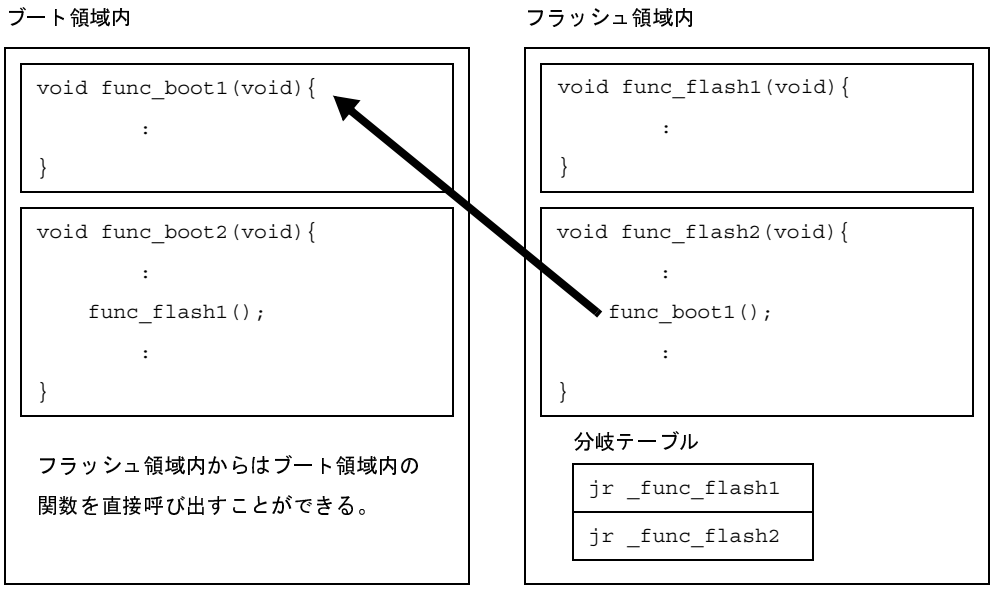
そのため，ブート領域内，フラッシュ領域内それぞれで同じ名前の外部変数を定義することができます。

その外部変数に対する参照は，それぞれの領域内からの参照のみとなります。

（d）フラッシュ領域内からブート領域内の関数を呼び出すとき

フラッシュ領域内からブート領域内にある関数を呼び出すとき，ブート領域内の内容は変わらないので，フラッシュ領域内からブート領域内にある関数を直接呼び出すことができます。

図 B—7　フラッシュ領域内からブート領域内



また，関数と同様に，外部変数の参照の可否にも関係します。

ブート領域内に定義されているグローバル変数は，フラッシュ領域内から参照することができます。

(3) 再リンク機能の実現方法

再リンク機能を実現する場合，“ブート領域側”と“フラッシュ領域側”を別々に作成します。

つまり，一度ブート領域側を作成したあと（ROM に書き込んだのち）は，フラッシュ領域側だけを変更することになります。

そのため，CubeSuite+ でプロジェクトを作成するときは，以下のように分けて作成してください。

- ブート領域側に配置するプロジェクト
- フラッシュ領域側に配置するプロジェクト（今後変更することがあるプロジェクト）

また，スタートアップ・ルーチン，およびリンク・ディレクティブ・ファイルも，それぞれの領域用に別々に用意します。

(a) $ext_func 制御命令の指定

ブート領域側からフラッシュ領域側の関数を呼び出したい場合，まず，ブート領域側に $ext_func 制御命令を使用して，呼び出す関数名（ラベル名）と ID 値を指定します。

$ext_func 制御命令の書式を以下に示します。

```
$ext_func 関数名 , ID 値
```

ブート領域側に $ext_func 制御命令を使用してフラッシュ領域側にある関数名を指定すると，分岐テーブル（ext_table）を作成します。

この分岐テーブルのアドレス（先頭シンボルは __ext_table_head となります）は，ブート領域側のロード・モジュール・ファイルを作成するとき，およびフラッシュ領域側のロード・モジュール・ファイルを作成するときに，-Xflash_ext_table オプションで指定します。

なお，リンク・ディレクティブ・ファイルの記述との関連により，分岐テーブルは，フラッシュ領域の先頭に配置することを推奨します。

ただし，内蔵 ROM がフラッシュ領域の場合は，先頭が INT セグメントであるため，その後ろに配置することになります。

ID 値は正数で指定します。

なお，同じ関数名で異なる ID 値を指定したり，異なる関数名に同じ ID 値を指定することはできません。

分岐テーブルのサイズ（単位：バイト）は，

（指定した ID 値の最大値＋ 1）×分岐テーブルのエントリ・サイズ

となります。

これを考慮して，分岐テーブルのアドレスを設定する必要があります。

考慮しない場合は，分岐テーブルとほかのセクションが重なり，エラーとなることがあります。

関数本体へ分岐するときは，作成した分岐テーブルの先頭から ID 値によるオフセット参照をすることによって実際の関数アドレスを取得し，そして分岐することになります。

例えば，関数 func_flash0，func_flash1，func_flash2 がフラッシュ領域に配置されていて，これらをブート領域側から呼び出したい場合，ブート領域側には以下のように記述します。

- C 言語で記述する場合

```
void dummy() {
#pragma asm
$ext_func _func_flash0, 0
$ext_func _func_flash1, 1
$ext_func _func_flash2, 2
#pragma endasm
}
```

- アセンブリ言語で記述する場合

```
$ext_func _func_flash0, 0
$ext_func _func_flash1, 1
$ext_func _func_flash2, 2
```

なお，これらの $ext_func 制御命令群の記述は，記述漏れやソース間の矛盾が生じることを防ぐため，つまり，同じ関数名で異なる ID 値を指定したり，異なる関数名に同じ ID 値を指定する，というような間違いを防ぐため，1 つのファイルにまとめて，すべてのソースに $include 制御命令（C 言語で記述する場合は #include 命令）でインクルードすることを推奨します。

再リンク機能のイメージを以下に示します。

ユーザが記述するアセンブラ・ソース	リンク後のアセンブラ・イメージ
[ext_table.inc] $ext_func _func_flash0, 0 $ext_func _func_flash1, 1 $ext_func _func_flash2, 2	
[boot.asm] $include(ext_table.inc) .extern _func_flash0 .extern _func_flash1 .extern _func_flash2 jarl _func_flash0, lp jarl _func_flash1, lp jarl _func_flash2, lp	[boot.lmf] .extern __ext_table_head jarl __ext_table_head+0x4*0,lp jarl __ext_table_head+0x4*1,lp jarl __ext_table_head+0x4*2,lp

<table>
<tr><th>ユーザが記述するアセンブラ・ソース</th><th>リンク後のアセンブラ・イメージ</th></tr>
<tr><td rowspan="2">[flash.asm]
$include(ext_table.inc)
.public _func_flash0
.public _func_flash2
_func_flash0:
:
jmp [lp]

.public _func_flash1
_func_flash1:
:
jmp [lp]

_func_flash2:
:
jmp [lp]</td><td>[flash.obj]
#（分岐テーブル）
.ext_table .cseg text
.public __ext_table_head
.extern _func_flash0
.extern _func_flash1
.extern _func_flash2
__ext_table_head:
jr _func_flash0
jr _func_flash1
jr _func_flash2</td></tr>
<tr><td>#（関数本体）
.public _func_flash0
_func_flash0:
:
jmp [lp]

.public _func_flash1
_func_flash1:
:
jmp [lp]

.public _func_flash2
_func_flash2:
:
jmp [lp]</td></tr>
</table>

上記のように $ext_func 制御命令を指定すると，ext_table というシンボルでテーブルが生成され，その先頭シンボルが“__ext_table_head”になります。

ブート領域側の“jarl _func_flash0, lp”というコードは，__ext_table_head からのオフセットで _func_flash0 のアドレスを取得し，jarl 命令で関数本体へジャンプします。

（b）スタートアップ・ルーチンの用意

ブート領域側のスタートアップ・ルーチンとフラッシュ領域側のスタートアップ・ルーチンは，それぞれに用意します。

それぞれのスタートアップ・ルーチンで必要な処理を，以下に示します。

- ブート領域側で tp，gp，ep 値をセットする
- ブート領域側で使用する RAM 領域を初期化するため，_rcopy 関数を呼び出す
- ブート領域側からフラッシュ領域側のスタートアップ・ルーチンへ分岐する
- フラッシュ領域側で使用する RAM 領域を初期化するため，_rcopy 関数を呼び出す
- フラッシュ領域側の処理へ移行

備考 1. tp，gp，ep をブート領域内で使用しない場合は，値のセットをフラッシュ領域で行っても問題ありません。
また，ROM 化処理を行わない場合は，_rcopy 関数の呼び出しは不要です。

2. tp，gp，ep 値は，ブート領域側，およびフラッシュ領域側で同じアドレス値を使用してください。
異なる値にすることも可能ですが，その場合は，ブート領域側とフラッシュ領域側の命令コードを行き来するたびに，各値を設定し直す必要があります。

（c）リンク・ディレクティブ・ファイルの用意

ブート領域側のリンク・ディレクティブ・ファイルとフラッシュ領域側のリンク・ディレクティブ・ファイルは，それぞれに用意します。

リンク・ディレクティブ・ファイルを記述する際の注意事項を以下に示します。

- ブート領域に置くセクションのアドレスは，ブート領域側とフラッシュ領域側でオーバラップしていても，プロジェクトが異なるため，リンク時にエラーを出力することができません。
つまり，セクションのオーバラップが可能です。
ただし，最後に RAM 領域に書き込んだ側だけが有効となるため，ブート領域側とフラッシュ領域側で同時に参照する必要のある RAM 領域は，オーバラップしないようにアドレス指定する必要があります。
- tp，gp，ep 値は，ブート領域側，およびフラッシュ領域側で同じアドレス値を使用してください。
異なる値にすることも可能ですが，その場合は，ブート領域側とフラッシュ領域側の命令コードを行き来するたびに，各値を設定し直す必要があります。
- 分岐テーブル（ext_table）の配置は，リンク・ディレクティブに記述する必要はありません。
-Xflash_ext_table オプションで指定したアドレスに自動的に配置します。
ただし，以下の点に注意が必要です。
  - -Xflash_ext_table オプションで指定したアドレスに，分岐テーブルのサイズ分の空き領域があった場合は，そのまま配置します。
  ほかのセグメントへの影響はありません。
  これが最も理想的なケースです。
  - -Xflash_ext_table オプションで指定したアドレスに，分岐テーブルのサイズ分の空き領域がなかった場合は，エラーとなります。
  たとえば，-Xflash_ext_table オプションで指定したアドレスは“アドレス指定された TEXT セグメント内”で，すでにコードが配置されている場合などです。
  これに該当する例は，以下のようになります。

