User's Manual

CubeSuite+ V2.02.00

統合開発環境

ユーザーズマニュアル　RX ビルド編

対象デバイス

RX ファミリ

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　2014.05

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。

このマニュアルの使い方

このマニュアルは，RX 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，RX 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、XXXXH または XXXXh

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	R20UT2444J	R20UT2444E
	V850 設計編	R20UT2134J	R20UT2134E
	RL78 設計編	R20UT2864J	R20UT2864E
	78K0R 設計編	R20UT2137J	R20UT2137E
	78K0 設計編	R20UT2138J	R20UT2138E
	RX コーディング編	R20UT2999J	R20UT2999E
	V850 コーディング編	R20UT0553J	R20UT0553E
	コーディング編（CX コンパイラ）	R20UT2659J	R20UT2659E
	RL78, 78K0R コーディング編	R20UT2774J	R20UT2774E
	78K0 コーディング編	R20UT2141J	R20UT2141E
	RX ビルド編	このマニュアル	R20UT2998E
	V850 ビルド編	R20UT0557J	R20UT0557E
	ビルド編（CX コンパイラ）	R20UT2142J	R20UT2142E
	RL78, 78K0R ビルド編	R20UT2623J	R20UT2623E
	78K0 ビルド編	R20UT0783J	R20UT0783E
	RX デバッグ編	R20UT2875J	R20UT2875E
	V850 デバッグ編	R20UT2446J	R20UT2446E
	RL78 デバッグ編	R20UT2867J	R20UT2867E
	78K0R デバッグ編	R20UT0732J	R20UT0732E
	78K0 デバッグ編	R20UT0731J	R20UT0731E
	解析編	R20UT2868J	R20UT2868E
	メッセージ編	R20UT2871J	R20UT2871E

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

1. 概　　説

この章では，RX ファミリ向け C/C++ コンパイラが行うコンパイル処理の概要と，プログラム開発の事例を説明します。

1.1 概　　要

ビルド・ツール（CC-RX）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，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。

CC-RX は，以下に示す実行ファイルで構成されています。

(1) ccrx: コンパイルドライバ

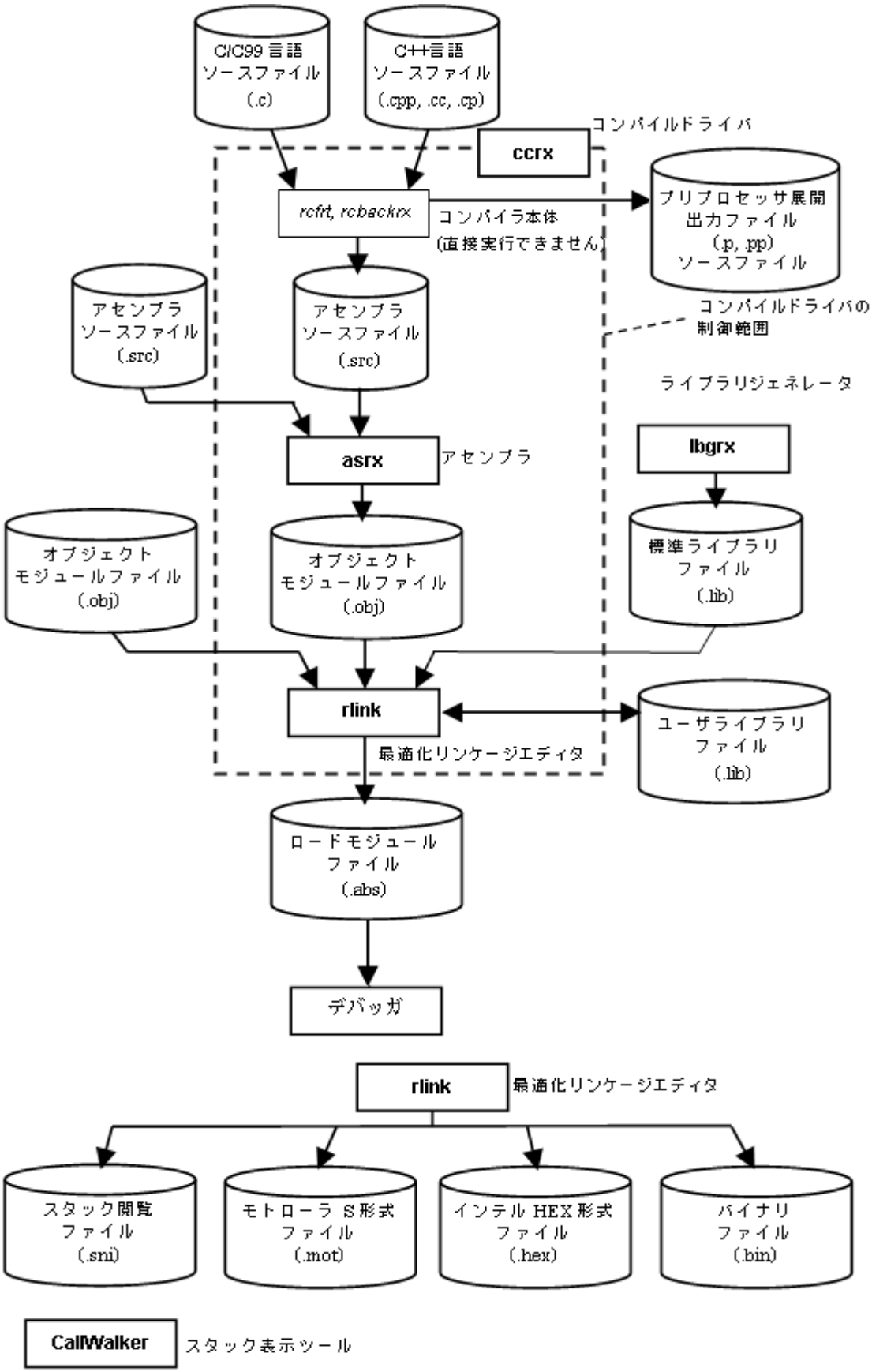
(2) asrx: アセンブラ

(3) rlink: 最適化リンケージエディタ

(4) lbgrx: ライブラリ・ジェネレータ

以下に，ビルド・ツールの処理の流れを示します。

図 1.1　ビルド・ツールの処理の流れ



1.2 著作権について

本ソフトウェアは LLVM Release License に従い、LLVM 技術をベースに開発されました。
その他のソフトウェア構成物は、ルネサス エレクトロニクス株式会社が著作権を有します。

s

2. 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。

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+ 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（「2.12 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。

備考 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（「2.12.6 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。

備考 2.　オブジェクト・モジュール・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（「2.12.2 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，ライブラリ・ジェネレータ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」，「2.9 ライブラリ・ジェネレート・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（「2.13 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（「2.13.1 更新ファイルのビルドを実行する」参照）
- リビルド（「2.13.2 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（「2.13.3 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（「2.13.4 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 (C)］タブ（C ソース・ファイルの場合），［個別コンパイル・オプション (C++)］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+ 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+ 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（「2.12 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。

備考 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（「2.12.6 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」，「2.7 リンク・オプションを設定する」，「2.8 ライブラリアン・オプションを設定する」参照）。

(5) ビルドの実行
ビルドを実行します（「2.13 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（「2.13.1 更新ファイルのビルドを実行する」参照）
- リビルド（「2.13.2 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（「2.13.3 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（「2.13.4 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 (C)］タブ（C ソース・ファイルの場合），［個別コンパイル・オプション (C++)］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+ 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。

備考 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。

備考 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，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 プロジェクトにファイルを追加する

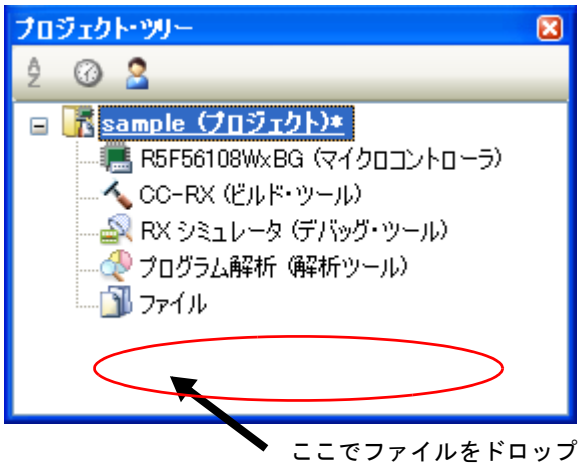
プロジェクトにファイルを追加するには，次の方法があります。

- 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 2.2 プロジェクト・ツリー パネル（ファイルのドロップ位置）

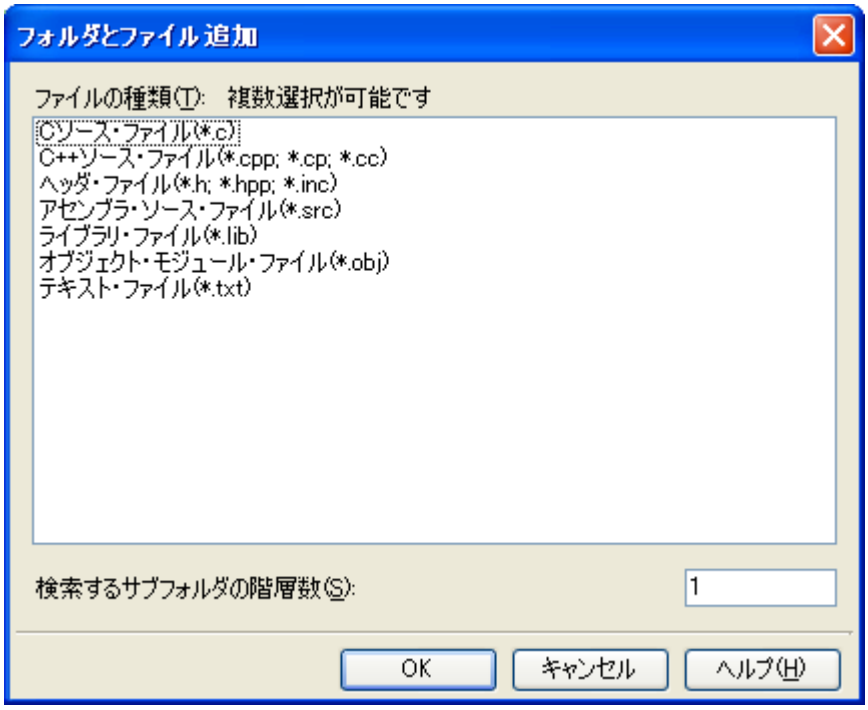


(b) フォルダ単位で追加する
エクスプローラなどからフォルダをドラッグし，プロジェクト・ツリー下部の空白部分にドロップすると，フォルダとファイル追加 ダイアログがオープンします。

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 2.3 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。

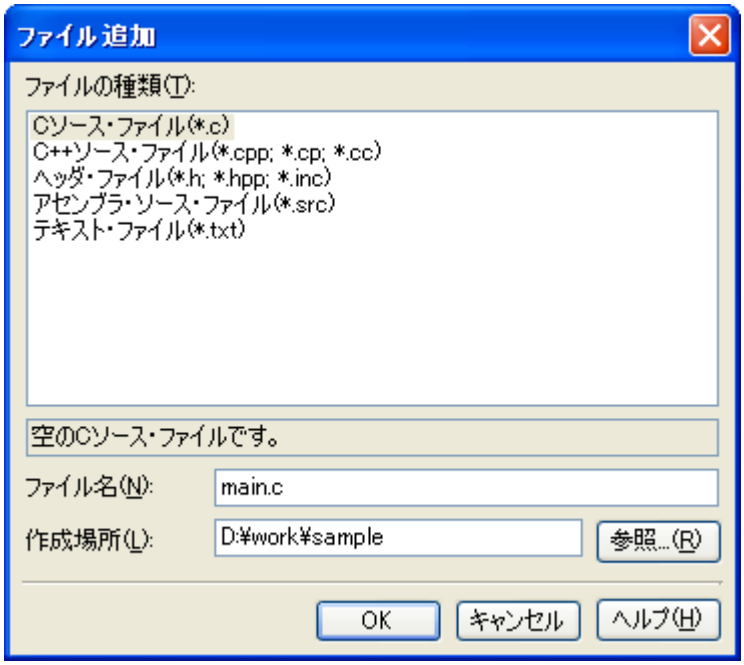
備考 ファイルの種類は，［Ctrl］キー＋左クリック，または［Shift］キー＋左クリックにより，複数選択することができます。
何も選択しない場合は，すべての種類を選択したものとみなします。

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，「2.3.4 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 2.4 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 2.5 プロジェクト・ツリー パネル（ファイル main.c 追加後）

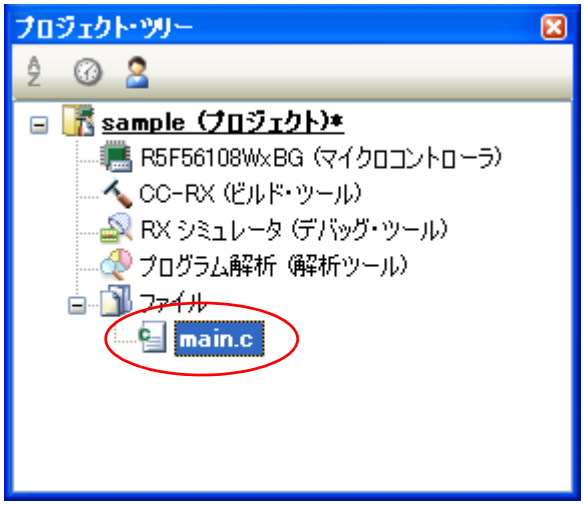


図 2.6 プロジェクト・ツリー パネル（フォルダ src 追加後）

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，「2.3.5 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，出力フォルダをそれぞれ異なるフォルダに設定してください。
C ソース・ファイルの出力フォルダの変更は，［個別コンパイル・オプション (C)］タブの［オブジェクト］カテゴリの［出力フォルダ］プロパティで行います。
C++ ソース・ファイルの出力フォルダの変更は，［個別コンパイル・オプション (C++)］タブの［オブジェクト］カテゴリの［出力フォルダ］プロパティで行います。
アセンブラ・ソース・ファイルの出力フォルダの変更は，［個別アセンブル・オプション］タブの［オブジェクト］カテゴリの［出力フォルダ］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。

注意 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。

注意 3. 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，空のファイルが作成されます。
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）
- C++ ソース・ファイル（*.cpp，*.cp，*.cc）
- アセンブラ・ソース・ファイル（*.src）
- ヘッダ・ファイル（*.h，*.hpp，*.inc）
- マップ・ファイル（*.map，*.lbp）
- ヘキサ・ファイル（*.hex）
- S レコード・ファイル（*.mot）
- 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。