分岐テーブルのアドレス指定

```
-Xflash_ext_table=0x300
```

リンク・ディレクティブ・ファイル（の一部）

```
TEXT : !LOAD ?RX V0x400{
        .pro_epi_runtime = $PROGBITS ?AX;
        .text            = $PROGBITS ?AX;
};
```

（TEXT セグメントのサイズが 0x100 バイト以上あるとします）

このとき，分岐テーブルは 0x400 番地に割り付けることができないため，リンク時にエラーとなります。

-Xflash_ext_table オプションで指定する値を変更してください。

- 再リンク機能を使用する前は，-Xfalsh_ext_table オプションで指定したアドレスにはほかのセグメントが割り当てられますが，リンク・ディレクティブに，そのセグメントのアドレス指定がなかった場合は，-Xflash_ext_table オプションで指定したアドレスには分岐テーブルが配置され，元々あったセグメントは，分岐テーブルの後ろに移動します。
ただし，セグメントがずれたことにより，さらに後ろの“アドレス指定されたセグメント”に重なった場合は，エラーとなります。

分岐テーブルのアドレス指定

```
-Xflash_ext_table=0x300
```

リンク・ディレクティブ・ファイル（の一部）

```
TEXT : !LOAD ?RX{
        .pro_epi_runtime = $PROGBITS ?AX;
        .text            = $PROGBITS ?AX;
};
```

（TEXT セグメントよりも前のセグメントから続きで，0x300 番地から TEXT セグメントが割り当てられていたとします）

このとき，TEXT セグメントにはアドレス指定がないため，分岐テーブルは 0x300 番地に割り付けられ，TEXT セグメントは分岐テーブルの後ろに割り付けられます。

（d）$ext_ent_size 制御命令の指定

フラッシュ領域内にある分岐テーブルから実際の関数を呼び出すとき，デフォルトでは，以下のようにjr 命令による分岐命令を生成します。

```
__ext_table_head:
        jr      _func_flash0
        jr      _func_flash1
        jr      _func_flash2
```

しかし，jr 命令では，22 ビット範囲内（± 1M バイト範囲内）にしか分岐することができません。

それ以上のアドレス，つまり，32 ビット空間すべてに分岐できるようにしたい場合は，$ext_ent_size制御命令を追加指定します。

$ext_ent_size 疑似命令の書式を以下に示します。

```
$ext_ent_size エントリ・サイズ
```

エントリ・サイズに指定可能な値は，4，または 8（V850Ex コアの場合）です。

エントリ・サイズとは，1 つの分岐処理に必要な命令サイズです。

デフォルトは 4 で，以下のように 4 バイト命令を配置します。

```
jr      _flash_func0    -- 4 バイト命令
```

8 を指定すると，以下のように合計で 8 バイトの命令を配置します。

```
mov     #_flash_func0, r1       -- 6 バイト命令
jmp     [r1]                    -- 2 バイト命令
```

なお，エントリ・サイズは，すべてのソースで唯一の値を持つものとします。

ソースごとに異なる値を指定した場合は，エラーとなります。

（e）ライブラリに対する $ext_func 制御命令の指定

ブート領域，またはフラッシュ領域からライブラリ関数を呼び出していた場合，ライブラリは呼び出した側のオブジェクトにリンクします。

たとえば，フラッシュ領域側にライブラリをリンクしていても，ブート領域からも同じライブラリ関数を呼び出していた場合は，ブート領域にも同じライブラリをリンクします。

つまり，ライブラリ関数を呼び出す場合は，ブート領域とフラッシュ領域との間で分岐は起こらないため，ライブラリ関数に対して $ext_func 制御命令で関数を指定する必要はありません。

ただし，ブート領域側にリンクされたライブラリがフラッシュ領域側の関数へ分岐する，というような特殊な場合に関しては，$ext_func 制御命令で関数を指定する必要があります。

なお，CX に添付されているライブラリに関しては，$ext_func 制御命令で関数を指定する必要はありません。

(f) 割り込みハンドラに対する $ext_func 制御命令の指定

割り込みハンドラの呼び出し部分は，割り込みハンドラ・アドレスのある領域側に記述してください。

以下の場合，割り込みハンドラ関数名に対しても，$ext_func 制御命令で関数を指定する必要があります。

- 割り込みハンドラ・アドレスがブート領域側の場合
- 割り込みハンドラ本体がフラッシュ領域側の場合

ユーザが記述するアセンブラ・ソース	リンク後のアセンブラ・イメージ
[ext_table.inc] $ext_func _int_flash0, 0	
[boot.asm] $include(ext_table.inc) .extern _int_flash0 .cseg TEXT jr _int_flash0	[boot.lmf] .cseg TEXT jr __ext_table_head+0x4*0,lp
[flash.asm] $include(ext_table.inc) .public _int_flash0 _int_flash0: : reti	[flash.lmf] #（分岐テーブル） .cseg TEXT .public __ext_table_head .extern _int_flash0 __ext_table_head: jr _int_flash0 #（ハンドラ本体） .public _int_flash0 _int_flash0: : reti

(g) 操作方法

コマンド・ラインでの操作方法を以下に示します。

備考 CubeSuite+ での操作方法については，「2. 12 ブート－フラッシュの再リンク機能を実現するための準備をする」を参照してください。

- ブート領域側のロード・モジュール・ファイルの生成

ブート領域側のスタートアップ・ルーチン cstartN_b.obj を生成します。

```
>cx -CF3746 cstartN_b.asm
```

スタートアップ・ルーチン cstartN_b.obj，リンク・ディレクティブ・ファイル directive_b.dir をリンクして，ブート領域側のロード・モジュール・ファイル boot.lmf を生成します。

フラッシュ領域側の分岐テーブル先頭アドレスは 0x300 番地にあるものとします。

なお，デフォルトで ROM 化処理も行います。

```
>cx -CF3746 -Xstartup=cstartN_b.obj boot.asm -Xlink_output=boot.lmf -
Xflash_ext_table=0x300 -Xlink_directive=directive_b.dir -oromp_b.lmf
```

ROM 化処理を行わない場合は，以下のように指定します。

```
>cx -CF3746 -Xstartup=cstartN_b.obj boot.asm -Xno_romize -oboot.lmf -
Xflash_ext_table=0x300 -Xlink_directive=directive_b.dir
```

- フラッシュ領域側のロード・モジュール・ファイルの生成

フラッシュ領域側のスタートアップ・ルーチン cstartN_f.obj を生成します。

```
>cx -CF3746 cstartN_f.asm
```

スタートアップ・ルーチン cstartN_f.obj，リンク・ディレクティブ・ファイル directive_f.dir をリンクして，フラッシュ領域側のロード・モジュール・ファイル flash.lmf を生成します。
その際，ブート領域側のロード・モジュール・ファイル boot.lmf のシンボル情報を参照して，リンク処理を行います。
また，0x300 番地に分岐テーブルを生成します。
なお，デフォルトで ROM 化処理も行います。

```
>cx -CF3746 -Xstartup=cstartN_f.obj flash.asm -Xflash=boot.lmf -
Xflash_ext_table=0x300 -Xlink_directive=directive_f.dir -oromp_f.lmf
```

ROM 化処理を行わない場合は，以下のように指定します。

```
>cx -CF3746 -Xstartup=cstartN_f.obj flash.asm -Xno_romize -oflash.lmf -
Xflash=boot.lmf -Xflash_ext_table=0x300 -Xlink_directive=directive_f.dir
```

注意 1. ブート領域側のロード・モジュール・ファイルは，ROM 化処理前のものである必要があります。

2. -Xflash_ext_table オプションのパラメータとして指定するアドレスは，ブート領域側とフラッシュ領域側で同じ値とする必要があります。

B. 1. 7　注意事項

ここでは，cx コマンドに関する注意事項について説明します。

（1）-Xsdata_info オプションの使用方法

ここでは，-Xsdata_info オプションの使用方法について説明します。

CubeSuite+ の場合，プロパティ パネルの［リンク・オプション］タブをオープンし，［その他］カテゴリの［GP 情報を表示する］プロパティで［はい (-Xsdata_info)］を選択します。

（a）機能

-Xsdata オプションのパラメータ *num* に対して，目安として用いることのできる情報を，標準出力に出力します。

CubeSuite+ の場合は，出力 パネルに出力します。

-Xsdata オプションは，*num* バイト以下のデータを .sdata セクション，または .sbss セクションに配置するオプションです。

cx は，sdata，sbss，data，bss 領域に配置するデータに対して，次のような規則に従ってコードを出力します。

まず，gp レジスタから 1 命令でアクセスできる領域である sdata 属性セクション，sbss 属性セクションへ配置しようとします（初期値ありデータを sdata 属性セクションへ，初期値なしデータを sbss 属性セクションへ配置します）。

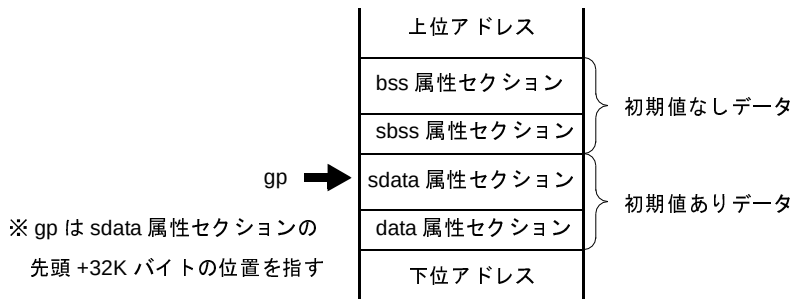
これらの領域は，gp と 16 ビットのディスプレイスメントを用いてアクセスするコードになるため，配置可能な範囲は gp から± 32K バイト内に限られます。