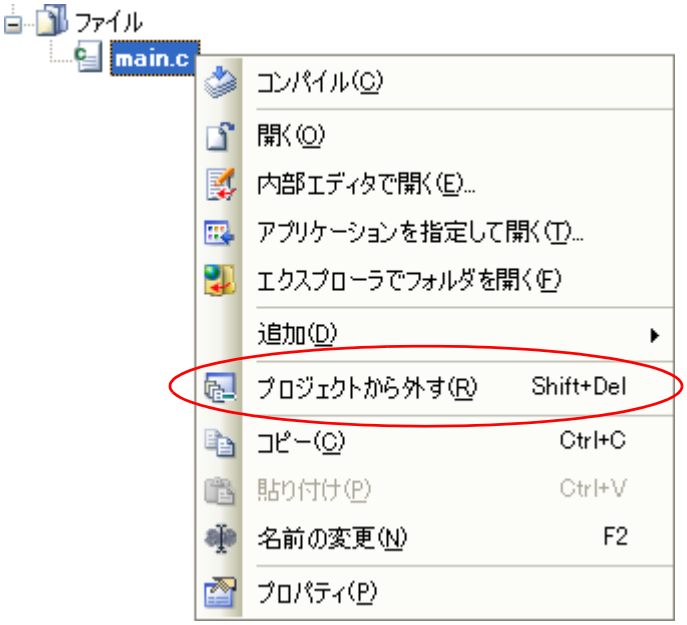
- ファイルをドラッグし，エディタ パネルにドロップする。
- ファイルを選択し，コンテキスト・メニューの［内部エディタで開く ...］を選択する。

備考 2. オプション ダイアログで，外部エディタを使用する設定になっている場合は，設定している外部エディタでオープンします。そうでないファイルは，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。

2.3.2 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。

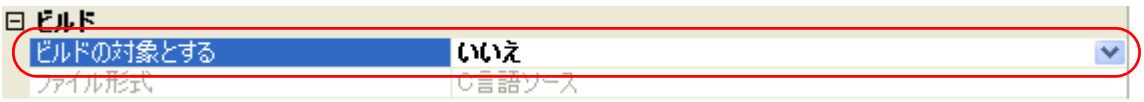
図 2.7 ［プロジェクトから外す］項目



2.3.3 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 2.8 ［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，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・ファイルです。

2.3.4 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 2.9 ［新しいカテゴリを追加］項目（ファイル・ノードの場合）

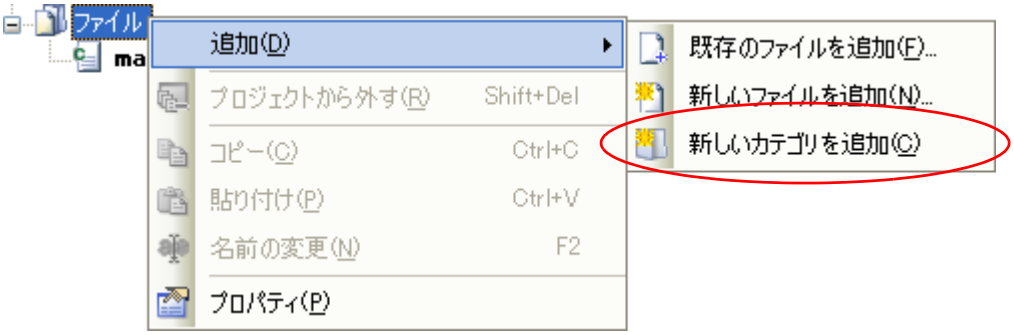
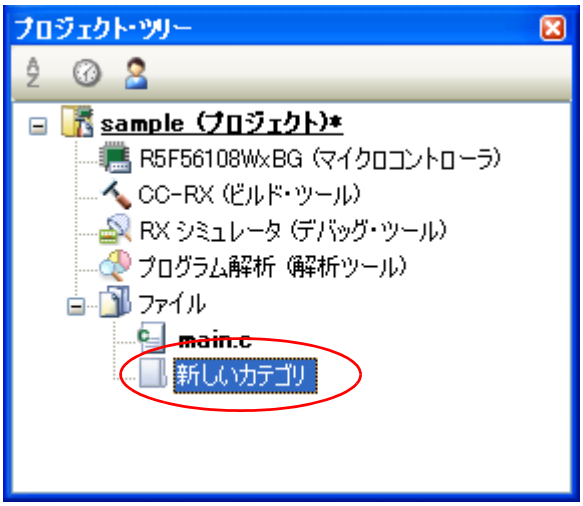


図 2.10 プロジェクト・ツリー パネル（カテゴリ・ノード追加後）



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。

備考 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。

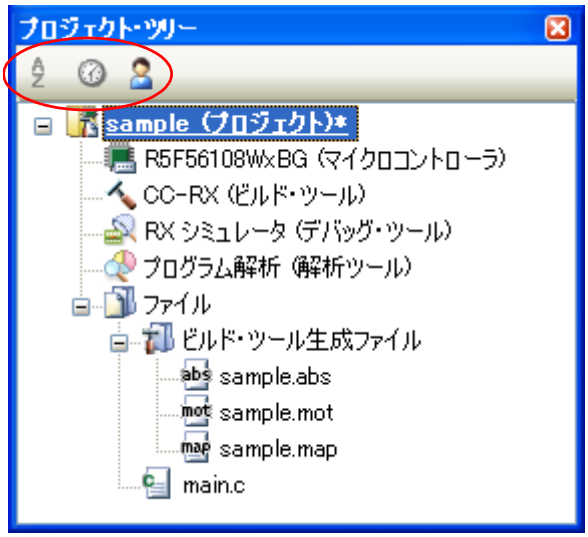
備考 3. カテゴリのネスト数の上限は 20 です。

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。

2.3.5 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 2.11 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。

2.3.6 ファイルの依存関係を更新する

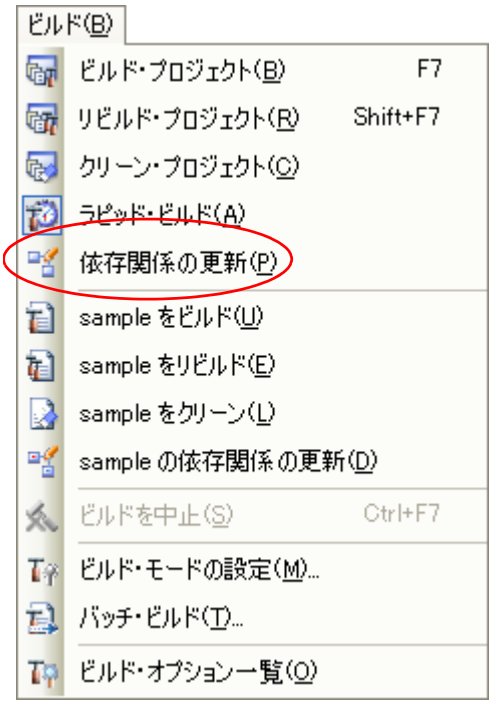
コンパイル・オプションの設定，アセンブル・オプションの設定で，ファイルの依存関係に影響する変更（インクルード・ファイルのパスの変更，C ソース・ファイル，C++ ソース・ファイル，およびアセンブラ・ソース・ファイル中にヘッダ・ファイルのインクルード文を追加など）を行った場合は，該当ファイルの依存関係を更新する必要があります。

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

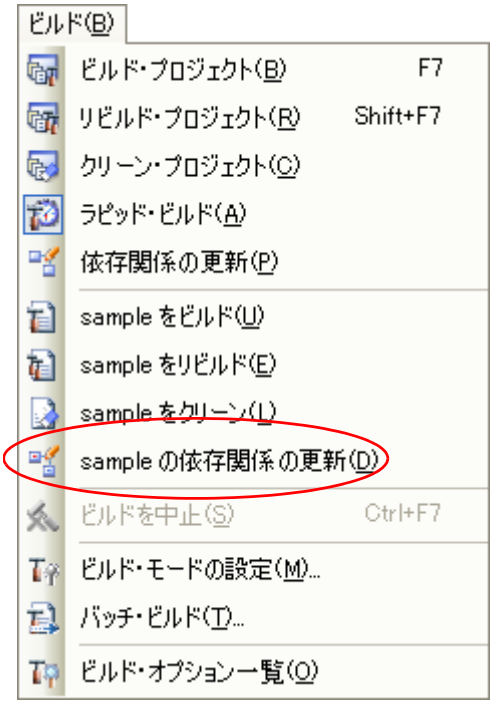
［ビルド］メニュー→［依存関係の更新］を選択してください。

図 2.12 ［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 2.13 ［*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。

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無視します。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誤認します（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 */
*/
```

注意 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無視します。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誤認します（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 */
```

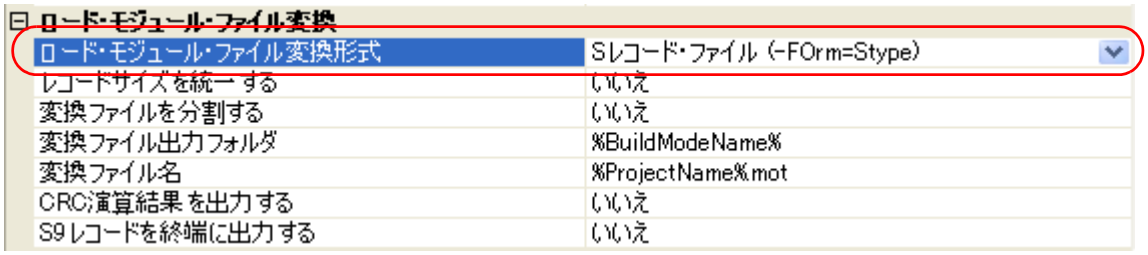
2.4 出力ファイルの種類を設定する

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。

(1) アプリケーション用のプロジェクト
ロード・モジュール・ファイルを出力します。
ロード・モジュール・ファイルがデバッグ対象となります。
ビルドの生成物として，ロード・モジュール・ファイルのほかに出力するファイルの種類を選択します。

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。［ロード・モジュール・ファイル変換］カテゴリの［ロード・モジュール・ファイル変換形式］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 2.14 ［ロード・モジュール・ファイル変換形式］プロパティ



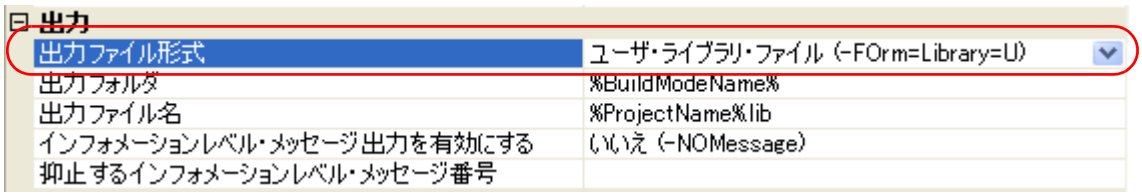
(a) ［ヘキサ・ファイル（-FOrm=Hexadecimal)］を選択した場合
ロード・モジュール・ファイルから，ヘキサ・ファイルを出力します。

(b) ［S レコード・ファイル（-FOrm=Stype)］を選択した場合（デフォルト）
ロード・モジュール・ファイルから，モトローラ S フォーマット・ファイルを出力します。

(c) ［バイナリ・データ・ファイル（-FOrm=Binary)］を選択した場合
ロード・モジュール・ファイルから，バイナリ・データ・ファイルを出力します。

(2) ライブラリ用のプロジェクト
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［ライブラリアン・オプション］タブを選択します。［出力］カテゴリの［出力ファイル形式］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 2.15 ［出力ファイル形式］プロパティ



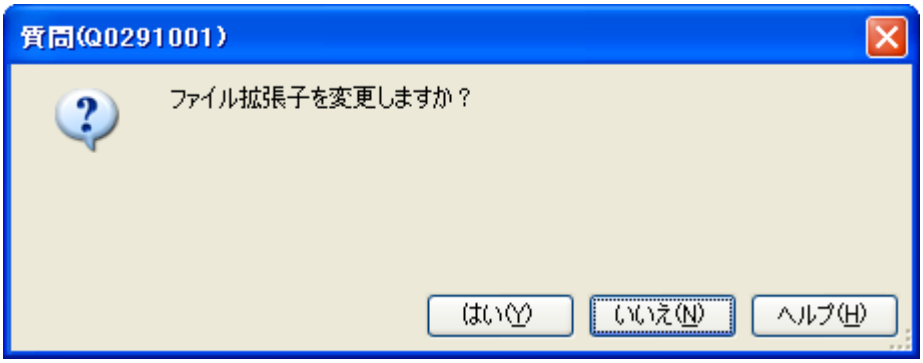
(a) ［ユーザ・ライブラリ・ファイル（-FOrm=Library=U)］を選択した場合（デフォルト）
ユーザ・ライブラリ・ファイルを出力します。

(b) ［システム・ライブラリ・ファイル（-FOrm=Library=S)］を選択した場合
システム・ライブラリ・ファイルを出力します。

(c) ［リロケータブル・モジュール・ファイル（-FOrm=Relocate)］を選択した場合
リロケータブル・モジュール・ファイルを出力します。

出力ファイルの拡張子が変更される場合，以下のメッセージ ダイアログがオープンします。

図 2.16 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，ファイルの拡張子を出力ファイルの種類に応じた拡張子に置き換えます。［いいえ］をクリックすると，現在のファイルの拡張子を置き換えません。

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，ヘキサ・ファイル，S レコード・ファイル，バイナリ・データ・ファイル，リロケータブル・モジュール・ファイル，ライブラリ・ファイルのファイル名は，デフォルトで次の名前が設定されています。

ロード・モジュール・ファイル名：%ProjectName%.abs
ヘキサ・ファイル名 ： %ProjectName%.hex
S レコード・ファイル名： %ProjectName%.mot
バイナリ・データ・ファイル名：%ProjectName%.bin
リロケータブル・モジュール・ファイル名：%ProjectName%.rel
ライブラリ・ファイル名： %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］カテゴリの［出力ファイル名］プロパティにおいて，変更するファイル名を入力してください。

図 2.17 ［出力ファイル名］プロパティ（ロード・モジュール・ファイルの場合）

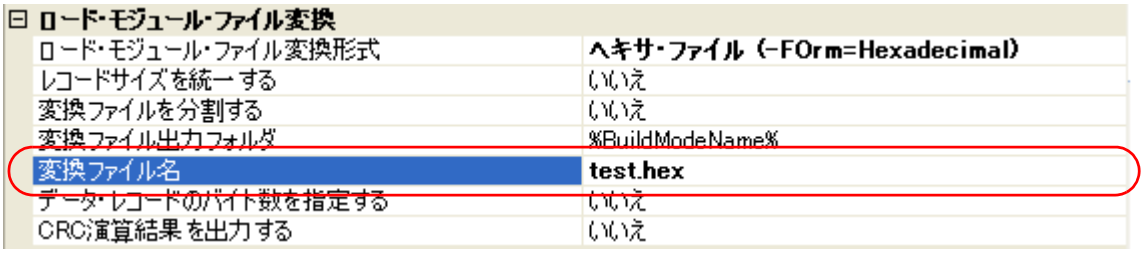
⊟ 出力	
出力ファイル形式	ロード・モジュール・ファイル (-FOrm=Absolute)
デバッグ情報を出力する	はい (出力ファイル内) (-DEBug)
⊞ ROMからRAMへマップするセクション	ROMからRAMへマップするセクション[3]
ロード・モジュール・ファイルを分割する	いいえ
出力フォルダ	%BuildModeName%
出力ファイル名	test.abs
外部シンボル割り付け情報ファイルを出力する	いいえ
インフォメーションレベル・メッセージ出力を有効にする	いいえ (-NOMessage)
抑止するインフォメーションレベル・メッセージ番号	
セクション終端にパディング・データを埋め込む	いいえ
⊞ 特定ベクタ番号のアドレス	特定ベクタ番号のアドレス[0]
可変ベクタの空き領域のアドレス	
ジャンプ・テーブルを出力する	いいえ