これらの領域に収まりきらなかった場合，gp レジスタから 2 命令でアクセスできる領域である data 属性セクション，bss 属性セクションへ配置しようとします（初期値ありデータを data 属性セクションへ，初期値なしデータを bss 属性セクションへ配置します）。

これらの領域は，まずアクセス領域のアドレス生成を行い，gp と 32 ビットのディスプレイスメントを用いてアクセスするコードを生成します。

そのため，4G バイトすべての空間へアクセスが可能になります。

図 B―8　gp オフセット参照セクションのメモリ配置イメージ



したがって，1 命令でアクセス可能な sdata 属性セクション，sbss 属性セクションに，より多くのデータを割り付けたほうが，実行効率やオブジェクト効率が良くなります。

データの割り付けは，C ソースの場合は #pragma section 指令，アセンブラ・ソースの場合は .section 疑似命令によって，ユーザが意図的に配置場所を指定する方法があります。

この方法のほかに，sdata 属性セクション，sbss 属性セクションへ割り付けるデータのサイズの閾値を設けて，そのサイズ以下のデータを sdata 属性セクション，sbss 属性セクションに配置する指定ができれば，ソース・プログラムに手を加えることなく，より多くのデータを配置することが可能になります。

この指定を行うのが -Xsdata オプションです。

ここで，*num* に指定する値は，データ・サイズになりますが，目安となる値がわかると便利です。

この情報を出力するのが，-Xsdata_info オプションです。

-Xsdata_info オプションを指定した場合は，-Xsdata オプションの *num* に対して，目安として用いることのできる情報を出力します。

（b）出力情報の説明

実行可能なオブジェクト・モジュール・ファイルの生成時（-Xrelinkable_object オプションを指定しない場合）に本オプションを指定した場合の出力情報の例と，再配置可能なオブジェクト・モジュール・ファイルの生成時（-Xrelinkable_object オプションを指定した場合）に本オプションを指定した場合の出力情報の例を，以下に示します。

例 1. 実行可能なロード・モジュール・ファイルに対する出力情報

```
        ******** LINK EDITOR GP INFORMATION ********
   (1)        (2)           (3)           (4)            (5)         (6)
GP SYMBOL    SECTION      SECTION      SECTION          GP
NAME         NAME         SIZE(REAL)   SIZE(ASSUMED)    NUMBER
_gp_DATA
             .sdata       0x000af10
                                       0x00002000       4           *OK*
                                       0x00003450       8           *OK*
                                       0x00004430       12          *OK*
                                       0x000050a8       16          *OK*
                                       0x00007b40       20          *OK*
                                       0x0000a010       24
                                       0x0000af10       32
             .sbss        0x00012050
                                       0x00000050       4           *OK*
                                       0x00002050       16          *OK*
                                       0x00007050       512         *OK*
                                       0x00010050       1024
```

2. 再配置可能なオブジェクト・モジュール・ファイルに対する出力情報

```
        ******** LINK EDITOR GP INFORMATION ********
   (1)        (2)           (3)           (4)            (5)         (6)
GP SYMBOL    SECTION      SECTION      SECTION          GP
NAME         NAME         SIZE(REAL)   SIZE(ASSUMED)    NUMBER
*(NOT AVAILABLE)
             .sdata       0x000af10
                                       0x00002000       4           *OK*
                                       0x00003450       8           *OK*
                                       0x00004430       12          *OK*
                                       0x000050a8       16          *OK*
                                       0x00007b40       20          *OK*
                                       0x0000a010       24
                                       0x0000af10       32
             .sbss        0x00012050
                                       0x00000050       4           *OK*
                                       0x00002050       16          *OK*
             *GpCommon*   0x00010000
                                       0x00005000       512         *OK*
                                       0x00010000       1024
```

項番	説明
(1)	グローバル・ポインタ・シンボルの名前 リンク時に用いられたグローバル・ポインタ・シンボルの名前です。 再配置可能なオブジェクト・モジュール・ファイルの場合，“*(NOT AVAILABLE)*”を表示します。
(2)	セクション名 データが割り付けられた sdata ／ sbss 属性セクションの名前です。 再配置可能なオブジェクト・モジュール・ファイルでは，未定義外部シンボルのセクションへの割り付けを確定することはできないため，仮想的なセクション“*GpCommon*”を内部的に生成して，このセクションに一時的に割り付けます。
(3)	セクションの実サイズ データの整列によって生じるホールの領域などが考慮されている“セクションの実サイズ”です。
(4)	想定されるセクションのサイズ (5) で示される値を *num* として -Xsdata オプションを指定してコンパイルした場合に想定されるセクションのサイズです。 このサイズの計算では，各データに対して実際の整列条件は考慮せず，一律に 4 バイト以上の整列条件を想定しているため，必ずしも実際に生成されるセクションの実サイズに一致するとはかぎりません。
(5)	想定された -Xsdata オプションの *num* の値 (4) で示される，“想定されるセクションのサイズ”の計算において想定されたコンパイル，およびアセンブル時における -Xsdata オプションの *num* の値です。

項番	説明
（6）	判定結果 （5）で示される値を *num* として -Xsdata オプションを指定してコンパイルした場合に，セクションのサイズが 15 ビット（0x0 ～ 0x7fff）の範囲に入るかどうかの判断結果**注**を表示します。 入る場合は“*OK*”が表示され，入らない場合は何も表示されません。

注 cx では，通常，データの割り付けられるセクションは data ／ sdata ／ sbss ／ bss 属性セクションの順に下位のアドレスから割り付けられ，グローバル・ポインタ（gp）は sdata 属性セクションの先頭アドレス＋ 32K バイトを指すよう，スタートアップ・ルーチンなどにおいて設定されることが想定されています。
そのため，この判定で OK が出た場合は，sdata ／ sbss 属性セクションは 16 ビットのディスプレースメントを用いて参照することのできるメモリの範囲に割り付けられていると考えることができます。

（c）注意事項

-Xsdata_info オプションで出力する情報は，あくまで目安であり，たとえば，以下のような場合は，判定結果が正しくない可能性があります。

- リンク・ディレクティブなどにおいて，ホールを生成するようなセクションの配置を指定した場合
- グローバル・ポインタ・シンボルに対して直接アドレスを指定した場合
- #pragma section 指令で，.sdata ／ .sbss セクションにデータを割り当てた場合

（2）ライブラリ・ファイル

ライブラリ・ファイルは，ライブラリアンによって複数のオブジェクト・モジュール・ファイルを結合することによって生成します。

cx は，すべてのオブジェクト・モジュール・ファイルをリンク後に，未解決な外部参照についてライブラリ・ファイルを検索し**注 1**，必要とされるオブジェクト・モジュール・ファイルのみをリンクします。

ライブラリ・ファイルは，リンク・ディレクティブのマッピング・ディレクティブにおいて指定することも可能です。

マッピング・ディレクティブにおいて指定した場合も，その指定された時点において未解決な外部参照について検索し，必要とされるオブジェクト・モジュール・ファイル**注 2** のみをリンクします。

注 1. ライブラリ・ファイルは，それが含んでいる各オブジェクト・モジュール・ファイルに属すシンボルのシンボル・テーブルを持っており，そのライブラリ・ファイルにより未解決な外部参照が解決されなくなるまで繰り返し検索します。

2. 参照されているシンボルが定義されているオブジェクト・モジュール・ファイルです。

(3) 予約シンボル

cx は，リンクの処理において，各出力セクションの先頭アドレス，各出力セクションの終端を越える最初のアドレス（4 バイトの整列条件で整列されたアドレス），および生成された実行可能なロード・モジュール・ファイルの終端を越える最初のアドレス（4 バイトの整列条件で整列されたアドレス）を値として持つ予約シンボルを生成します。

ユーザがこれらの予約シンボルと同名のシンボルを定義した場合，cx は定義されたシンボルを用い，独自に生成することはしません。

セクションの先頭アドレス値を値として持つ予約シンボルとしては，その出力セクションの名前の先頭に“__s”を付けることによって構成される名前のシンボルが用いられます。

ただし，そのセクション名が“.”で始まっている場合，その“.”を取った後ろの名前の先頭に“__s”を付けることによって構成される名前のシンボルが用いられます。

セクションの終端を越える最初のアドレス（4 バイトの整列条件で整列されたアドレス）を値として持つ予約シンボルとしては，その出力セクションの名前の先頭に“__e”を付けることによって構成される名前のシンボルが用いられます。

ただし，そのセクション名が“.”で始まっている場合，その“.”を取った後ろの名前の先頭に“__e”を付けることによって構成される名前のシンボルが用いられます。

生成された実行可能なロード・モジュール・ファイルの終端を越える最初のアドレス（4 バイトの整列条件で整列されたアドレス）を値として持つ予約シンボルとしては，__end が用いられます。

cx の用いるデフォルトのリンク・ディレクティブでは，出力セクションとして以下の予約セクションが用いられています。

.text，.pro_epi_runtime，.data，.sdata，.sbss，.bss，.sconst，.const，.sedata，.sebss，.sidata，.sibss，.tidata，.tidata.byte，.tidata.word，.tibss，.tibss.byte，.tibss.word

このため，cx は，通常，以下の予約シンボルを生成することになります。

__end，__ebss，__econst，__edata，__epro_epi_runtime，__esbss，__esconst，__esdata，__esebss，__esedata，__esibss，__esidata，__etext，__etibss，__etibss.byte，__etibss.word，__etidata，__etidata.byte，__etidata.word，__sbss，__sconst，__sdata，__spro_epi_runtime，__ssbss，__ssconst，__ssdata，__ssebss，__ssedata，__ssibss，__ssidata，__stext，__stibss，__stibss.byte，__stibss.word，__stidata，__stidata.byte，__stidata.word

注意　生成するシンボルは，上記のうち，リンク処理後の実行形式ファイルにセクションが存在するもののみとなります。

cx では，リンク・ディレクティブ・ファイルに対してマッピング・ディレクティブを記述しても，実際に割り付けられるセクションが存在しなければ，セクションは存在しないものとして扱います。

（4）期待したセクションに割り付けられない場合

リンク・ディレクティブ・ファイル内で，セクションに割り付けるオブジェクト・モジュール・ファイルやライブラリ・ファイルを指定しても，ファイル名の記述の仕方によってはそれらが期待したセクションに割り付けられないことがあります。

その場合は，マップ・ファイルに表示されているファイル名とパス名も含めて，まったく同じ名前でリンク・ディレクティブ・ファイルに指定して，再リンクしてください。

（5）main 関数

main 関数を作らずにリンクした場合，_main シンボルが undefined symbol エラーとして出力されることがあります。

特に，スタートアップ・ルーチンを独自に指定せず，デフォルトのスタートアップ・ルーチン（cstart.obj, cstartN.obj）をリンクした場合，また cstart.asm，cstartN.asm をそのままアセンブル，リンクして使用した場合に発生します。

これは，cstart.asm，cstartN.asm の最後の方に書かれている“jarl _main, lp”というコードが原因です。

main 関数が必要ない場合は，ここを独自に書き換え，再アセンブルして作成したオブジェクト・モジュール・ファイルをスタートアップ・ルーチンとしてください。

また，リアルタイム OS を使用したアプリケーションの場合，通常 main 関数はありません。

リアルタイム OS のサンプルとして提供されているスタートアップ・ルーチンを使用してください。

（6）プロローグ／エピローグ・ランタイム・ライブラリ

プロローグ／エピローグ・ランタイム・ライブラリは，専用の .pro_epi_runtime セクションに配置する必要があります。

配置していない場合は，以下のメッセージを出力して，リンクを中止します。

```
F0560657:Section "section" must be specified in link directive.
```

リンク・ディレクティブ・ファイルを指定している場合には，.text セクションの手前にマッピング・ディレクティブを記述してください。

```
.pro_epi_runtime = $PROGBITS ?AX .pro_epi_runtime;
.text = $PROGBITS ?AX;
```

.pro_epi_runtime セクションを .text セクションの後ろに配置すると，ROM 化の際，パッキングされたセクションのデフォルト動作での配置位置と重なるため，.text セクションの手前に配置することを推奨します。

リンク・ディレクティブ・ファイルを指定しない場合には，.text セクションの手前にリンクします。

その他の注意事項を以下に示します。

- プロローグ／エピローグ・ランタイム・ライブラリは，標準ライブラリ libc.lib に含まれています。
- .pro_epi_runtime セクションは通常のセクションと異なり，入力セクション名が固定されており，専用のセクションだけを配置します。

- V850Ex ／ V850E2 コアのデバイス指定時，プロローグ／エピローグ・ランタイム・ライブラリは callt 命令を使用しています。
  スタートアップ・ルーチンで，CTBP の設定を行ってください。

(7) プログラマブル周辺 I/O レジスタ

プログラマブル周辺 I/O レジスタ機能を使用するアプリケーション・プログラムの場合，アセンブル時に予約セクションの .bpc セクションを出力します。

cx は，リンク時の入力オブジェクト・モジュール・ファイルに .bpc セクションが存在する場合，BPC 値として指定されている値のチェックを行います。

入力オブジェクト・モジュール・ファイル間で値が統一されていない場合は，cx は以下のようなエラー・メッセージを出力して，リンク処理を中断します。

```
F0560114:Input files have different BPC value.
0x00001234      file1.obj
0x00001234      file2.obj
0x00001235      file3.obj
*(none)*        file4.obj
```

上記の場合，file3.obj で設定されている値が異なるため，エラーとなっています。

なお，プログラマブル周辺 I/O レジスタを参照していないオブジェクトは，チェックの対象とはなりません。

上記の file4.obj のように，“*(none)*”と表示します。

BPC 値のチェックでエラーがなかった場合，セクション・タイプ SHT_PROGBITS，セクション属性 none，セクション・サイズ 0x4 の .bpc セクションを生成します。

.bpc セクションには，BPC 値を既定ビット数分シフトして得られる，プログラマブル I/O レジスタ領域の先頭アドレスを格納します。

例 V850E/IA1 使用時に BPC 値を“0x1234”と指定した場合，プログラマブル周辺 I/O レジスタ領域の先頭アドレスは，この値を 14 ビット左シフトした“0x48d0000”となります。この際，.bpc セクション内の情報は次のようになります。

```
.bpc
  Address    00  01  02  03  04  05  06  07  -  08  09  0A  0B  0C  0D  0E  0F
0x00000000 : 00  00  8d  04                  -                              ...
```

- 以上の処理は，再配置可能なオブジェクト・モジュール・ファイルの生成時，実行可能なオブジェクト・モジュール・ファイルの生成時を問わずに行われます。
- .bpc セクションは，情報用の特殊な予約セクションであり，メモリにロードされることはありません。
  したがって，通常のセクションのように，リンク・ディレクティブに記述する必要はありません。

（8）デバッグ情報

typedef で別名を指定した型のデバッグ情報において，元の型がソース・ファイル中で記述した型と一致しない（サイズと符号の有無が同じ別の型になる）場合があります。

- 元の型が unsigned，unsigned int の場合，unsigned long になります。

例

```
typedef unsigned int UI;     /* デバッグ情報では，UI の元の型は unsigned long となる */
```

- 元の型が signed，int，signed int，signed long の場合，long になります。

例

```
typedef signed int SI;       /* デバッグ情報では，SI の元の型は long となる */
```

- 元の型が signed short の場合，short になります。

例

```
typedef signed short SS;     /* デバッグ情報では，SS の元の型は short となる */
```

- 元の型が signed char の場合，char になります。

例

```
typedef signed char SC;      /* デバッグ情報では，SC の元の型は char となる */
```

- 元の型が char，かつ -Xchar=unsigned オプションを指定した場合，unsigned char になります。

例

```
typedef char SC;             /* デバッグ情報では，SC の元の型は unsigned char となる */
```

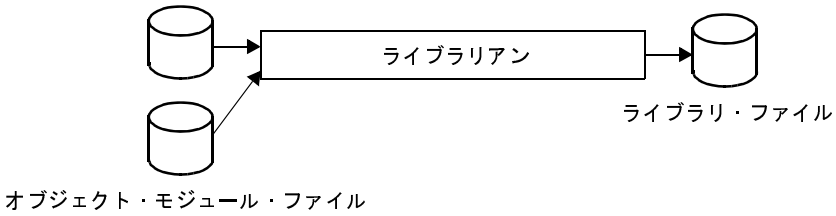
B. 2　ライブラリアン

ライブラリアンは，指定したリロケータブルなオブジェクト・モジュール・ファイルを結合し，１つのライブラリ・ファイルを生成します。

つまり，ユーザが複数のオブジェクト・モジュール・ファイルをまとめ，“ライブラリ”として作成する場合に使用します。

CX に入っている“lb”がライブラリアンです。

図 B―9　ライブラリアンにおける動作の流れ



以下に，ライブラリアンの機能詳細を示します。

- オブジェクト・モジュールのライブラリ化

cx は，１つの出力モジュールを１つのファイルに作成します。
したがって，モジュールの数が多い場合，ファイルの数も増加します。
このため，複数のモジュールを１つのファイルにまとめる機能が用意されています。
これをモジュールのライブラリ化と呼び，ライブラリ化されたファイルをライブラリ・ファイルと呼びます。

また，ライブラリアンによって生成するライブラリ・ファイルは，cx の入力ファイルとして指定することができます。
ライブラリ・ファイルを指定した場合，cx は指定したライブラリ・ファイル内から必要とされるオブジェクト・モジュール・ファイルを検索し，見つかったオブジェクト・モジュール・ファイルのみをリンクします。
したがって，モジュラ・プログラミングを行った場合，共通的なモジュールをライブラリ・ファイルとして作成しておけば，ファイル管理の面でも操作性の面においても効率が良くなります。

- ライブラリ・ファイルに対する編集機能

ライブラリアンには，ライブラリ・ファイルに対して以下の編集機能があります。
- ライブラリ・ファイルへのオブジェクト・モジュール・ファイルの追加
- ライブラリ・ファイル内のオブジェクト・モジュール・ファイルの削除
- ライブラリ・ファイル内のオブジェクト・モジュール・ファイルの移動
- ライブラリ・ファイル内のオブジェクト・モジュール・ファイルの入れ替え
- ライブラリ・ファイル内のオブジェクト・モジュール・ファイルの抽出

B. 2. 1　入出力ファイル

ライブラリアンの入出力ファイルを以下に示します。

表 B—11　ライブラリアンの入出力ファイル

ファイル種別	拡張子	入出力	説明
オブジェクト・モジュール・ファイル	.obj	入力	機械語情報と機械語の配置アドレスに関する再配置情報，およびシンボル情報を含んだバイナリ・ファイル
ライブラリ・ファイル	.lib	出力	複数のオブジェクト・モジュール・ファイルが登録されたファイル

備考　上記の拡張子はデフォルトであり，任意の拡張子に変更可能です。

B. 2. 2　操作方法

ここでは，ライブラリアンの操作方法について説明します。

(1) コマンド・ラインでの操作方法

コマンドは，コマンド・ラインで以下のように入力します。

```
>lb[ △エラー出力制御オプション] △キー[ オプション][ △メンバ名注] △ライブラリ・ファイル名[ △メンバ名，またはファイル名]...
```

[ ]　：[ ] 内は省略可能です。

...　：直前の [ ] 内のパターンの繰り返しが可能です。

△　：1 個以上の空白を示します。

注　オブジェクト・モジュール・ファイルは，ライブラリ・ファイル内に連結されるとメンバと呼ばれます。
メンバは，連結される前のファイルのときと同じ名前を持ちます。

備考　キーはライブラリアン起動時に 1 つだけ指定しなければなりませんが，オプションは省略可能です。

(2) CubeSuite+ でのオプション設定

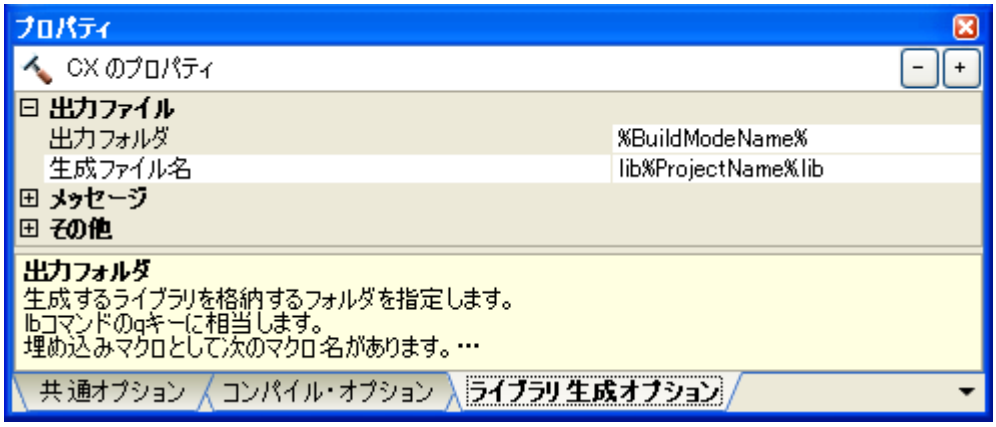
CubeSuite+ からライブラリ生成オプションを設定する方法について説明します。