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。

備考 ［ロード・モジュール・ファイルを分割する］プロパティで［はい］を選択した場合，複数のロード・モジュール・ファイルに分割出力が可能です。

(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［リンク・オプション］タブを選択します。［ロード・モジュール・ファイル変換］カテゴリの［変換ファイル名］プロパティにおいて，変更するファイル名を入力してください。

図 2.18 ［変換ファイル名］プロパティ（ヘキサ・ファイルの場合）

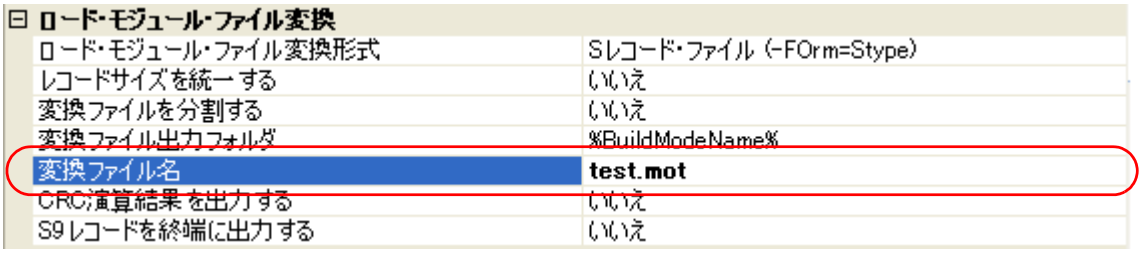


本プロパティは，次のプレースホルダに対応しています。
%ActiveProjectName%: アクティブ・プロジェクト名に置換します。
%MainProjectName%: メイン・プロジェクト名に置換します。
%ProjectName%: プロジェクト名に置換します。

備考 ［変換ファイルを分割する］プロパティで［はい］を選択した場合，複数のヘキサ・ファイルに分割出力が可能です。

(3) S レコード・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。［ロード・モジュール・ファイル変換］カテゴリの［変換ファイル名］プロパティにおいて，変更するファイル名を入力してください。

図 2.19 ［変換ファイル名］プロパティ（S レコード・ファイルの場合）

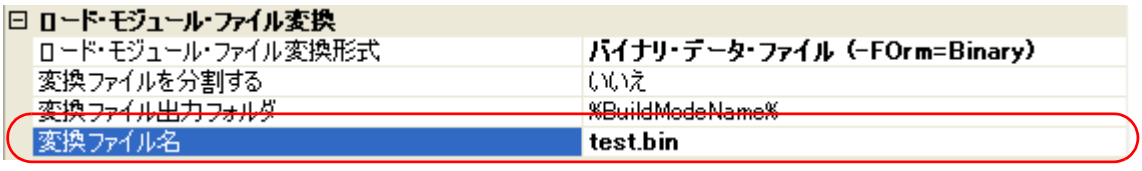


本プロパティは，次のプレースホルダに対応しています。
%ActiveProjectName%: アクティブ・プロジェクト名に置換します。
%MainProjectName%: メイン・プロジェクト名に置換します。
%ProjectName%: プロジェクト名に置換します。

備考 ［変換ファイルを分割する］プロパティで［はい］を選択した場合，複数の S レコード・ファイルに分割出力が可能です。

(4) バイナリ・データ・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。［ロード・モジュール・ファイル変換］カテゴリの［変換ファイル名］プロパティにおいて，変更するファイル名を入力してください。

図 2.20 ［変換ファイル名］プロパティ（バイナリ・データ・ファイルの場合）

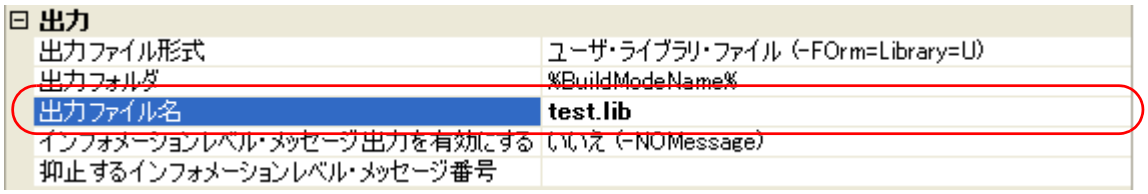


本プロパティは，次のプレースホルダに対応しています。
%ActiveProjectName%: アクティブ・プロジェクト名に置換します。
%MainProjectName%: メイン・プロジェクト名に置換します。
%ProjectName%: プロジェクト名に置換します。

備考 ［変換ファイルを分割する］プロパティで［はい］を選択した場合，複数のバイナリ・データ・ファイルに分割出力が可能です。

(5) ユーザ・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［ライブラリアン・オプション］タブを選択します。［出力］カテゴリの［出力ファイル名］プロパティにおいて，変更するファイル名を入力してください。

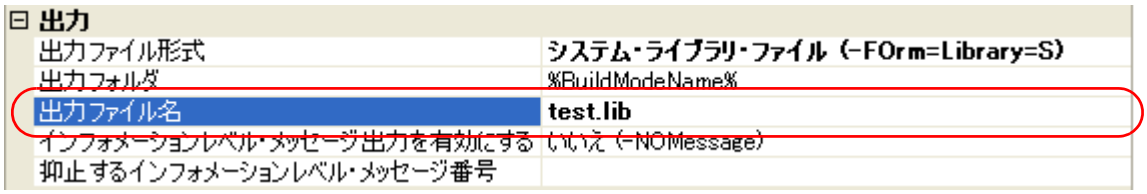
図 2.21 ［出力ファイル名］プロパティ（ユーザ・ライブラリ・ファイルの場合）



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。

(6) システム・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［ライブラリアン・オプション］タブを選択します。［出力］カテゴリの［出力ファイル名］プロパティにおいて，変更するファイル名を入力してください。

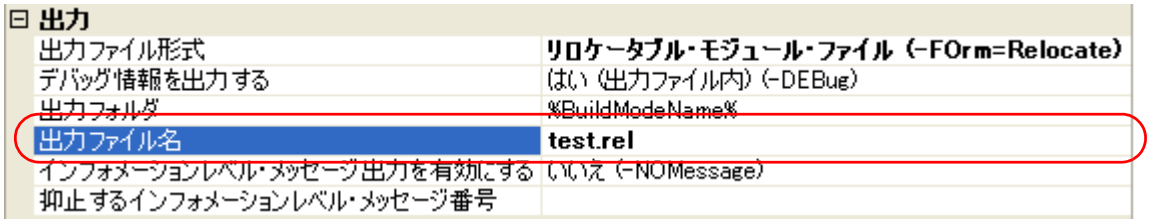
図 2.22 ［出力ファイル名］プロパティ（システム・ライブラリ・ファイルの場合）



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。

(7) リロケータブル・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［ライブラリアン・オプション］タブを選択します。［出力］カテゴリの［出力ファイル名］プロパティにおいて，変更するファイル名を入力してください。

図 2.23 ［出力ファイル名］プロパティ（リロケータブル・モジュール・ファイルの場合）



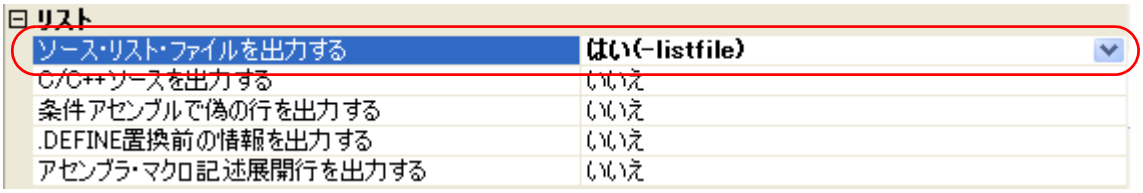
本プロパティは，次のプレースホルダに対応しています。
%ActiveProjectName%: アクティブ・プロジェクト名に置換します。
%MainProjectName%: メイン・プロジェクト名に置換します。
%ProjectName%: プロジェクト名に置換します。

2.4.2 アセンブル・リストを出力する

アセンブル結果はアセンブル・リスト・ファイルに出力されます。

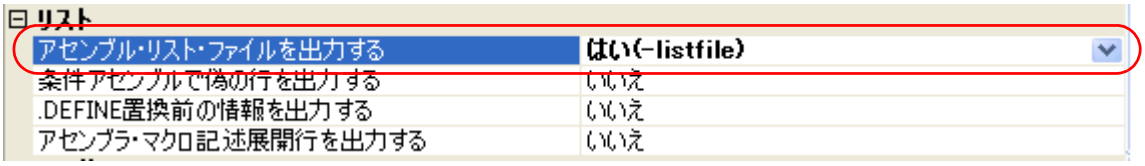
(1) C ソース・ファイル／C++ ソース・ファイルの場合
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。
アセンブル・リストを出力するには，［リスト］カテゴリの［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile)］を選択してください。

図 2.24 ［ソース・リスト・ファイルを出力する］プロパティ



(2) アセンブラ・ソース・ファイルの場合
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［アセンブル・オプション］タブを選択します。
アセンブル・リストを出力するには，［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 (-listfile)］を選択してください。

図 2.25 ［アセンブル・リスト・ファイルを出力する］プロパティ



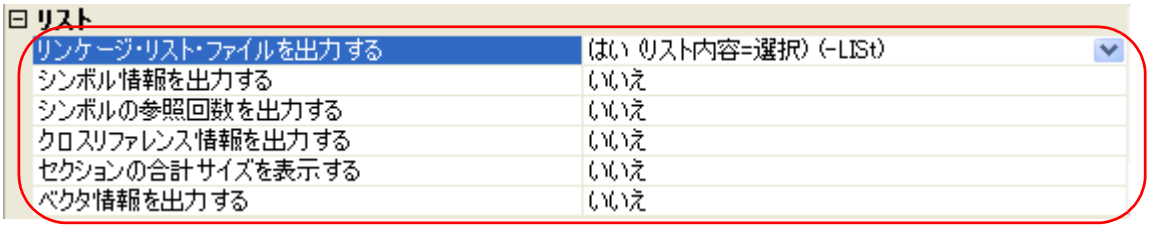
備考 アセンブル・リスト・ファイルについての詳細は，「3.1 アセンブル・リスト・ファイル」を参照してください。

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，リンケージ・リスト・ファイルに出力します。

(1) ロード・モジュール・ファイルの場合
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。
リンケージ・リスト・ファイルの出力の設定は，［リスト］カテゴリで行います。

図 2.26 ［リンケージ・リスト・ファイルを出力する］プロパティ



リンケージ・リスト・ファイルの出力の設定は，［リスト］カテゴリの［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］を選択してください。
リンケージ・リスト・ファイルを出力する場合，リンカが出力するリンケージ・リスト・ファイルの内容を選択することができます。

(a) シンボル情報の出力設定
［シンボル情報を出力する］プロパティで［はい (-SHow=SYmbol)］を選択してください。

(b) シンボルの参照回数の出力設定
［シンボルの参照回数を出力する］プロパティで［はい (-SHow=Reference)］を選択してください。

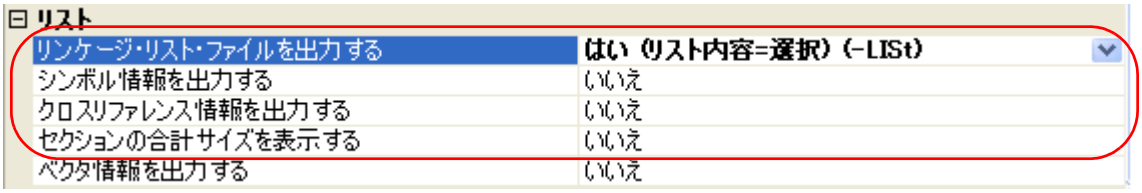
(c) クロスリファレンス情報の出力設定
［クロスリファレンス情報を出力する］プロパティで［はい (-SHow=Xreference)］を選択してください。

(d) セクションの合計サイズの出力設定
［セクションの合計サイズを表示する］プロパティで［はい (-SHow=Total_size)］を選択してください。

(e) ベクタ情報の出力設定
［ベクタ情報を出力する］プロパティで［はい (-SHow=VECTOR)］を選択してください。

(2) リロケータブル・モジュール・ファイルの場合
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［ライブラリアン・オプション］タブを選択します。
リンケージ・リスト・ファイルの出力の設定は，［リスト］カテゴリで行います。

図 2.27 ［リンケージ・リスト・ファイルを出力する］プロパティ



リンケージ・リスト・ファイルの出力の設定は，［リスト］カテゴリの［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］を選択してください。
リンケージ・リスト・ファイルを出力する場合，リンカが出力するリンケージ・リスト・ファイルの内容を選択することができます。

(a) シンボル情報の出力設定
［シンボル情報を出力する］プロパティで［はい (-SHow=SYmbol)］を選択してください。

(b) クロスリファレンス情報の出力設定
［クロスリファレンス情報を出力する］プロパティで［はい (-SHow=Xreference)］を選択してください。

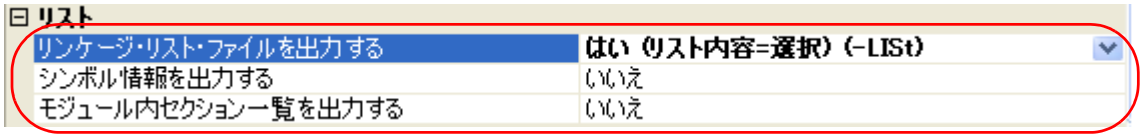
(c) セクションの合計サイズの出力設定
［セクションの合計サイズを表示する］プロパティで［はい (-SHow=Total_size)］を選択してください。

備考 リンケージ・リスト・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 ライブラリ情報を出力する

ライブラリ情報（リンク結果の情報）は，ライブラリ・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［ライブラリアン・オプション］タブを選択します。
ライブラリ・リスト・ファイルの出力の設定は，［リスト］カテゴリで行います。

図 2.28 ［リンケージ・リスト・ファイルを出力する］プロパティ



ライブラリ・リスト・ファイルを出力するには，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］を選択してください。
ライブラリ・リスト・ファイルを出力する場合，リンカが出力するライブラリ・リスト・ファイルの内容を選択することができます。