CubeSuite+ のプロジェクト・ツリー パネルにおいて，ビルド・ツール・ノードを選択したのち，［表示］メニュー→［プロパティ］を選択すると，プロパティ パネルがオープンします。

次に，［ライブラリ生成オプション］タブを選択します。

タブ上で各プロパティを設定することにより，対応するライブラリ生成オプションを設定することができます。

図 B—10　プロパティ パネル：［ライブラリ生成オプション］タブ



コマンド・ライン上からライブラリアンを起動する場合は，オブジェクト・モジュールファイル群をまとめて，ライブラリ・ファイルを作成します。

また，ライブラリ・ファイル内のオブジェクトの操作など，細かい作業を行うことができます。

一方，CubeSuite+ を使用してライブラリ・ファイルを生成する場合は，ソース・ファイルをコンパイル，アセンブルし，生成されたオブジェクト・モジュール・ファイルをライブラリ・ファイルにまとめるという作業になります。

CubeSuite+ 上からは，完成されたライブラリ・ファイルに対して操作することはできません。

そのため，コマンド・ラインと CubeSuite+ を使い分けていく必要があります。

B. 2. 3　キー／オプション

ライブラリアンのキー／オプションの分類と説明を以下に示します。

表 B—12　ライブラリ生成キー

分類	キー	説明
バージョン表示指定	V	ライブラリアンのバージョン情報を出力します。
メンバ削除指定	d	指定したライブラリ・ファイルからメンバを削除します。
メンバ移動制御	m	指定したライブラリ・ファイル内の指定したメンバをライブラリ・ファイルの最後に移動します。
	ma	指定したライブラリ・ファイル内の指定したメンバを指定したメンバの直後に移動します。
	mb	指定したライブラリ・ファイル内の指定したメンバを指定したメンバの直前に移動します。
メンバ追加指定	q	指定したライブラリ・ファイルの最後に指定したオブジェクト・モジュール・ファイルを追加します。
メンバ入れ替え制御	r	指定したオブジェクト・モジュール・ファイルを指定したライブラリ・ファイル内の同名のメンバと入れ替えます。
	ra	指定したオブジェクト・モジュール・ファイルを指定したライブラリ・ファイル内の同名のメンバと入れ替え，指定したメンバの直後に移動します。
	ru	指定したオブジェクト・モジュール・ファイルが指定したライブラリ・ファイル内の同名のメンバより最近に更新されている場合に，メンバの入れ替えを行います。
メンバ出力指定	t	指定したライブラリ・ファイル内に存在しているメンバの名前を標準出力に出力します。
オブジェクト生成制御	x	指定したライブラリ・ファイル内に存在しているメンバを取り出し，同名のオブジェクト・モジュール・ファイルを生成します。

表 B—13　ライブラリ生成オプション

分類	オプション	説明
メッセージ出力抑止指定	c	メッセージを出力しません。
実行状況出力指定	v	ライブラリアンの実行状況を出力します。
コマンド・ファイル指定	@	コマンド・ファイルを指定します。
エラー出力制御	+err_file	エラー・メッセージを指定したファイルに追加保存します。
	-err_file	エラー・メッセージを指定したファイルに上書き保存します。

バージョン表示指定

バージョン表示指定キーには，次のものがあります。

- V

V

ライブラリアンのバージョン情報を出力します。

[指定形式]

```
V
```

- 省略時解釈

  ライブラリアンのバージョン情報を出力しません。

[詳細説明]

- ライブラリアンのバージョン情報を標準エラー出力に出力して，終了します。

[使用例]

- ライブラリアンのバージョン情報を標準エラー出力に出力します。

```
>lb V
```

メンバ削除指定

メンバ削除指定キーには，次のものがあります。

- d

d

指定したライブラリ・ファイルからメンバを削除します。

[指定形式]

```
d
```

- 省略時解釈

  なし

[詳細説明]

- 指定したライブラリ・ファイルからメンバを削除します。

[使用例]

- ライブラリ・ファイル libmain.lib からメンバ sub.obj を削除します。

```
>lb d libmain.lib sub.obj
```

メンバ移動制御

メンバ移動制御キーには，次のものがあります。

- m
- ma
- mb

m

指定したライブラリ・ファイル内の指定したメンバをライブラリ・ファイルの最後に移動します。

[指定形式]

```
m
```

- 省略時解釈

  メンバの移動を行いません。

[詳細説明]

- 指定したライブラリ・ファイル内の指定したメンバをライブラリ・ファイルの最後に移動します。

[使用例]

- ライブラリ・ファイル libmain.lib 内のメンバ sub.obj をライブラリ・ファイルの最後に移動します。

```
>lb m libmain.lib sub.obj
```

ma

指定したライブラリ・ファイル内の指定したメンバを指定したメンバの直後に移動します。

[指定形式]

```
ma member
```

- 省略時解釈

  メンバの移動を行いません。

[詳細説明]

- 指定したライブラリ・ファイル内の指定したメンバをメンバ *member* の直後に移動します。
- *member* を省略した場合は，処理を中止します。

[使用例]

- ライブラリ・ファイル libmain.lib 内のメンバ sub.obj をメンバ main.obj の直後に移動します。

```
>lb ma main.obj libmain.lib sub.obj
```

mb

指定したライブラリ・ファイル内の指定したメンバを指定したメンバの直前に移動します。

[指定形式]

```
mb member
```

- 省略時解釈

  メンバの移動を行いません。

[詳細説明]

- 指定したライブラリ・ファイル内の指定したメンバをメンバ *member* の直前に移動します。
- *member* を省略した場合は，処理を中止します。

[使用例]

- ライブラリ・ファイル libmain.lib 内のメンバ sub.obj をメンバ main.obj の直前に移動します。

```
>lb mb main.obj libmain.lib sub.obj
```

メンバ追加指定

メンバ追加指定キーには，次のものがあります。

- q

q

指定したライブラリ・ファイルの最後に指定したオブジェクト・モジュール・ファイルを追加します。

[指定形式]

```
q
```

- 省略時解釈

  なし

[詳細説明]

- 指定したライブラリ・ファイルの最後に指定したオブジェクト・モジュール・ファイルを追加します。
- 指定したライブラリ・ファイルが存在しない場合は，指定したオブジェクト・モジュール・ファイルを含んだライブラリ・ファイルを新規に生成します。
- 指定したオブジェクト・モジュール・ファイルと同名のメンバが存在しているかどうかのチェックは行いません。
  同名のメンバが存在する場合は，ライブラリ・ファイルに複数の同名メンバが含まれ，リンク時に最も古いメンバを選択します。
  同名のメンバが混在するのを避けるため，ライブラリ・ファイルの新規作成を行う場合は，古いライブラリ・ファイルを必ず削除してください。
  または，r キーを使用することを推奨します。
- 同名のメンバを入れ替える場合は，r キーを使用してください。

[使用例]

- ライブラリ・ファイル libmain.lib の最後にオブジェクト・モジュール・ファイル sub.obj を追加します。

```
>lb q libmain.lib sub.obj
```

メンバ入れ替え制御

メンバ入れ替え制御キーには，次のものがあります。

- r
- ra
- ru

r

指定したオブジェクト・モジュール・ファイルを指定したライブラリ・ファイル内の同名のメンバと入れ替えます。

[指定形式]

```
r
```

- 省略時解釈

  メンバの入れ替えを行いません。

[詳細説明]

- 指定したオブジェクト・モジュール・ファイルを指定したライブラリ・ファイル内の同名のメンバと入れ替えます。
- 指定したライブラリ・ファイル内に同名のメンバが存在しない場合は，指定したオブジェクト・モジュール・ファイルをライブラリ・ファイルの最後に追加します。
- 指定したライブラリ・ファイルが存在しない場合は，指定したオブジェクト・モジュール・ファイルを含んだライブラリ・ファイルを新規に生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ライブラリ生成オプション］タブの［出力ファイル］カテゴリの［出力フォルダ］，［生成ファイル名］

[使用例]

- sub.obj をライブラリ・ファイル libmain.lib 内の同名のメンバと入れ替えます。

```
>lb r libmain.lib sub.obj
```

ra

指定したオブジェクト・モジュール・ファイルを指定したライブラリ・ファイル内の同名のメンバと入れ替え，指定したメンバの直後に移動します。

[指定形式]

```
ra member
```

- 省略時解釈

  メンバの入れ替えを行いません。

[詳細説明]

- 指定したオブジェクト・モジュール・ファイルを指定したライブラリ・ファイル内の同名のメンバと入れ替え，メンバ *member* の直後に移動します。
- 指定したライブラリ・ファイル内に同名のメンバが存在しない場合は，指定したオブジェクト・モジュール・ファイルをライブラリ・ファイルの最後に追加します。
- *member* を省略した場合は，処理を中止します。

[使用例]

- sub.obj をライブラリ・ファイル libmain.lib 内の同名のメンバを入れ替え，メンバ main.obj の直後に移動します。

```
>lb ra main.obj libmain.lib sub.obj
```

ru

　指定したオブジェクト・モジュール・ファイルが指定したライブラリ・ファイル内の同名のメンバより最近に更新されている場合に，メンバの入れ替えを行います。

[指定形式]

```
ru
```

- 省略時解釈
  メンバの入れ替えを行いません。

[詳細説明]

- 指定したオブジェクト・モジュール・ファイルが指定したライブラリ・ファイル内の同名のメンバより最近に更新されている場合に，メンバの入れ替えを行います。
- 指定したライブラリ・ファイル内に同名のメンバが存在しない場合は，指定したオブジェクト・モジュール・ファイルをライブラリ・ファイルの最後に追加します。
- 指定したライブラリ・ファイルが存在しない場合は，指定したオブジェクト・モジュール・ファイルを含んだライブラリ・ファイルを新規に生成します。

[使用例]

- sub.obj がライブラリ・ファイル libmain.lib 内の sub.obj より最近に更新されている場合に，それらの入れ替えを行います。

```
>lb ru libarc.lib sub.obj
```

メンバ出力指定

メンバ出力指定キーには，次のものがあります。

- t

t

指定したライブラリ・ファイル内に存在しているメンバの名前を標準出力に出力します。

[指定形式]

```
t
```

- 省略時解釈

  メンバの出力を行いません。

[詳細説明]

- メンバ名を指定した場合は，指定したライブラリ・ファイル内に存在している指定したメンバの名前を標準出力に出力します。
- メンバ名を指定しない場合は，指定したライブラリ・ファイル内に存在しているすべてのメンバの名前を標準出力に出力します。

[使用例]

- ライブラリ・ファイル libmain.lib 内に存在しているすべてメンバの名前を標準出力に出力します。

```
>lb t libmain.lib
```

オブジェクト生成制御

オブジェクト生成制御キーには，次のものがあります。

- x

x

指定したライブラリ・ファイル内に存在しているメンバを取り出し，同名のオブジェクト・モジュール・ファイルを生成します。

[指定形式]

```
x
```

- 省略時解釈

  メンバの取り出しを行いません。

[詳細説明]

- メンバ名を指定した場合は，指定したメンバが指定したライブラリ・ファイル内に存在していればそのメンバを取り出し，同名のオブジェクト・モジュール・ファイルを生成します。
- メンバ名を指定しない場合は，指定したライブラリ・ファイル内に存在しているすべてのメンバを取り出し，同名のオブジェクト・モジュール・ファイルを生成します。

  ライブラリ・ファイルの内容は変更しません。

[使用例]

- ライブラリ・ファイル libmain.lib 内に存在しているメンバ sub.obj を取り出し，sub.obj を生成します。

```
>lb x libmain.lib sub.obj
```

メッセージ出力抑止指定

メッセージ出力抑止指定オプションには，次のものがあります。

- c

c

メッセージを出力しません。

[指定形式]

```
c
```

- 省略時解釈

  メッセージを出力します。

[詳細説明]

- メッセージを出力しません。

[使用例]

- メッセージを出力しません。

```
>lb tc libmain.lib
```

実行状況出力指定

実行状況出力指定オプションには，次のものがあります。

- v

v

ライブラリアンの実行状況を出力します。

[指定形式]

```
v
```

- 省略時解釈

  ライブラリアンの実行状況を出力しません。

[詳細説明]

- ライブラリアンの実行状況を“[a|d|q|m|r|x] - *file*”の形式で出力します。

a - *file*	追加
d - *file*	削除
q - *file*	新規作成，または追加
m - *file*	移動
r - *file*	置換
x - *file*	抽出

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  - ［ライブラリ生成オプション］タブの［メッセージ］カテゴリの［実行状況を表示する］

[使用例]

- ライブラリ・ファイル libmain.lib からメンバ sub.obj を削除します。

  その際，ライブラリアンの実行状況を出力します。

```
>lb dv libmain.lib sub.obj
```

コマンド・ファイル指定

コマンド・ファイル指定オプションには，次のものがあります。

- @

@

コマンド・ファイルを指定します。

[指定形式]

```
@file
```

- 省略時解釈

  コマンド・ファイルの指定がないものとみなします。

[詳細説明]

- *file* をコマンド・ファイルとして扱います。
- コマンド・ファイルについての詳細は，「(2) コマンド・ファイルによる操作方法」を参照してください。

[使用例]

- command をコマンド・ファイルとして扱います。

```
>lb @command
```

エラー出力制御

エラー出力制御オプションには，次のものがあります。

- +err_file
- -err_file

+err_file

エラー・メッセージを指定したファイルに追加保存します。

[指定形式]

```
+err_file=file
```

- 省略時解釈

  エラー・メッセージ・ファイルを生成しません。

[詳細説明]

- エラー・メッセージをファイル *file* に追加保存します。
- 本オプションは，コマンド・ラインの先頭（キーの前）に指定する必要があります。

[使用例]

- ライブラリ・ファイル libmain.lib からメンバ sub.obj を削除します。

  その際，エラー・メッセージをファイル err に追加保存します。

```
>lb +err_file=err d libmain.lib sub.obj
```

-err_file

エラー・メッセージを指定したファイルに上書き保存します。

[指定形式]

```
-err_file=file
```

- 省略時解釈

  エラー・メッセージ・ファイルを生成しません。

[詳細説明]

- エラー・メッセージをファイル *file* に上書き保存します。
- 本オプションは，コマンド・ラインの先頭（キーの前）に指定する必要があります。

[使用例]

- ライブラリ・ファイル libmain.lib からメンバ sub.obj を削除します。

  その際，エラー・メッセージをファイル err に上書き保存します。

```
>lb -err_file=err d libmain.lib sub.obj
```

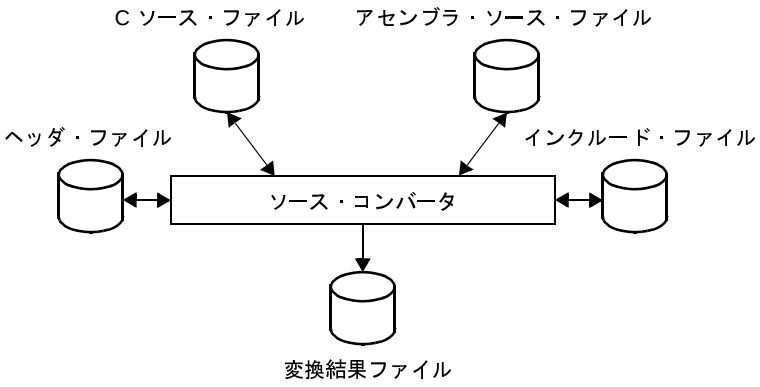
B. 3　ソース・コンバータ

ソース・コンバータは，CA850 向けに作成した各ソース・ファイル（C ソース・ファイル，アセンブラ・ソース・ファイルなど）に対して，CX で利用できるように変換処理を行います。

変換処理後の各出力ファイル内には，CA850 用記述をコメントとして残します。

なお，オプション指定により，変換結果をテキスト・ファイルに出力することも可能です。

図 B—11　ソース・コンバータにおける処理の流れ



B. 3. 1　入出力ファイル

ソース・コンバータの入出力ファイルを以下に示します。

表 B—14　ソース・コンバータの入出力ファイル

ファイル種別	拡張子	入出力	説明
C ソース・ファイル	.c	入出力	CA850 用拡張記述を含む C ソース・ファイルを入力し，CX 用記述を含むファイルを出力します。
ヘッダ・ファイル	.h	入出力	CA850 用ヘッダ・ファイルを入力し，CX 用記述を含むファイルを出力します。
アセンブラ・ソース・ファイル	.s	入出力	CA850 用アセンブラ・ソース・ファイルを入力し，CX 用記述を含むファイルを出力します。
インクルード・ファイル	.inc	入出力	CA850 用インクルード・ファイルを入力し，CX 用記述を含むファイルを出力します。
変換結果ファイル	任意	出力	変換結果を出力したテキスト・ファイルです。 -r オプション指定時に出力します。

B. 3. 2　操作方法

ここでは，ソース・コンバータの操作方法について説明します。

なお，CubeSuite+ では，ビルド・ツールが CA850 であるプロジェクトを流用して，ビルド・ツールが CX であるプロジェクトを新規作成する際に，変換処理を行います。

詳細については，「CubeSuite+ 統合開発環境 ユーザーズマニュアル 起動編」を参照してください。

注意　CubeSuite+ では，オプションを設定することはできません。
オプションを設定する場合は，コマンド・ラインで操作してください。

(1) コマンド・ラインでの操作方法

コマンドは，コマンド・ラインで以下のように入力します。

```
>cnv850[ △ファイル名 ][ △オプション ]...
```

[ ]　：[ ] 内は省略可能です。
...　：直前の [ ] 内のパターンの繰り返しが可能です。
△　：1 個以上の空白を示します。

- ファイル名（オプションのパラメータも含む）を指定する場合は，パス付き（絶対パス，または相対パス）での指定が可能です。
  パスなし，および相対パスで指定する場合は，カレント・フォルダを基準とします。
- ファイル名（オプションのパラメータも含む）に空白を含める場合（パス名など）は，そのパラメータ全体をダブルクォーテーション（”“）で囲んでください。
- 指定可能なファイル名（オプションのパラメータも含む）の長さは，パスを含めて 259 文字までです。
- ファイル名のアルファベットは，大文字／小文字を区別しません。
- 複数のファイル名を指定した場合は，エラーとなります。
  入力として，複数のファイルを指定する場合は，-I オプションを使用してください。
- 指定したファイル名（オプションのパラメータも含む）が存在しない場合は，エラーとなります。
  ただし，-o オプションのパラメータとして指定したファイルが存在しない場合は，エラーとはなりません。
  また，-I オプションのパラメータとして指定したファイルが存在しない場合は，警告を出力して，そのファイルの処理をスキップします。
- オプションの大文字／小文字は区別します。
- 「表 B—15　ソース・コンバート・オプション」に記載されていないオプションを指定した場合は，警告を出力せずに，指定したオプションを無視します。

(2) 変換結果の表示内容

(a) 概要

変換結果の概要は，標準エラー出力に表示します。

以下に，出力フォーマットを示します。

```
ファイル名
    結果
```

ファイル名 ：コマンド・ラインの指定イメージです。
結果 ：以下のいずれかを表示します。

- 変換の必要な箇所を検出した場合

```
Converted(総数)(個数△deleted,△個数△inserted,△個数△changed,△個数△information)
```