(1) モジュール内シンボル名一覧の出力設定
［シンボル情報を出力する］プロパティで［はい (-SHow=SYmbol)］を選択してください。

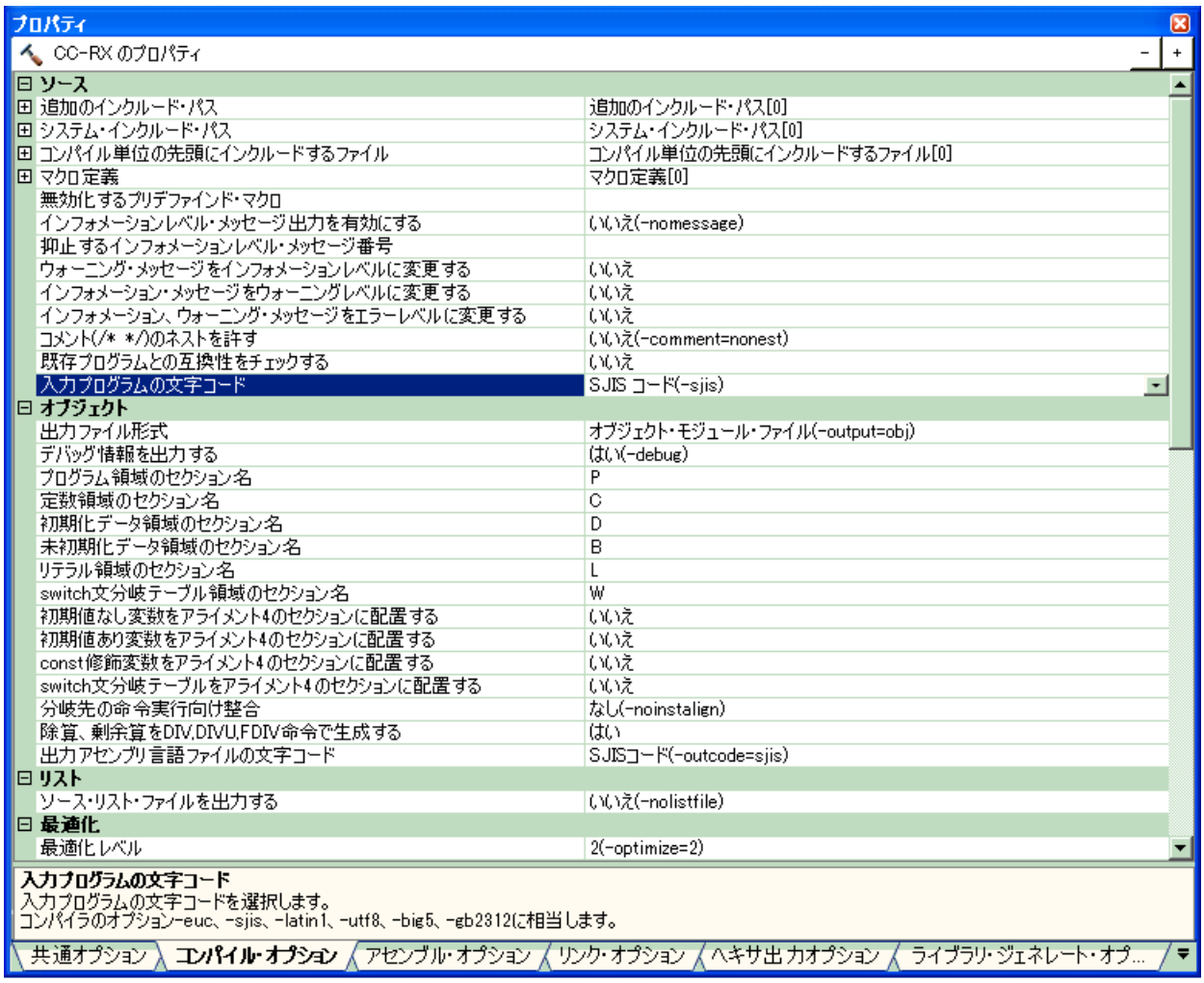
(2) 各モジュール内セクション名／シンボル名一覧の出力設定
［モジュール内セクション一覧を出力する］プロパティで［はい (-SHow=SEction)］を選択してください。

備考 ライブラリ・リスト・ファイルについての詳細は，「3.3 ライブラリ・リスト」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 2.29　プロパティ パネル：［コンパイル・オプション］タブ

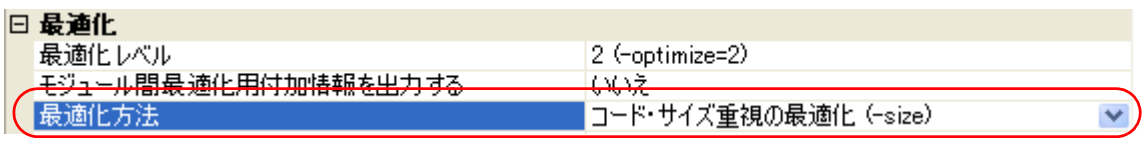


備考　　よく使うオプションについては，［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリにまとめられています。

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［最適化方法］プロパティで［コード・サイズ重視の最適化 (-size)］を選択してください。

図 2.30　［最適化方法］プロパティ（コード・サイズ優先の場合）

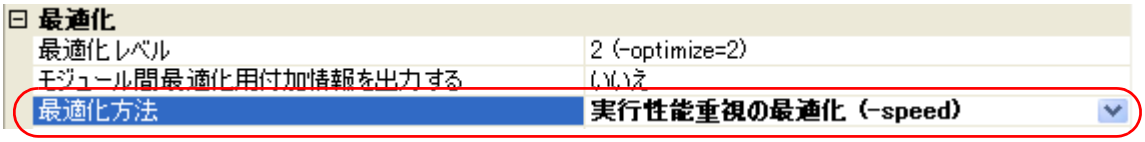


備考　　［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［最適化方法］プロパティでも，同様に設定することができます。

2.5.2 実行性能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。
実行性能を優先した最適化を行うには，［最適化］カテゴリの［最適化方法］プロパティで［実行性能重視の最適化 (-speed)］を選択してください（デフォルトでは，［コード・サイズ重視の最適化 (-size)］が選択されています）。

図 2.31　［サイズとスピード］プロパティ（実行速度優先の場合）

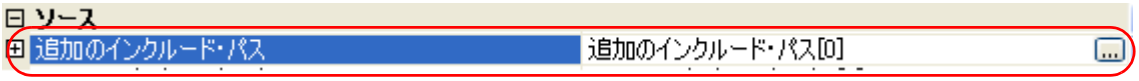


備考　［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［最適化方法］プロパティでも，同様に設定することができます。

2.5.3 インクルード・パスを追加する

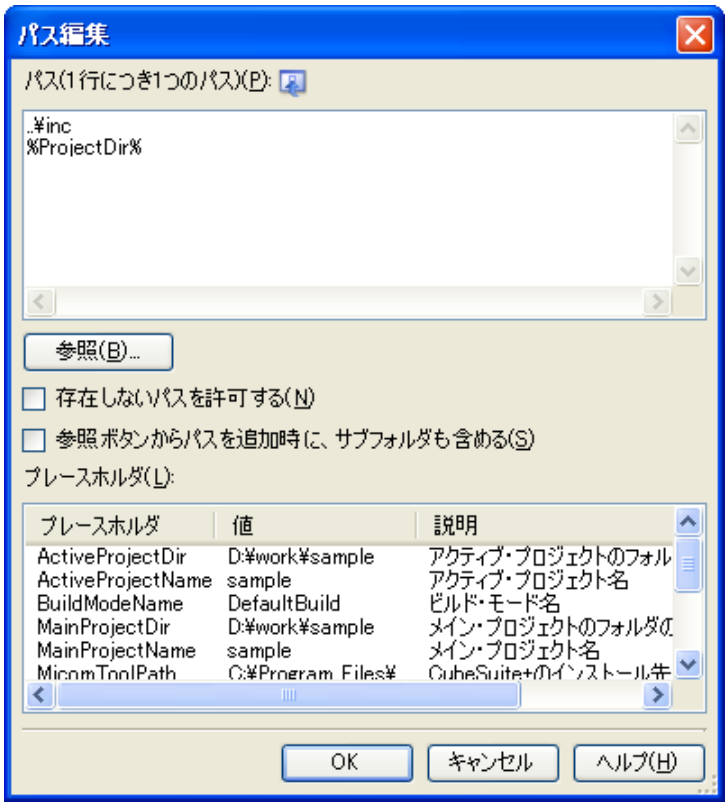
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。
インクルード・パスの設定は，[ソース］カテゴリの［追加のインクルード・パス］プロパティで行います。

図 2.32　［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 2.33　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 247 文字まで指定可能です。

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のドラッグ・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

備考 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 階層分までのパスが［パス (1 行につき 1 つのパス)］に追加されます。

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

図 2.34　［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）

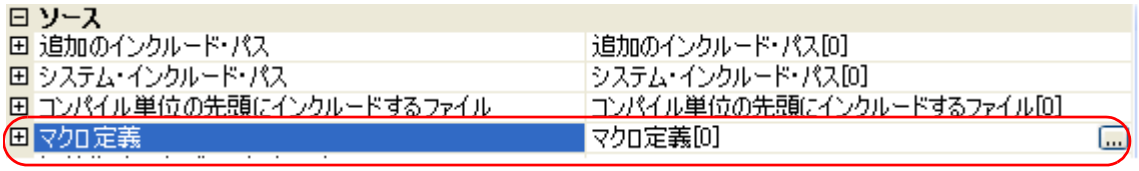
インクルード・パスの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。
また，プロジェクト・ツリーにインクルード・ファイルを追加すると，そのインクルード・パスをサブプロパティの一番最初に自動で追加します。

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 マクロ定義を設定する

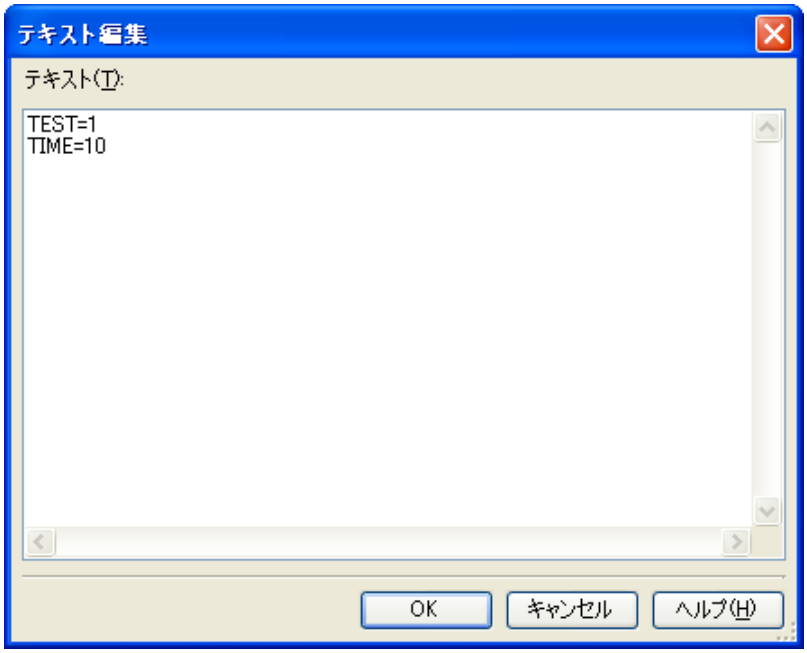
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［コンパイル・オプション］タブを選択します。
マクロ定義の設定は，［ソース］カテゴリの［マクロ定義］プロパティで行います。

図 2.35　［マクロ定義］プロパティ



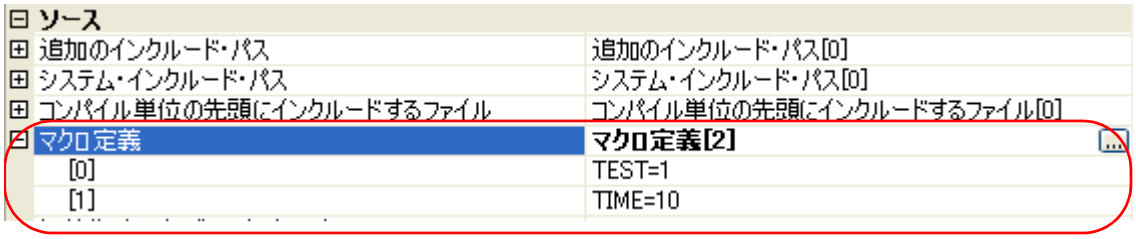
［...］ボタンをクリックすると，テキスト編集 ダイアログがオープンします。

図 2.36　　テキスト編集 ダイアログ



［テキスト］にマクロ定義を「*マクロ名* = *文字列*」の形式で 1 行に 1 つずつ入力します。
1 行に 256 文字まで，65535 行まで指定可能です。
「= *文字列*」の部分は省略可能で，省略した場合，そのマクロ名が定義されたものと仮定します。
［OK］ボタンをクリックすると，入力したマクロ定義をサブプロパティとして表示します。

図 2.37　　［マクロ定義］プロパティ（マクロ定義設定後）



マクロ定義の変更は，［...］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。

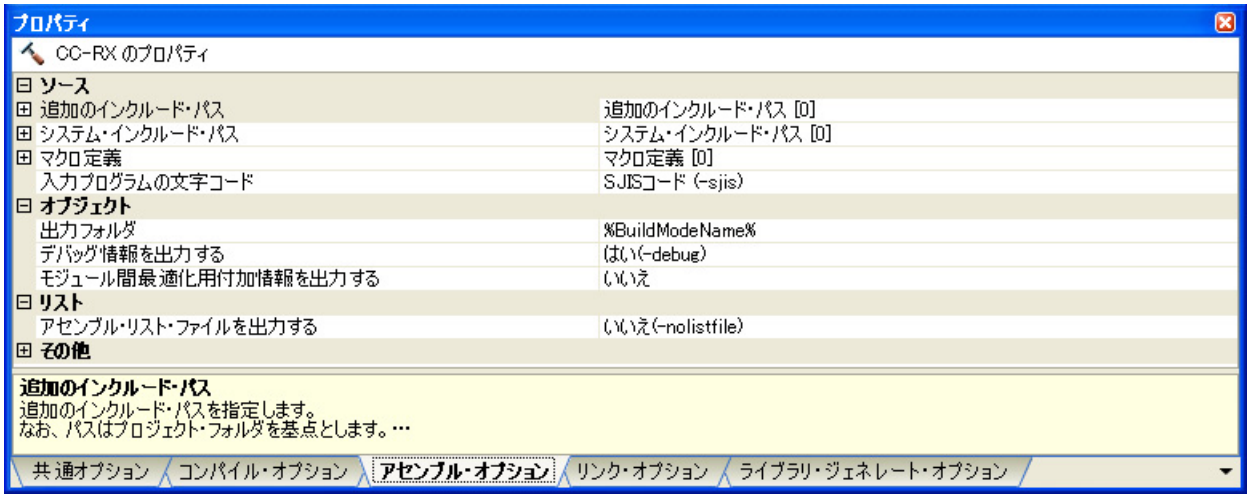
備考　　［共通オプション］タブの［よく使うオプション（コンパイル）］カテゴリの［マクロ定義］プロパティでも，同様に設定することができます。

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 2.38　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。

2.6.1 インクルード・パスを追加する

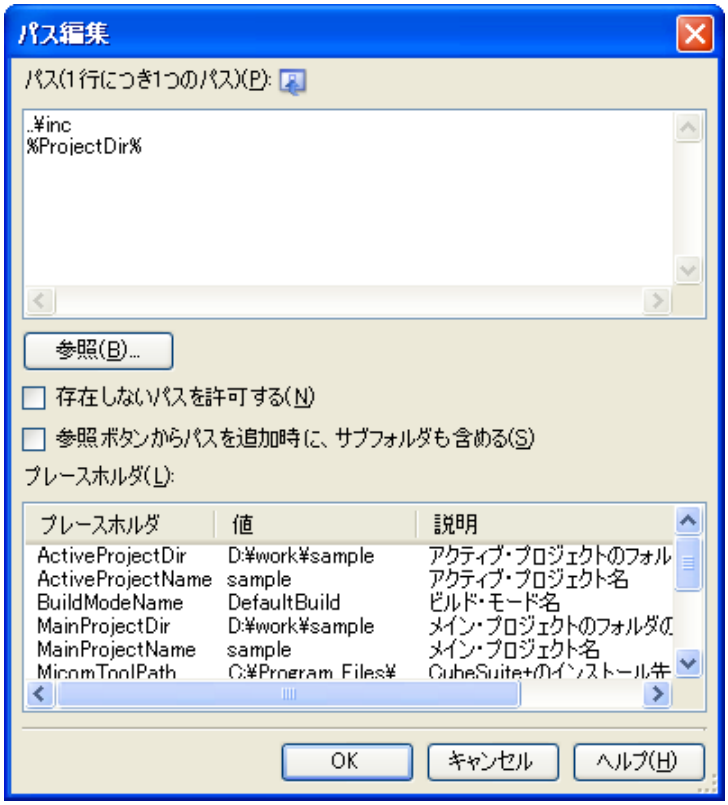
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［アセンブル・オプション］タブを選択します。

インクルード・パスの設定は，［ソース］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 2.39　［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 2.40　　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 247 文字まで指定可能です。

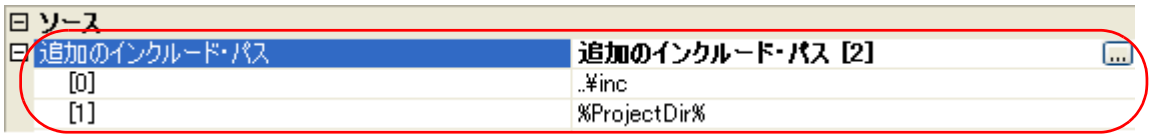
備考 1.　　インクルード・パスは，以下のいずれかの方法で指定することも可能です。

- エクスプローラなどからフォルダのドラッグ・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

備考 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 階層分までのパスが［パス (1 行につき 1 つのパス)］に追加されます。

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

図 2.41　　［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）



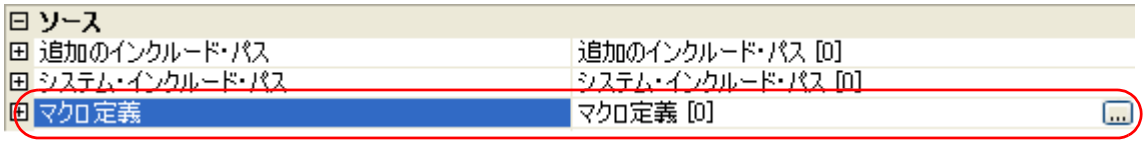
インクルード・パスの変更は，［...］ボタン，またはサブプロパティのテキスト・ボックスへの直接入力により行うことができます。
また，プロジェクト・ツリーにインクルード・ファイルを追加すると，そのインクルード・パスをサブプロパティの一番最初に自動で追加します。

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 マクロ定義を設定する

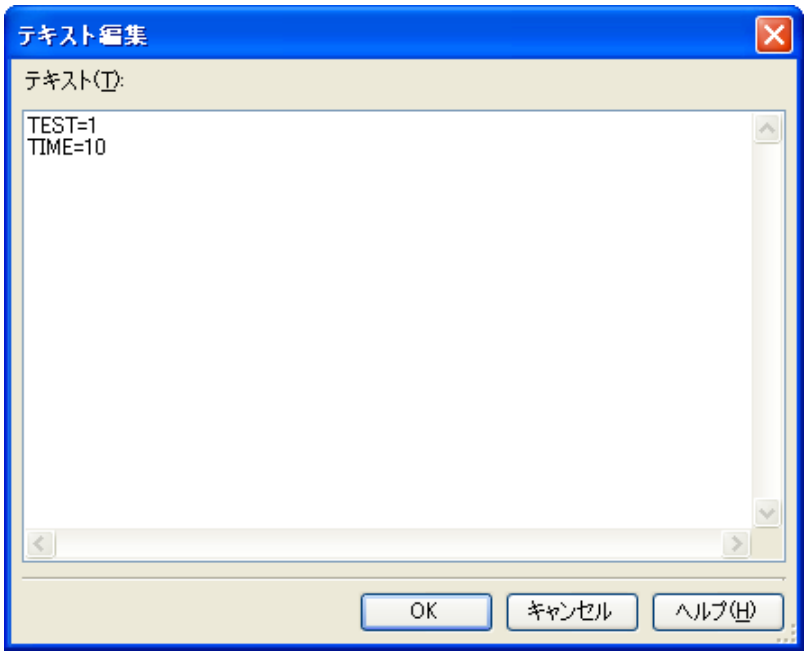
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［アセンブル・オプション］タブを選択します。
マクロ定義の設定は，［ソース］カテゴリの［マクロ定義］プロパティで行います。

図 2.42　［マクロ定義］プロパティ



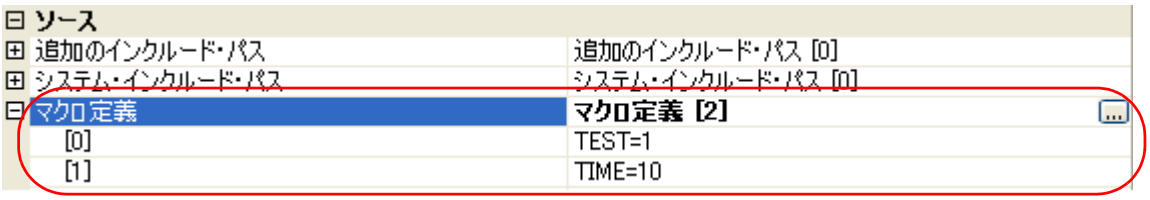
［...］ボタンをクリックすると，テキスト編集 ダイアログがオープンします。

図 2.43　テキスト編集 ダイアログ



［テキスト］にマクロ定義を「*マクロ名* = *文字列*」の形式で 1 行に 1 つずつ入力します。
1 行に 256 文字まで，65535 行まで指定可能です。
［OK］ボタンをクリックすると，入力したマクロ定義をサブプロパティとして表示します。

図 2.44　［マクロ定義］プロパティ（マクロ定義設定後）



マクロ定義の変更は，［...］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。

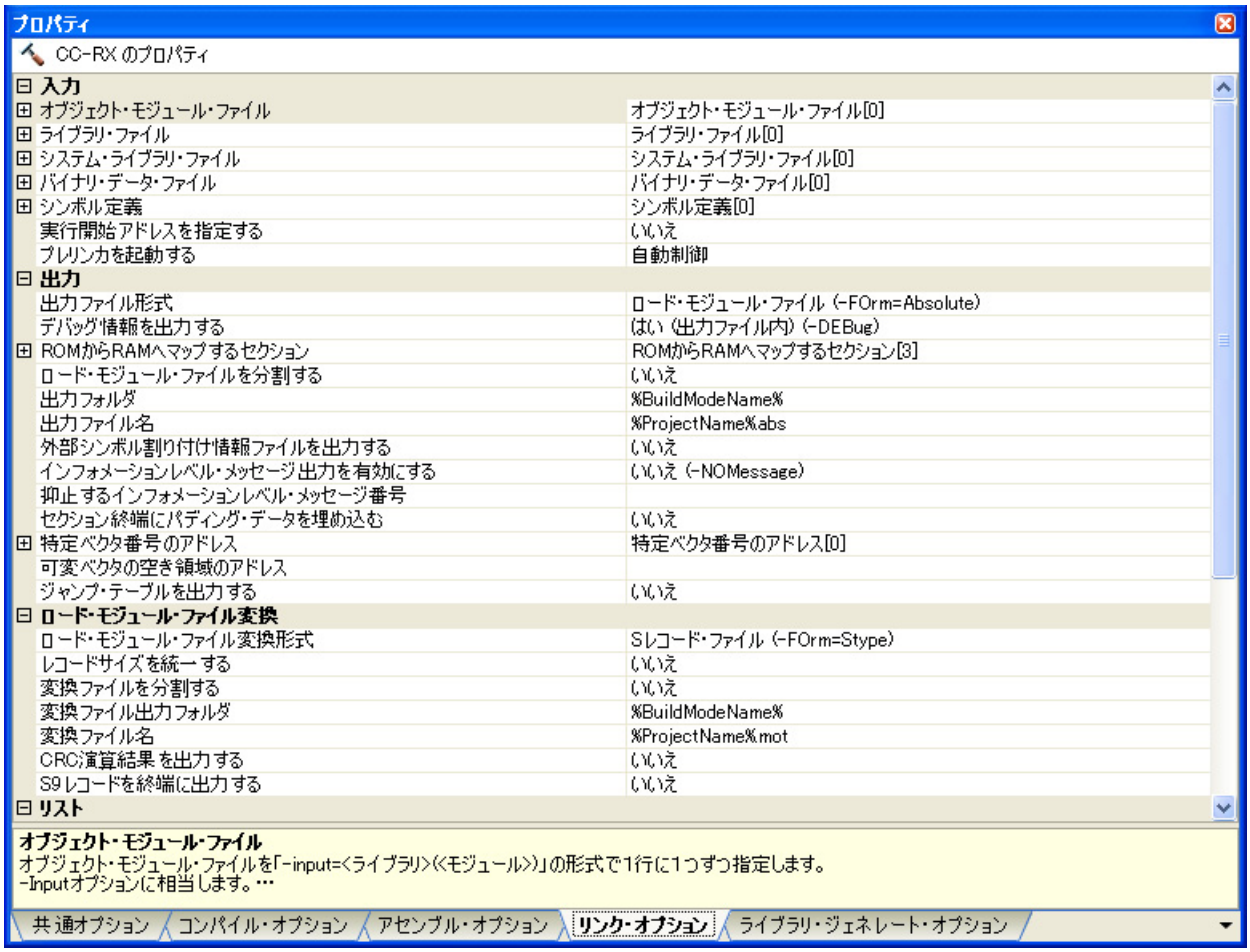
備考　［共通オプション］タブの［よく使うオプション（アセンブル）］カテゴリの［マクロ定義］プロパティでも，同様に設定することができます。

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 2.45　　プロパティ パネル：［リンク・オプション］タブ

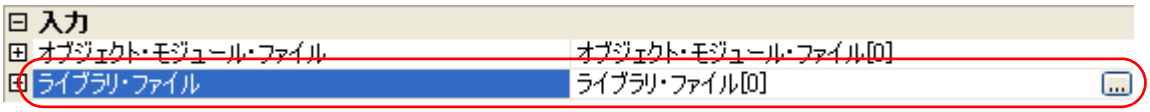


備考　　よく使うオプションについては，［共通オプション］タブの［よく使うオプション（リンク）］カテゴリにまとめられています。

2.7.1 ユーザ・ライブラリを追加する

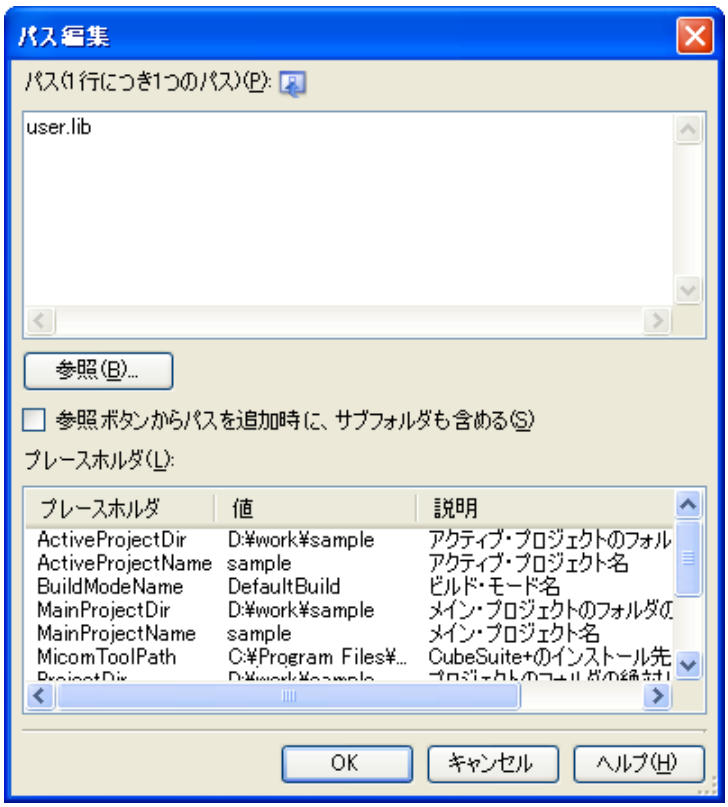
プロジェクト・ツリーでビルド・ツール・ノードを選択し，プロパティ パネルの［リンク・オプション］タブを選択します。
ユーザ・ライブラリの追加は，［入力］カテゴリの［ライブラリ・ファイル］プロパティで行います。

図 2.46　　［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，パス編集 ダイアログがオープンします。

図 2.47　　パス編集 ダイアログ



［パス (1 行につき 1 つのパス)］にライブラリ・ファイル名を 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.　　ライブラリ・ファイルは，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のドラッグ・アンド・ドロップ
- ［参照 ...］ボタンをクリックし，使用するライブラリ・ファイルを指定 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

備考 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 階層分までのパスが［パス (1 行につき 1 つのパス)］に追加されます。

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

図 2.48　　［ライブラリ・ファイル］プロパティ（ライブラリ・ファイル追加後）

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。

2.7.2 オーバーレイ・セクションの選択機能を使用するための準備をする

CC-RX が使用するリンカ（rlink）では，プログラム中に定義した複数のセクションを同じアドレスに割り付けることができます。このように割り付けられたセクションを “ オーバーレイ・セクション ” と呼びます。

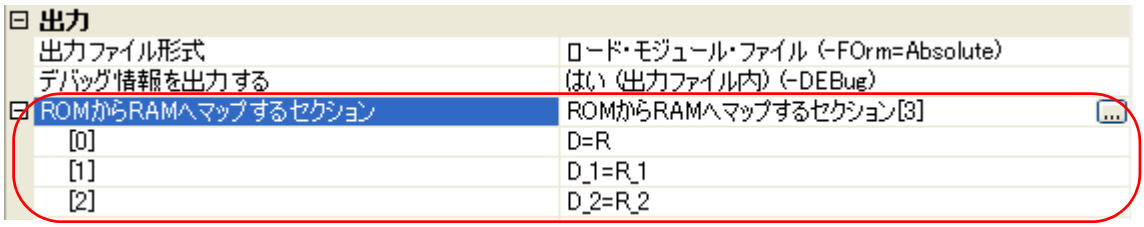
同じアドレスに割り付けられたセクションのうち，デバッグ対象とするオーバーレイ・セクション（優先セクション）を選択する機能をデバッグ・ツールが提供しています。この機能を“オーバーレイ・セクションの選択機能”と呼びます。
オーバーレイ・セクションを設定したロード・モジュールは，プログラム実行前に優先セクションを切り替えてデバッグすることが可能になります。
オーバーレイ・セクションの選択機能を実現するための，ロード・モジュールの作成方法を以下に示します。