総数 ：変換箇所の総数です。
個数 ：情報種別（削除，追加，変更，情報）ごとの変換箇所の個数です。

- 変換の必要な箇所がない場合

```
None
```

- エラーが発生した場合

```
Error
```

備考 -I オプションを使用して複数の入力ファイルを指定した場合は，入力ファイルごとに表示します。

（b）詳細

変換結果の詳細は，変換が必要な箇所ごとに，以下のフォーマットで出力します。
デフォルトでは標準出力に出力しますが，-r オプションを指定した場合は，ファイルに出力します。

```
ファイル名(行番号):△メッセージ番号:[情報種別]△内容
```

ファイル名 ：変換対象ファイル名のコマンド・ラインの指定イメージです。
行番号 ：変換対象行の番号です。
メッセージ番号 ：出力メッセージの番号です。
情報種別 ：以下のいずれかを表示します。

情報種別	意味
削除	対象行を削除したことを示します。
追加	対象行を追加したことを示します。
変更	対象行を変更したことを示します。
情報	対象行はユーザが手動で修正してください。

内容 ：出力メッセージの内容です。

B. 3. 3　オプション

ソース・コンバータのオプションの分類と説明を以下に示します。

表 B―15　ソース・コンバート・オプション

分類	オプション	説明
バージョン／ヘルプ表示指定	-V	ソース・コンバータのバージョン情報を表示します。
	-h	ソース・コンバータのオプションの説明を表示します。
文字コード指定	-c	日本語の文字コードを指定します。
リスト・ファイル指定	-l	入出力ファイル名を記述したリスト・ファイルを指定します。
出力ファイル指定	-o	出力ファイル名を指定します。
変換結果ファイル指定	-r	変換結果ファイル名を指定します。
ファイル種別指定	-t	入力ファイルの種別を指定します。

バージョン／ヘルプ表示指定

バージョン／ヘルプ表示指定オプションには，次のものがあります。

- -V
- -h

-V

ソース・コンバータのバージョン情報を表示します。

[指定形式]

```
-V
```

- 省略時解釈

  ソース・コンバータのバージョン情報を表示せずに，変換を行います。

[詳細説明]

- ソース・コンバータのバージョン情報を標準エラー出力に出力します。

  変換は行いません。
- ほかのオプションを同時に指定した場合は，警告を出力せずに，ほかのオプションを無視します。

[使用例]

- ソース・コンバータのバージョン情報を標準エラー出力に出力します。

```
>cnv850 -V
```

-h

ソース・コンバータのオプションの説明を表示します。

[指定形式]

```
-h
```

- 省略時解釈

  ソース・コンバータのオプションの説明を表示しません。

[詳細説明]

- ソース・コンバータのオプションの説明を標準エラー出力に出力します。

  変換は行いません。
- ほかのオプション（-V を除く）と同時に指定した場合は，警告を出力せずに，ほかのオプションを無視します。
- -V オプションと同時に指定した場合は，-V オプションが有効となります。

[使用例]

- ソース・コンバータのオプションの説明を標準エラー出力に出力します。

```
>cnv850 -h
```

文字コード指定

文字コード指定オプションには，次のものがあります。

- -c

-c

日本語の文字コードを指定します。

[指定形式]

```
-c=code
```

- 省略時解釈

  日本語の文字コードを SJIS として扱います。

[詳細説明]

- 入力ファイル中の日本語のコメント，文字列に対して，使用する文字コードを指定します。
- *code* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，警告を出力せずに，無視します。

none	日本語の文字コードを処理しません（ASCII として処理します）
euc	EUC（日本語）
sjis	SJIS

- *code* を省略した場合は，エラーとなります。

[使用例]

- 入力ファイル中の日本語のコメント，文字列に対して，使用する文字コードに EUC を指定します。

```
>cnv850 main.c -c=euc
```

リスト・ファイル指定

リスト・ファイル指定オプションには，次のものがあります。

- -l

-l

入出力ファイル名を記述したリスト・ファイルを指定します。

[指定形式]

```
-l=file
```

- 省略時解釈

  コマンド・ラインでファイル名を指定している場合は，そのファイルを入力ファイル名とみなします。
  ただし，その場合は，-o オプションを指定する必要があります。
  コマンド・ラインでファイル名を指定しない場合は，何も行いません。

[詳細説明]

- 入力ファイル名と出力ファイル名を記述したリスト・ファイルを *file* に指定します。
  リスト・ファイルのフォーマットを以下に示します。

```
[-t △][-c △]入力ファイル名△出力ファイル名
```

  - 指定可能な入力ファイル名，および出力ファイル名は，コマンド・ラインに指定可能なものと同一とします。
    また，指定規則も，コマンド・ラインでの指定規則に準じます。
  - 入力ファイルが存在しない場合は，警告を出力して，そのファイルの処理をスキップします。
  - リスト・ファイル中で -c オプションを指定した場合，コマンド・ラインで指定した -c オプションとパラメータが異なっていれば，警告を出力して，リスト・ファイル中の指定を有効とします。
  - リスト・ファイルの日本語の文字コードは，UTF-8（BOM あり）のみを受容します。
  - 改行コードは，CR ＋ LF のみを受容します。

- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。
- 本オプションを指定した場合は，コマンド・ラインで指定した入力ファイル名は，警告を出力して，無視します。

［使用例］

- リスト・ファイル io.lst を指定します。

```
>cnv850 -l=io.lst
```

出力ファイル指定

出力ファイル指定オプションには，次のものがあります。

- -o

-o

出力ファイル名を指定します。

[指定形式]

```
-o=file
```

- 省略時解釈

  コマンド・ラインでファイル名を指定した場合は，省略することはできません。

  省略した場合は，エラーとなります（-h，または -V オプション指定時を除く）。

[詳細説明]

- 変換後の出力ファイル名を *file* に指定します。
- *file* がすでに存在する場合は，警告を出力せずに，そのファイルを上書きします。
- *file* が存在しない場合は，エラーとなります。
- *file* を省略した場合は，エラーとなります。

[使用例]

- 変換後のファイルをファイル名 main_cnv.c で出力します。

```
>cnv850 main.c -o=main_cnv.c
```

変換結果ファイル指定

変換結果ファイル指定オプションには，次のものがあります。

- -r

-r

変換結果ファイル名を指定します。

[指定形式]

```
-r=file
```

- 省略時解釈

  変換結果を標準出力に出力します。

[詳細説明]

- 変換結果をファイル *file* に出力します。
- *file* を省略した場合は，エラーとなります。

[使用例]

- 変換結果をファイル result に出力します。

```
>cnv850 main.c -r=result
```

ファイル種別指定

ファイル種別指定オプションには，次のものがあります。

- -t

-t

入力ファイルの種別を指定します。

[指定形式]

```
-t=type
```

- 省略時解釈

  入力ファイルの拡張子により，ファイル種別を判断します。

[詳細説明]

- 入力ファイルを *type* に指定した種別であるものとして，処理を行います。
- *type* に指定可能なものを以下に示します。

  これ以外のものを指定した場合は，警告を出力せずに，無視して，入力ファイルは変換せずにそのまま出力します。

c	C ソース・ファイル，または C ヘッダ・ファイル
asm	アセンブラ・ソース・ファイル，またはアセンブラ・インクルード・ファイル

- *type* を省略した場合は，エラーとなります。

[使用例]

- 入力ファイルを C ソース・ファイルであるものとして，処理を行います。

```
>cnv850 sample.test -t=c
```

B. 3. 4 変換仕様

ここでは，C ソース・ファイル，およびアセンブラ・ソース・ファイルについて，変換内容の詳細を説明します。

（1）C ソース・ファイル

（a）データの内部表現

CA850 では double 型は 32 ビットで float 型と同一ですが，CX では double 型は 64 ビットです。

CA850 用ソースで“double”と記述した場合は“float”に変換します。

CA850 用ソース	`double`
CX 用への変換後	`float`

（b）デバイス指定

CA850 では C ソース上でのデバイス指定が可能ですが，CX では C ソース上での指定ができず，デバイス指定ディレクティブは削除します。

なお，CX でデバイスを指定するには，-C オプションにて指定する必要があります。

CA850 用ソース	`#pragma cpu` *デバイス名*
CX 用への変換後	削除

（c）データのセクション割り当て

CA850 のデータのセクション指定の書式を以下に示します。

```
#pragma section セクション種別 [" セクション名 "] begin
変数の宣言／定義
#pragma section セクション種別 [" セクション名 "] end
```

CX のデータのセクション指定の書式を以下に示します。

```
#pragma section 属性指定文字 [" セクション名 "]
変数の宣言／定義
```

セクション種別と属性指定文字は同じもので，書式のみ異なるので，CX 用書式に変換します。また，end 指定行は不要なので削除します。

CA850 用ソース	`#pragma section` *セクション種別* `"` *セクション名* `" begin` `#pragma section` *セクション種別* `"` *セクション名* `" end`
CX 用への変換後	`#pragma section` *属性指定文字* `"` *セクション名* `"` `#pragma section default`

(d) レジスタ名を用いた周辺機能レジスタへのアクセス

- アクセス方法

CA850 と CX ではレジスタ名を使用するための #pragma ioreg 記述は同一であるため，変換は行いません。

- ビット・アクセス

CA850 にある I/O レジスタのビット・アクセスは，CX では削除されました。これを補うために，ビット・フィールドの型宣言とマクロにて置き換えを行います。

I/O レジスタのビット・アクセスがあった場合，ファイル先頭に型宣言，およびマクロを出力し，アクセス部ではマクロ呼び出しに変更します。

ただし，本マクロを使用したビット・アクセスでは，コンパイル時にリード・ライト属性のチェックはできません。デバイスのユーザーズ・マニュアルを参照の上，アクセスしてください。

ソース・コンバータは I/O レジスタの型が不明であるため，ビット位置に応じて 8，16，32 ビットのビット・フィールドを作成します。以下の例では，3，および 5 は 8 ビットに収まるため，ビット・フィールドは 8 ビットのものを作成します。

CA850 用ソース	`i = PAHL.3;` `PAHL.5 = 0;`

CX 用への変換後	`#ifndef __BIT8`
	`typedef struct {`
	`unsigned int b0:1;`
	`unsigned int b1:1;`
	`unsigned int b2:1;`
	`unsigned int b3:1;`
	`unsigned int b4:1;`
	`unsigned int b5:1;`
	`unsigned int b6:1;`
	`unsigned int b7:1;`
	`} __Bits8;`
	