(1) ROM 領域の内容を RAM にコピー
ROM 領域の内容を RAM 領域にコピーすることによって，コードやデータを RAM 上へ展開します。

(2) ビルド・オプションの設定
オーバーレイ・セクションの選択機能を対応するため，ROM から RAM へマップするセクション，オーバーレイ・セクションを設定します。
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［リンク・オプション］タブを選択します。

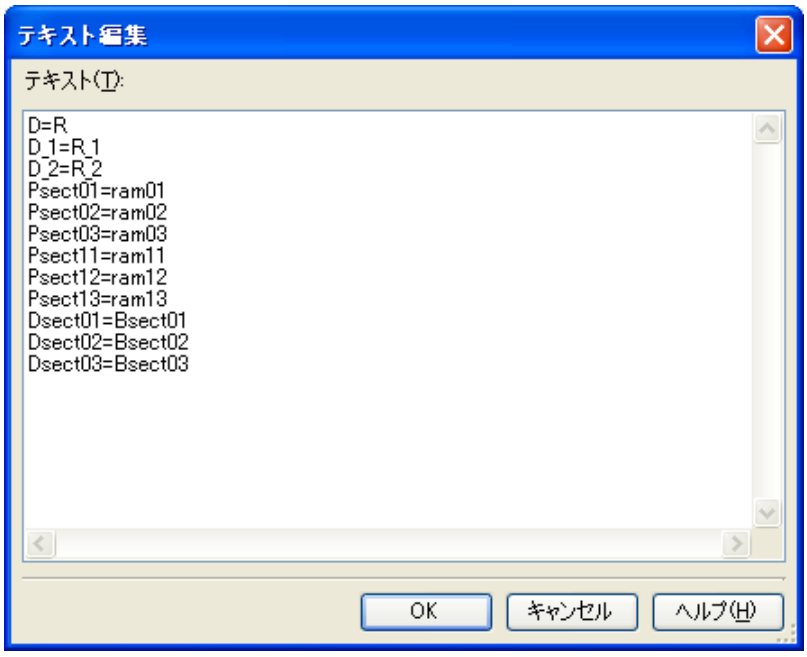
(a) ROM から RAM へマップするセクションの設定
［出力］カテゴリの［ROM から RAM へマップするセクション］プロパティで行います。
ROM セクションと同サイズの RAM セクションを確保し，ROM セクション内定義シンボルを RAM セクション上のアドレスでリロケーションします。

図 2.49 ［ROM から RAM へマップするセクション］プロパティ



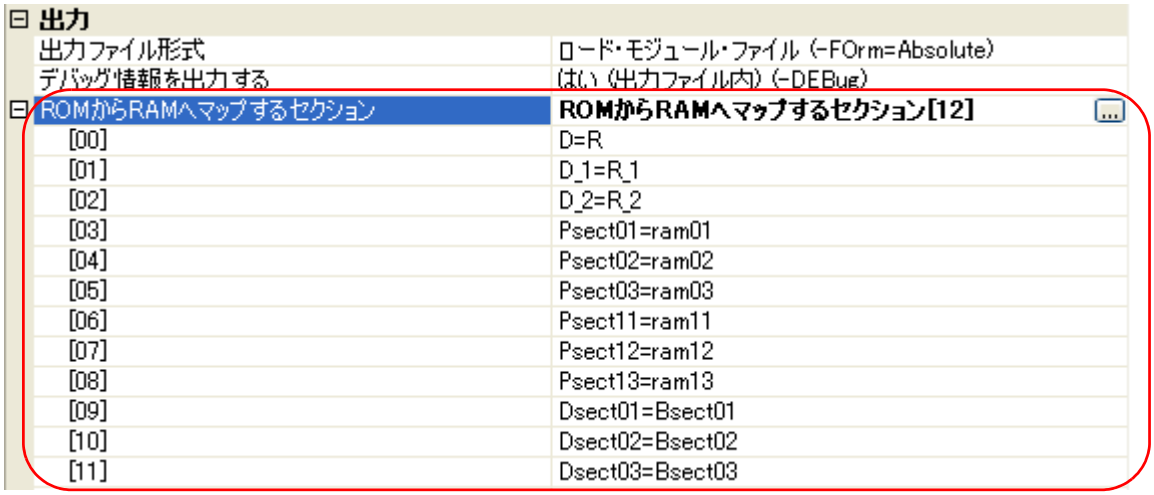
［...］ボタンをクリックすると，テキスト編集 ダイアログがオープンします。

図 2.50 テキスト編集 ダイアログ



［テキスト］にセクション名を「*ROM セクション名=RAM セクション名*」の形式で 1 行に 1 つずつ入力します。
1 行に 256 文字まで，30 行まで指定可能です。
［OK］ボタンをクリックすると，入力したセクション名がサブプロパティとして表示されます。

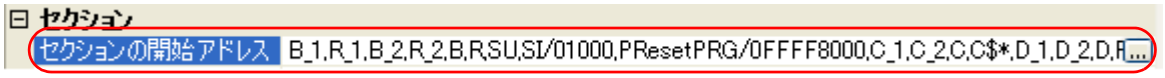
図 2.51　［ROM から RAM へマップするセクション］プロパティ（ROM から RAM へマップするセクション設定後）



セクション名の変更は，［...］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。

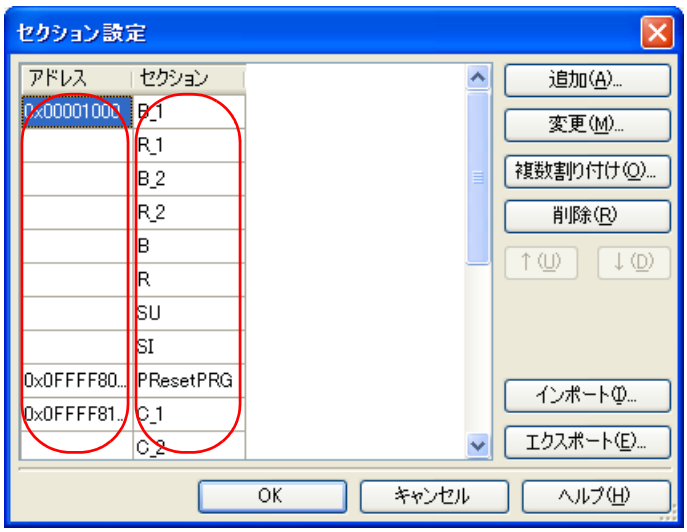
(b)　ROM セクション，RAM セクション（オーバーレイ・セクション）の設定
セクションを設定するには，［セクション］カテゴリの［セクションの開始アドレス］プロパティで行います。

図 2.52　［セクションの開始アドレス］プロパティ



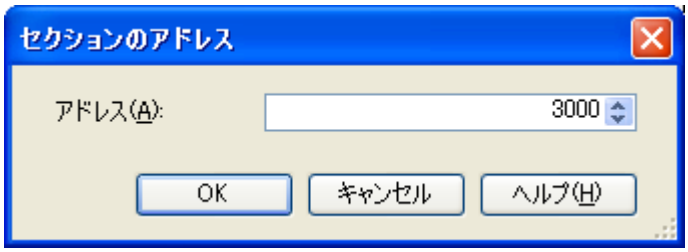
［...］ボタンをクリックすると，セクション設定 ダイアログがオープンします。このダイアログは，右下隅をドラッグしダイアログのサイズを変更することが可能です。

図 2.53　［セクション設定］ダイアログ



- セクションのアドレスの追加
セクション設定 ダイアログで［アドレス］エリアをポイントしたのち，［追加］ボタンをクリックすると，セクションのアドレス ダイアログがオープンします。

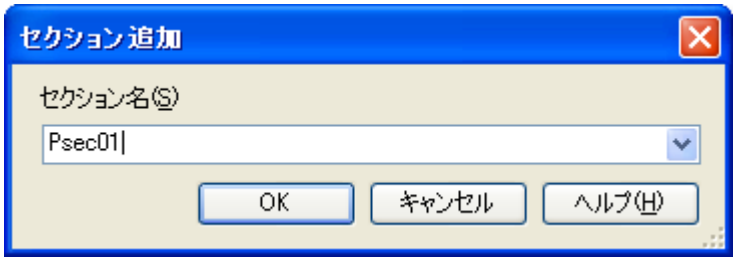
図 2.54　セクションのアドレス ダイアログ



［アドレス］に追加したいセクションのアドレスを入力します。［OK］ボタンをクリックすると，入力したアドレスがセクション設定 ダイアログの［アドレス］エリアに表示されます。

- ROM セクションの追加
  セクション設定 ダイアログで［セクション］エリアをポイントしたのち，［追加］ボタンをクリックすると，セクションの追加 ダイアログがオープンします。

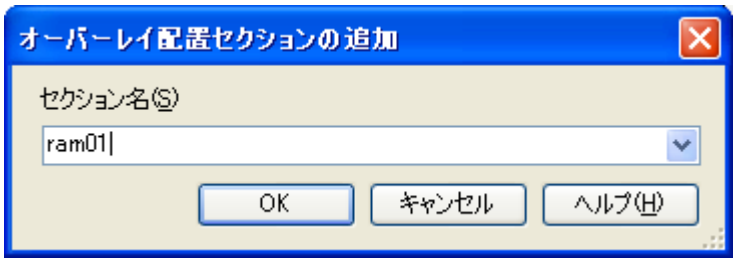
図 2.55　セクションの追加 ダイアログ



［セクション名］に追加したいセクション名を入力します。［OK］ボタンをクリックすると，入力したセクションがセクション設定 ダイアログの［セクション］エリアに表示されます。

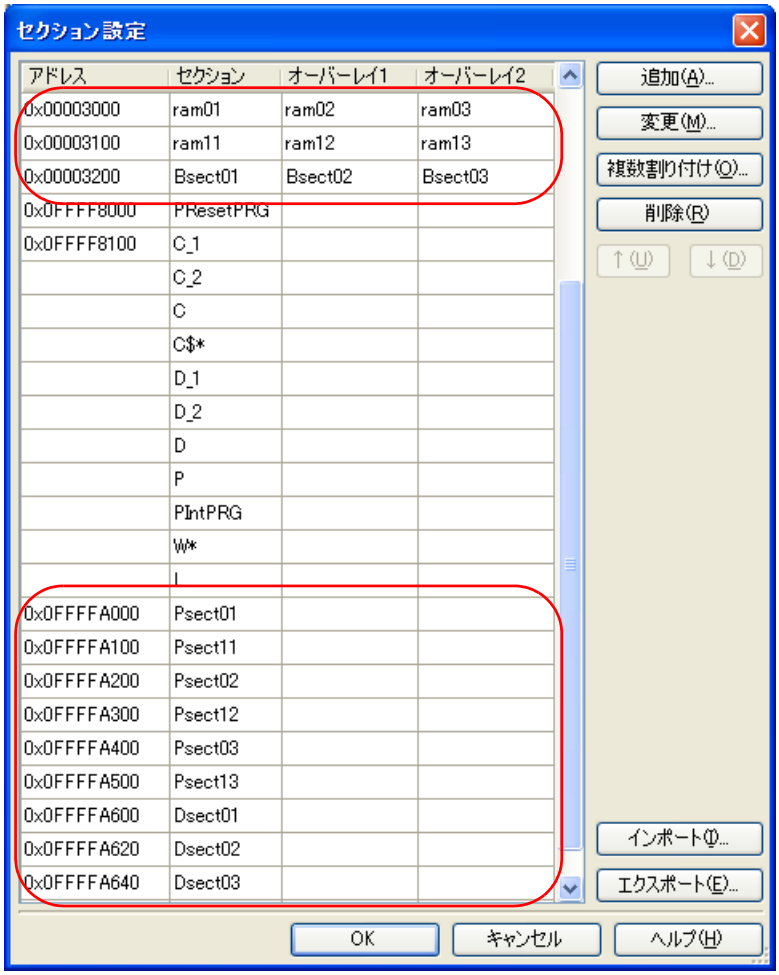
- RAM セクション（オーバーレイ・セクション）の追加
  セクション設定 ダイアログで［アドレス］エリアをポイントしたのち，［複数割り付け］ボタンをクリックすると，オーバーレイ配置セクションの追加 ダイアログがオープンします。

図 2.56　オーバーレイ配置セクションの追加 ダイアログ



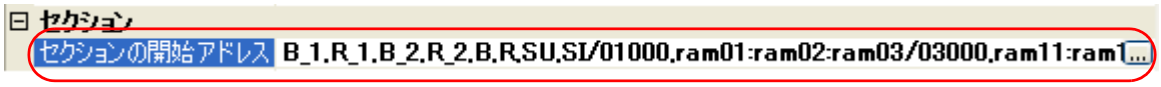
［セクション名］に追加したいセクション名を入力します。［OK］ボタンをクリックすると，入力したセクションがセクション設定 ダイアログの［セクション］エリアに表示されます。
同一アドレスに配置するセクション数分，同様に，セクション設定 ダイアログの［複数割り付け］ボタンをクリックし，セクションを追加します。追加したセクションは，［オーバーレイ *n*］エリアに表示されます（*n* は 1 で始まる数字です）。

図 2.57　　RAM セクション，ROM セクションの設定結果（[セクション設定] ダイアログ）



[OK] ボタンをクリックすると，設定した ROM セクション，RAM セクション（オーバーレイ・セクション）がテキスト・ボックスに表示されます。

図 2.58　　[セクションの開始アドレス] プロパティ（セクション設定後）



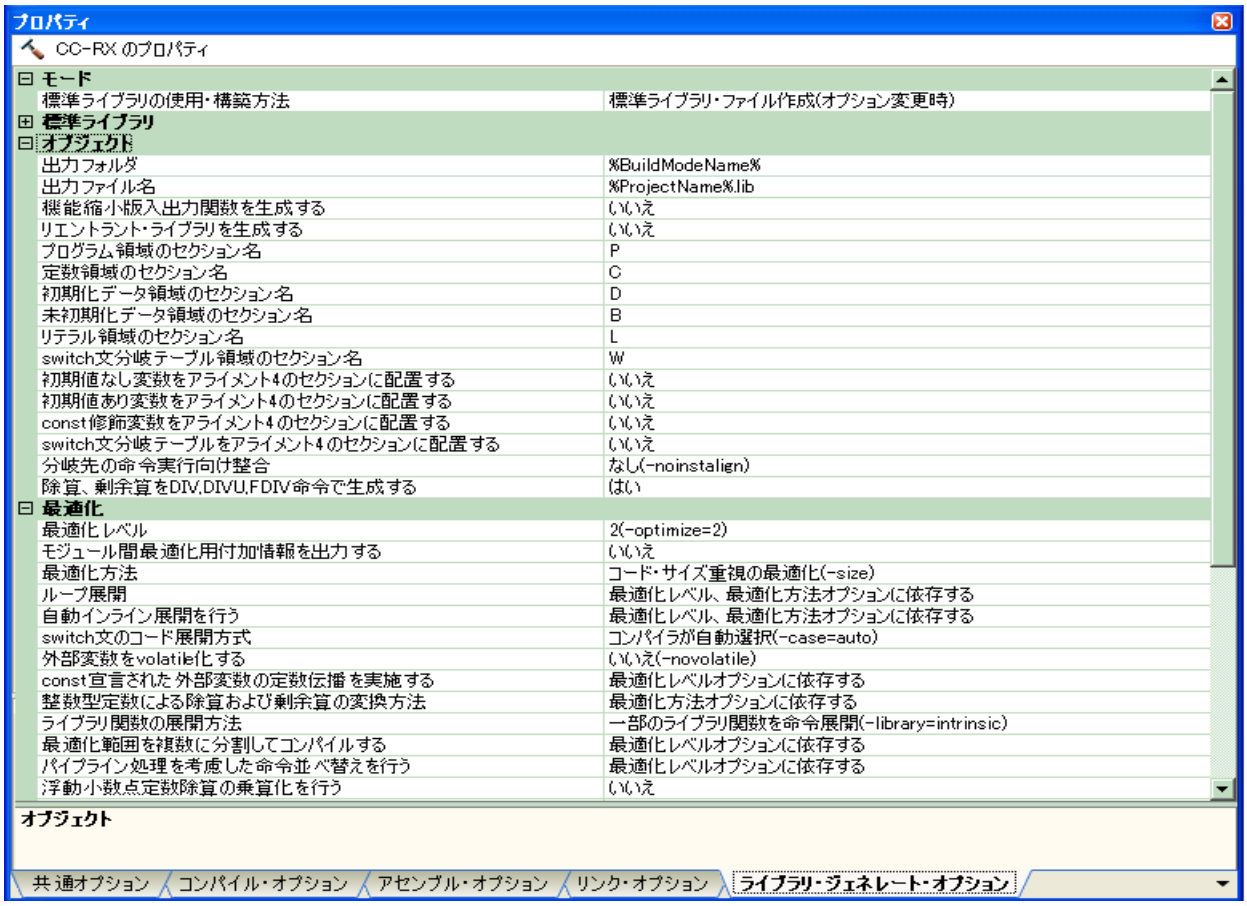
(3)　プロジェクトのビルドの実行
プロジェクトのビルドを実行します。
オーバーレイ・セクションの選択機能を実現するためのロード・モジュール・ファイルが生成されます。

2.8 ライブラリアン・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［ライブラリアン・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アン・オプションを設定することができます。

注意　　本タブは，アプリケーション用のプロジェクトの場合は表示しません。

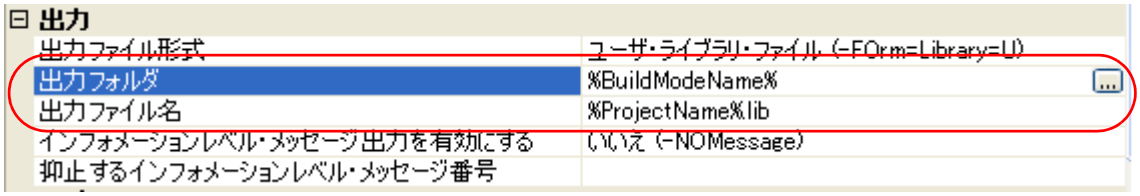
図 2.59　　プロパティ パネル：［ライブラリアン・オプション］タブ



2.8.1 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［ライブラリアン・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］カテゴリで行います。

図 2.60　　［出力］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。

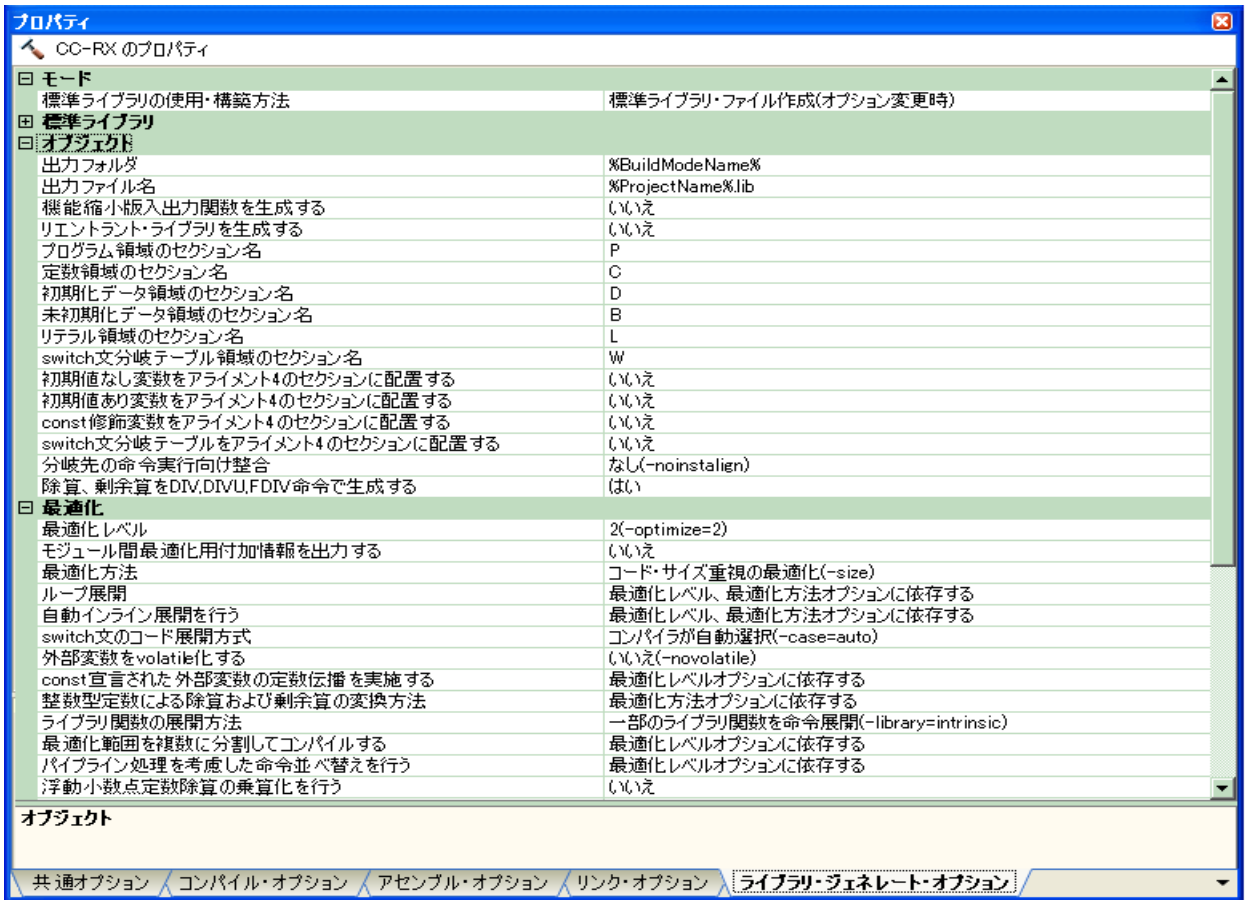
デフォルトでは，“%ProjectName%.lib” を設定しています。

2.9 ライブラリ・ジェネレート・オプションを設定する

ライブラリ・ジェネレータ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・ジェネレート・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・ジェネレート・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。

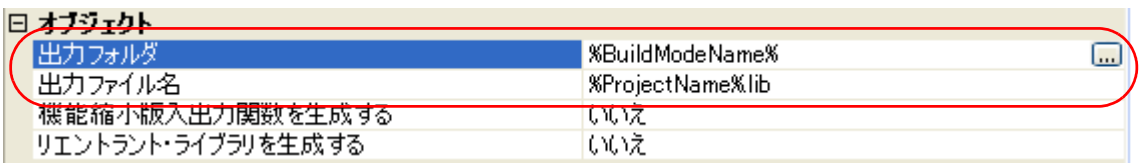
図 2.61　　プロパティ パネル：［ライブラリ・ジェネレート・オプション］タブ



2.9.1 標準ライブラリ・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・ジェネレート・オプション］タブを選択します。
標準ライブラリ・ファイルの出力の設定は，［オブジェクト］カテゴリで行います。

図 2.62　　［オブジェクト］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，“%ProjectName%.lib” を設定しています。

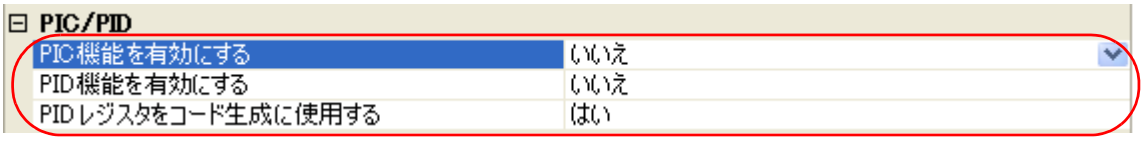
2.10 PIC/PID 機能を使用するための準備をする

PIC/PID 機能では，ROM 上のコードやデータを PIC や PID にしたプログラムをアプリケーション，アプリケーションを実行させるのに必要なプログラムをマスタと呼びます。
マスタやアプリケーションをビルドするときは，構成するオブジェクト間で PIC/PID 機能に関係するビルド・オプションの指定を合わせておく必要があります。
アプリケーションおよびマスタのビルド・オプションの設定方法を以下に示します。

備考　PIC/PID 機能の内容，オプションの組み合わせ，マスタやアプリケーションのスタートアップの作成方法については，「RX コーディング編 第 7 章 スタートアップ」を参照してください。

(1) ビルド・オプションの設定
PIC/PID 機能のビルド・オプションの設定は，プロジェクト・ツリーで，マスタまたはアプリケーションのビルド・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］タブの［PIC/PID］カテゴリで行います。

図 2.63　［PIC/PID］カテゴリ



(a) マスタのビルド・オプションの設定
［PIC 機能を有効にする］プロパティで［いいえ］を選択します（デフォルト）。
［PID 機能を有効にする］プロパティで［いいえ］を選択します（デフォルト）。
［PID レジスタをコード生成に使用する］プロパティで［はい］を選択します（デフォルト）。

(b) アプリケーションのビルド・オプションの設定
［PIC 機能を有効にする］プロパティで［はい（-pic）］を選択します。
［PID 機能を有効にする］プロパティで［はい（オフセットの最大幅：16 ビット）（-pid=16）］，または［はい（オフセットの最大幅：制限なし）（-pid=32）］を選択します。

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。
コンポーネント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（アプリケーション・プロジェクト）→［リンク・オプション］タブ
リンク・フェーズ（ライブラリ・プロジェクト）→［ライブラリアン・オプション］タブ
ライブラリ・ジェネレート・フェーズ→［ライブラリ・ジェネレート・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

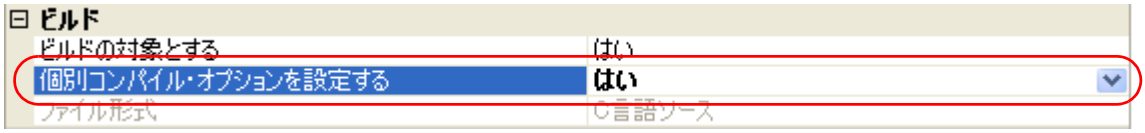
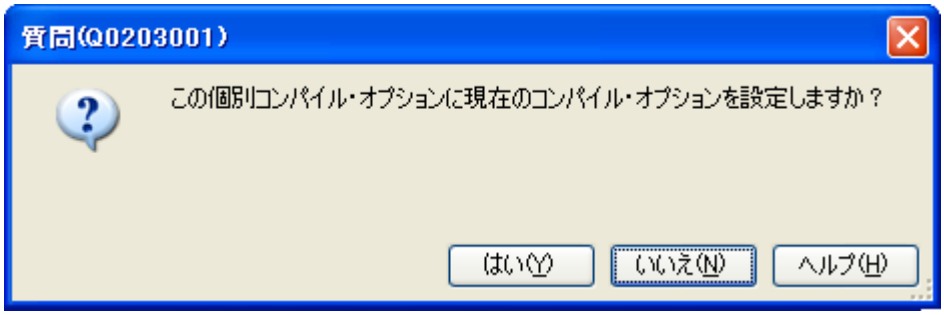
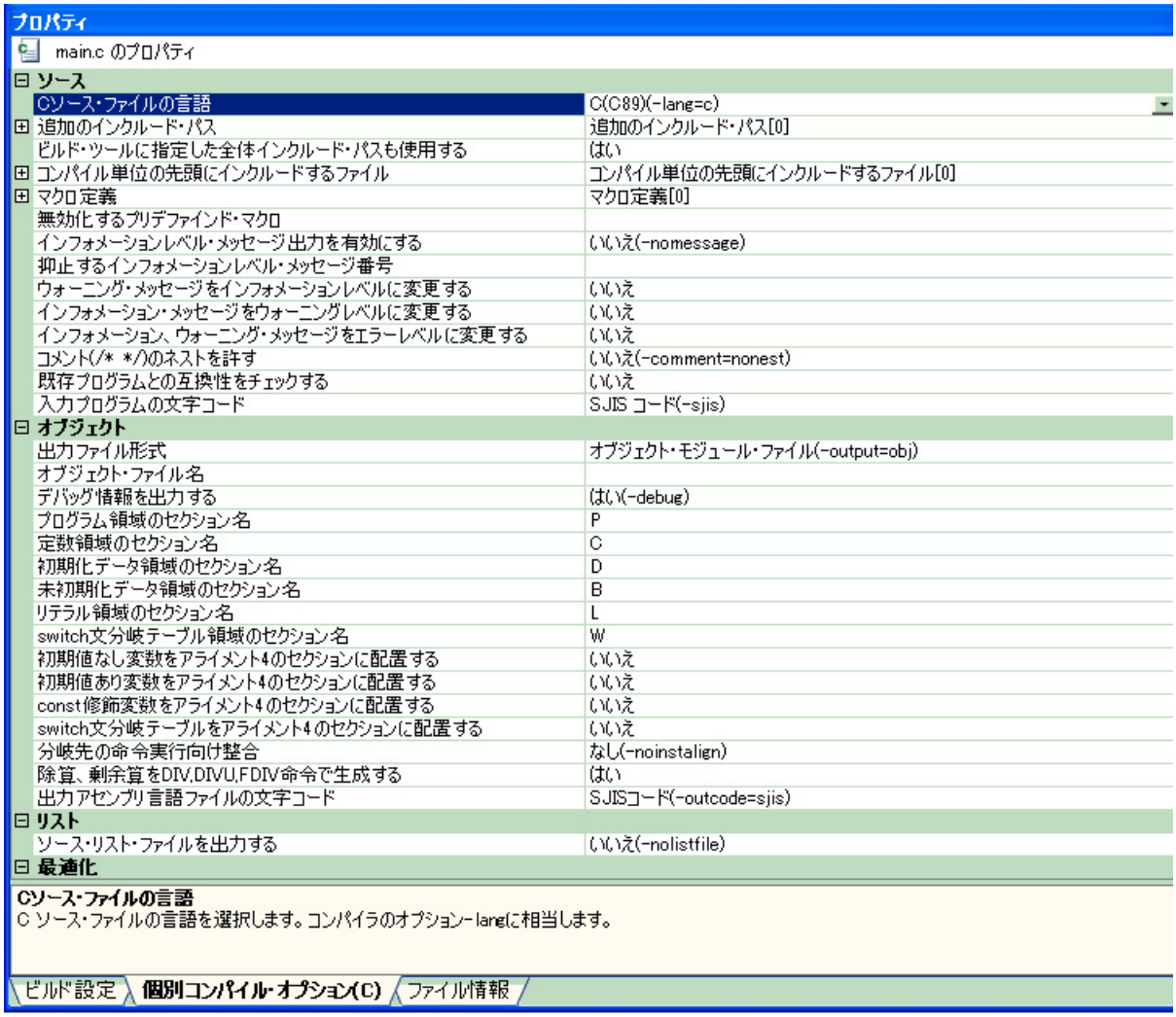