	`#define __BIT8(name,bit)(((__Bits8*)&name)->b##bit)`
	`#endif`
	
	`i = __BIT8(PAHL,3);`
	`__BIT8(PAHL,5) = 0;`

(e) アセンブラ命令の記述

CA850 と CX はアセンブラ命令の記述方法が同一であるため，変換は行いません。

ただし，疑似命令のように差分がある記述については，ユーザが手で変換する必要があります。

(f) インライン展開の指定

CA850 と CX はインライン展開の指定方法が同一であるため，変換は行いません。

(g) 名前指定セクションへの関数指定

CA850 と CX は名前指定セクションへの関数指定方法が同一であるため，変換は行いません。

(h) 割り込みレベルの設定

CA850 と CX は割り込みレベルの設定方法が同一であるため，変換は行いません。

(i) 割り込み禁止の記述

CA850 と CX は割り込み禁止の記述方法が同一であるため，変換は行いません。

(j) 割り込み／例外処理用ハンドラの記述

CA850 の割り込みハンドラの書式を以下に示します。

```
#pragma interrupt 割り込み要求名 関数名 [ 配置方法 ]
__interrupt 関数定義または関数宣言
__multi_interrupt 関数定義または関数宣言
```

CX の割り込みハンドラの書式を以下に示します。

```
#pragma interrupt 割り込み要求名 関数名 [ 配置方法 ] [ オプション ]
```

関数指示子 __interrupt，__multi_interrupt は廃止され，多重割り込みについてはオプションで指定するようになったため，これらの指示子は削除します。

#pramga interrupt については，CX の仕様に変換します。関数に関数指示子 __multi_interrupt が付属していた場合は，#pragma interrupt に multi オプションを設定します。

また，#pragma interrupt の記述がなく，関数指示子のみ記述されていた場合は，#pragma interrupt NO_VECT を出力します。

CA850 用ソース	1. #pragma interrupt 割り込み要求名 関数名 配置方法 __interrupt 関数定義 2. #pragma interrupt 割り込み要求名 関数名 配置方法 __multi_interrupt 関数定義 3. __interrupt 関数定義
CX 用への変換後	1. #pragma interrupt 割り込み要求名 関数名 配置方法 関数定義 2. #pragma interrupt 割り込み要求名 関数名 配置方法 multi 関数定義 3. #pragma interrupt NO_VECT 関数名 関数定義

なお，多重割り込みについては，CA850 ではユーザが割り込み関数内に ei 命令を記述する必要がありましたが，CX では関数プロローグ中に ei 命令を自動的に生成するようになりました。この点については，ユーザが手で変更する必要があります。

また，#define において，マクロ定義内に書かれた __interrupt と __multi_interrupt は変換ができないため，ユーザが手で変更する必要があります。

（k）RTOS 対応機能

CA850 と CX は RTOS 対応機能の記述方法が同一であるため，変換は行いません。

（l）命令の組み込み関数による記述

CA850 と CX は命令の組み込み関数の記述が上位コンパチブルであるため，変換は行いません。

（m）構造体パッキング機能

CA850 と CX は構造体パッキング機能の記述方法が同一であるため，変換は行いません。

（n）__rcopy 関数の呼び出し

CA850 と CX とでは ROM 化処理が異なるため，C ソース・ファイル中（main 関数など）で RAM 領域データのコピー関数（_rcopy ／ _rcopy1 ／ _rcopy2 ／ _rcopy4）を呼び出している場合，CX ではエラーとなります。このため，これらのコピー関数を参照している箇所に対して警告を出力します（変換は行いません）。

（2）アセンブラ・ソース・ファイル

アセンブラ・ソース・ファイルの変換仕様を以下に示します。

表 B―16　アセンブラ・ソース・ファイル変換仕様

変換前	変更後	情報種別	備考
;	改行	変更	マルチステートメント
--	;	変更	コメント
hi1	highw1	変更	
hi	highw	変更	
lo	loww	変更	
.text	.cseg text	変更	
.const	.cseg const	変更	
.sconst	.cseg sconst	変更	
.bss	.dseg bss	変更	
.data	.dseg data	変更	

変換前	変更後	情報種別	備考
.previous	変換しない	情報	対象行を削除しても正しくセクションを切り替えることができないため，セクションを変更するようメッセージを出力します。
.sbss	.dseg sbss	変更	
.sdata	.dseg sdata	変更	
.sebss	.dseg sebss	変更	
.sedata	.dseg sedata	変更	
.sibss	.dseg sibss	変更	
.sidata	.dseg sidata	変更	
.tibss	.dseg tibss	変更	
.tibss.byte	.dseg tibss.byte	変更	
.tibss.word	.dseg tibss.word	変更	
.tidata	.dseg tidata	変更	
.tidata.byte	.dseg tidata.byte	変更	
.tidata.word	.dseg tidata.word	変更	
.section ".pro_epi_runtime", text	.pro_epi_runtime .cseg text	変更	
.section ".text", text	.cseg text	変更	
.section ".data", data	.dseg data	変更	
.section ".sedata", data	.dseg sedata	変更	
.section ".sidata", data	.dseg sidata	変更	
.section ".tidata", data	.dseg tidata	変更	
.section ".tidata.byte", data	.dseg tidata.byte	変更	
.section ".tidata.word", data	.dseg tidata.word	変更	
.section ".bss", bss	.dseg bss	変更	
.section ".sebss", bss	.dseg sebss	変更	
.section ".sibss", bss	.dseg sibss	変更	
.section ".tibss", bss	.dseg tibss	変更	
.section ".tibss.byte", bss	.dseg tibss.byte	変更	
.section ".tibss.word", bss	.dseg tibss.word	変更	
.section ".sdata", sdata	.dseg sdata	変更	
.section ".sbss", sbss	.dseg sbss	変更	
.section ".const", const	.cseg const	変更	
.section ".sconst", const	.cseg sconst	変更	
.section ".version", comment	.version .vseg	変更	
.section "*name*", text	*name* .cseg text 注	変更	“*name*”は任意のセクション名です。

変換前	変更後	情報種別	備考
.section "*name*", const	*name* .cseg const 注	変更	同上
.section "*name*", bss	*name* .dseg bss 注	変更	同上
.section "*name*", data	*name* .dseg data 注	変更	同上
.section "*name*", sbss	*name* .dseg sbss 注	変更	同上
.section "*name*", sdata	*name* .dseg sdata 注	変更	同上
.section "*name*", comment	*name* .vseg 注	変更	同上
.section "*name*"	*name* .cseg text 注	変更	配置属性の指定がない場合です。
.vdbstrtab	変換しない	情報	対象行を削除してもセクションのコードは削除できないため，セクション自身を削除するようメッセージを出力します。
.vdebug	変換しない	情報	同上
.vline	変換しない	情報	同上
.ext_ent_size	$ext_ent_size	変更	
.ext_func	$ext_func	変更	
.file	変換しない	対象外	
.frame	.func	変更	スタック・サイズは 0 を設定します。
.set	.set	変更	シンボルの指定方法を変更します。
.size	削除	削除	
.align	変換しない	対象外	
.org	変換しない	対象外	
.byte val	.db val	変更	
.byte bit : val	変換しない	情報	
.float	変換しない	対象外	
.hword val	.dhw val	変更	
.hword bit : val	変換しない	情報	
.lcomm label, size, byte	.align byte label: .ds (size)	追加	
.shword val	.dshw val	変更	
.shword bit : val	変換しない	情報	
.space	.ds	変更	
.str	.db	変更	
.word val	.dw val	変更	
.word bit : val	変換しない	情報	
.comm	変換しない	対象外	

変換前	変更後	情報種別	備考
.extern	変換しない	対象外	
.globl	.public	変更	
.binclude	$binclude	変更	
.include	$include	変更	
.irepeat	.irp	変更	
.repeat	.rept	変更	
.else	$else	変更	
.elseif	$elseif	変更	
.elseifn	$elseifn	変更	
.endif	$endif	変更	
.if	$if	変更	
.ifdef	$ifdef	変更	
.ifn	$ifn	変更	
.ifndef	$ifndef	変更	
.exitm	変換しない	対象外	
.exitma	変換しない	対象外	
.endm	変換しない	対象外	
.local	変換しない	対象外	
.macro	.macro	変更	マクロ名，仮パラメータの指定方法を変更します。
.option asm	削除	削除	
.option az_info_j	削除	削除	
.option az_info_r	削除	削除	
.option az_info_ri	削除	削除	
.option c	削除	削除	
.option callt	$callt	変更	
.option cpu	$processor	変更	
.option data	$data	変更	
.option ep_label	$ep_label	変更	
.option macro	$macro	変更	
.option mask_reg	削除	削除	
.option new_fcall	削除	削除	
.option no_ep_label	$no_ep_label	変更	
.option nomacro	$nomacro	変更	
.option nooptimize	削除	削除	
.option novolatile	削除	削除	
.option nowarning	$nowarning	変更	

変換前	変更後	情報種別	備考
.option optimize	削除	削除	
.option sdata	$sdata	削除	
.option volatile	削除	削除	
.option warning	$warning	変更	
.option reg_mode 5 5	$reg_mode 22	変更	
.option reg_mode 7 7	$reg_mode 26	変更	
.option reg_mode 10 10	$reg_mode 32	変更	
.option reg_mode x x	変換しない	情報	

注 セクション名 *name* の先頭文字が英字，および英字相当文字（@？_ . ~）以外の場合は，*name* をダブルクォーテーション（“”）で囲みます。
また，セクション名 *name* に英数字以外の文字が含まれている場合も，ダブルクォーテーション（“”）で囲みます。

付録C 索 引

【ま行】

【ら行】

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2012.09.01	―	初版発行

CubeSuite+ V1.03.00　ユーザーズマニュアル
ビルド編（CXコンパイラ）

発行年月日　2012年9月1日　　Rev.1.00

発行　　ルネサス エレクトロニクス株式会社
〒211-8668　神奈川県川崎市中原区下沼部1753

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル）

(03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/

Colophon 1.1

CubeSuite+ V1.03.00

RENESAS

ルネサス エレクトロニクス株式会社

R20UT2142JJ0100