図 2.65 メッセージ ダイアログ



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 (C)］タブが表示されます。

図 2.66　プロパティ パネル：［個別コンパイル・オプション (C)］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，デフォルトでは［コンパイル・オプション］タブの設定内容を継承します。

(2)　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 2.67　［個別コンパイル・オプションを設定する］プロパティ

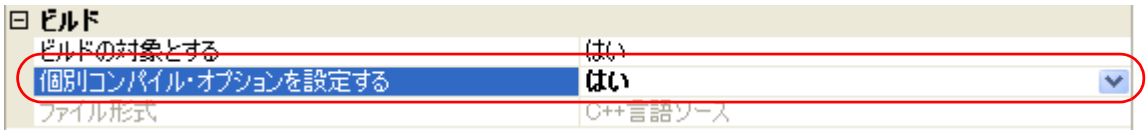
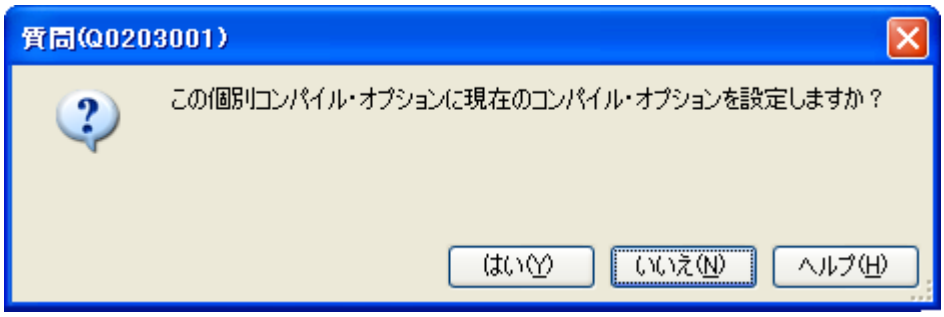
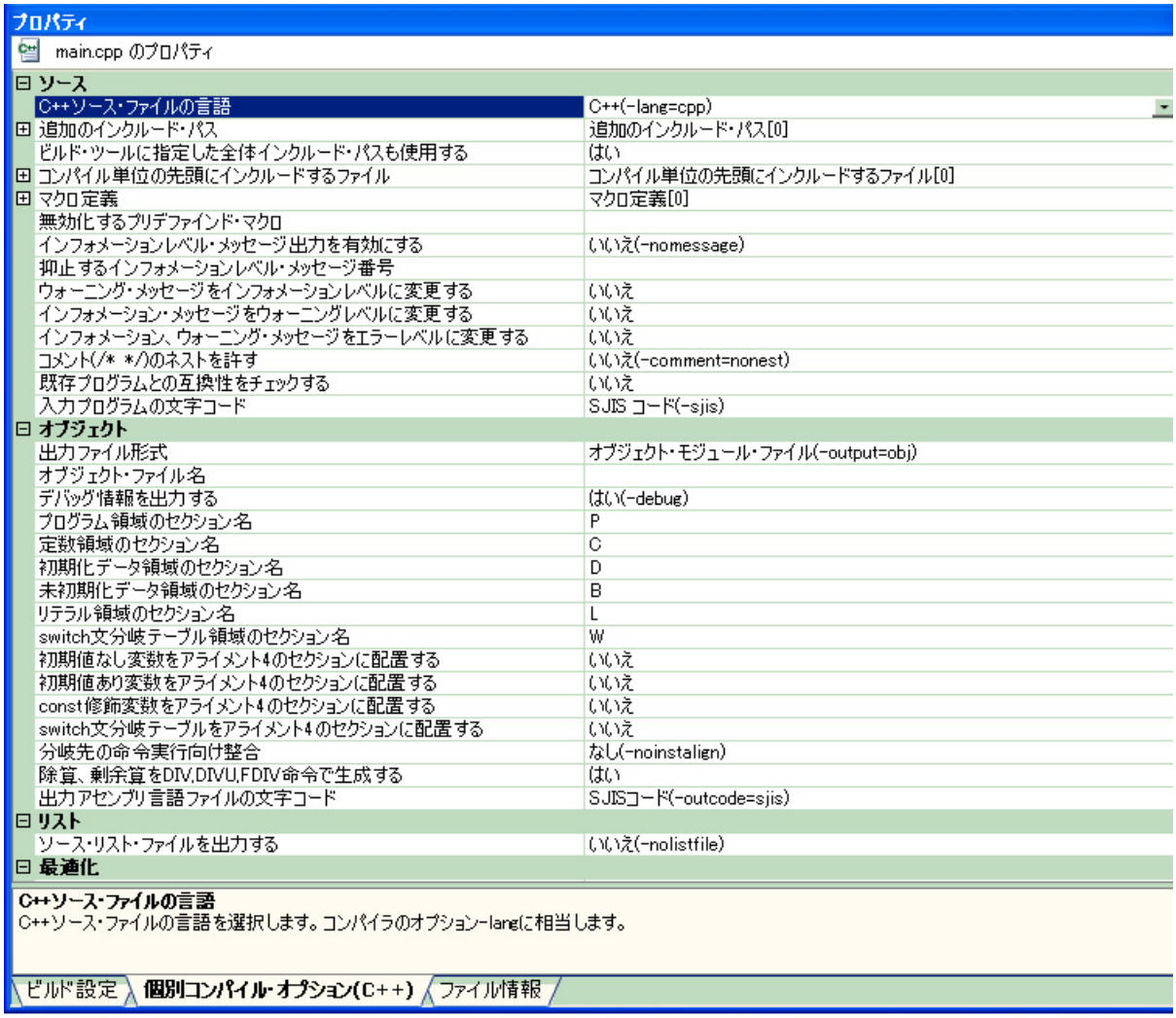


図 2.68　メッセージ ダイアログ



ダイアログ上で［はい］をクリックすると，［個別コンパイル・オプション (C++)］タブが表示されます。

図 2.69　プロパティ パネル：［個別コンパイル・オプション (C++)］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，デフォルトでは［コンパイル・オプション］タブの設定内容を継承します。

(3)　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 2.71 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 2.70　［個別アセンブル・オプションを設定する］プロパティ

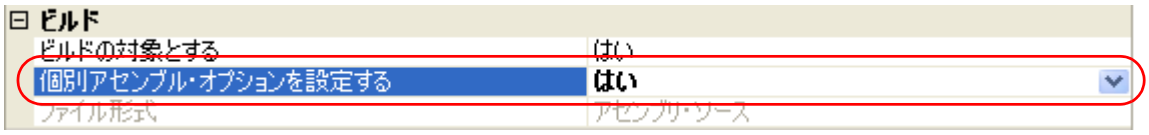
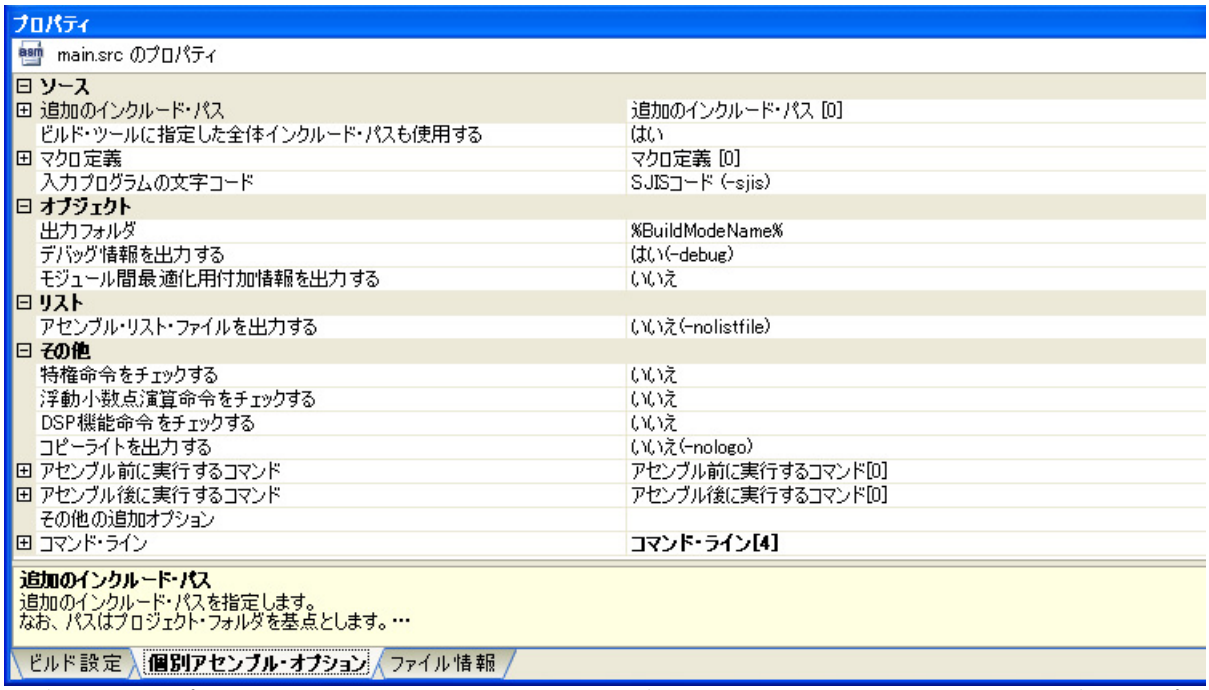


図 2.71　メッセージ ダイアログ

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］タブが表示されます。

図 2.72　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，デフォルトでは［アセンブル・オプション］タブの設定内容を継承します。

2.12 ビルドの設定をする

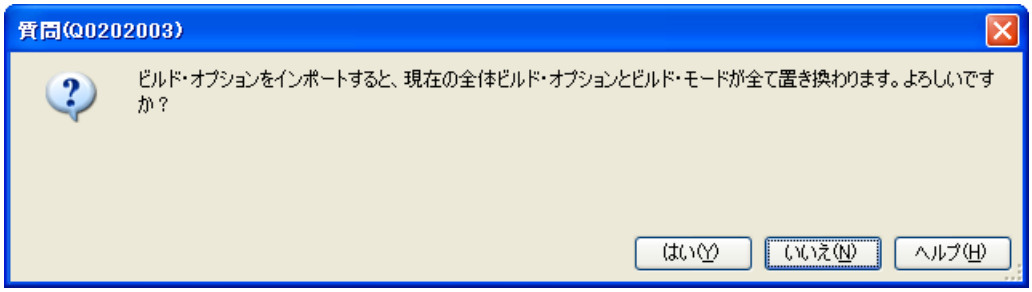
ここでは，ビルドに関する以下の操作を説明します。

- 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

2.12.1 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

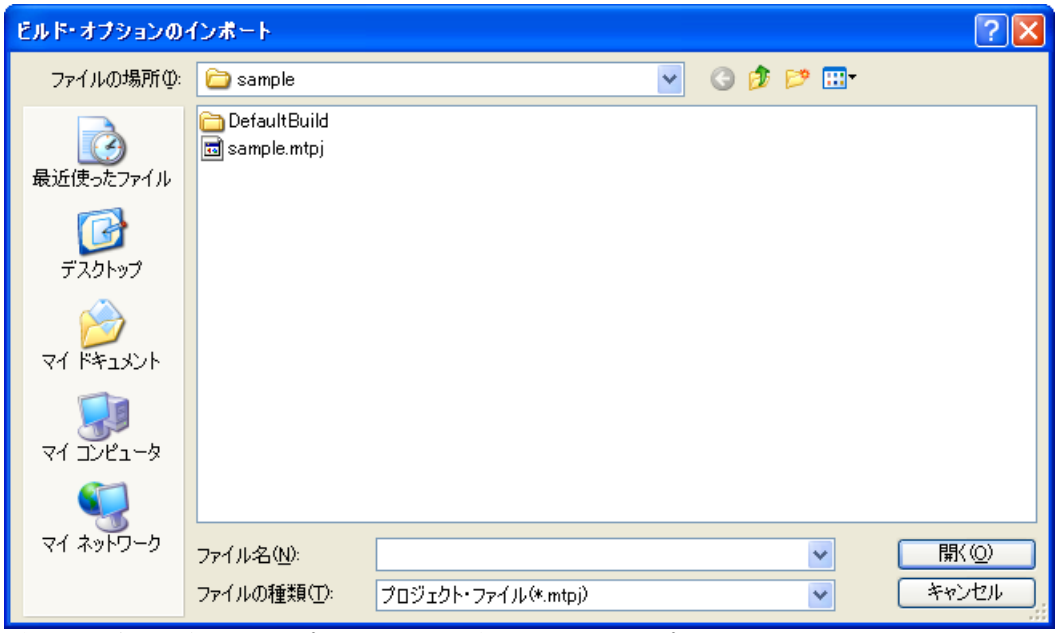
他のプロジェクトのビルド・オプションを現在のプロジェクトにインポートすることができます。
プロジェクト・ツリーでビルド・ツール・ノードを選択し，コンテキスト・メニューの［ビルド・オプションのインポート ...］を選択すると，以下のメッセージ ダイアログがオープンします。

図 2.73　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 2.74　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。

備考 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（「2.12.9 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。

備考 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。

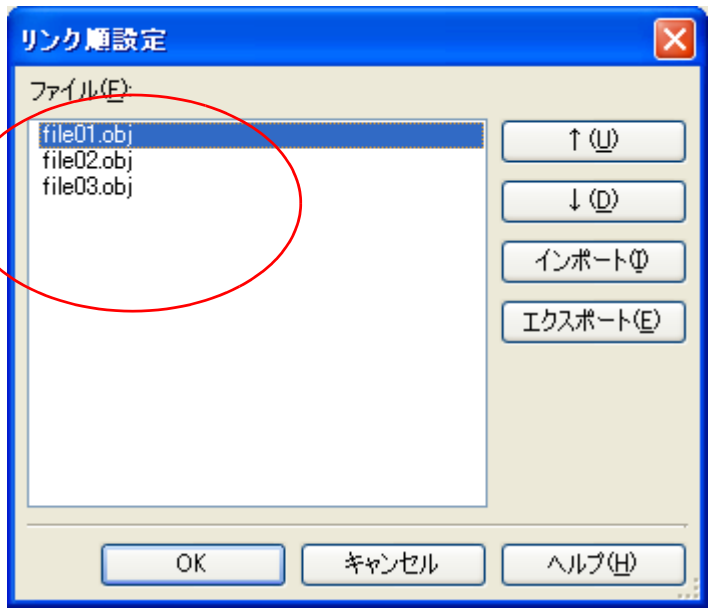
備考 4.　使用するビルド・ツールのバージョンのインポートも行います。

2.12.2 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・ファイルのリンク順は，自動で決定されますが，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 2.75　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が，リンカへの入力順で表示されます。

- 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されているソース・ファイルから生成されるオブジェクト・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

備考　デフォルトでは，プロジェクトに追加されている順番となります。
新規に追加したソース・ファイルから生成されるオブジェクト・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されます。

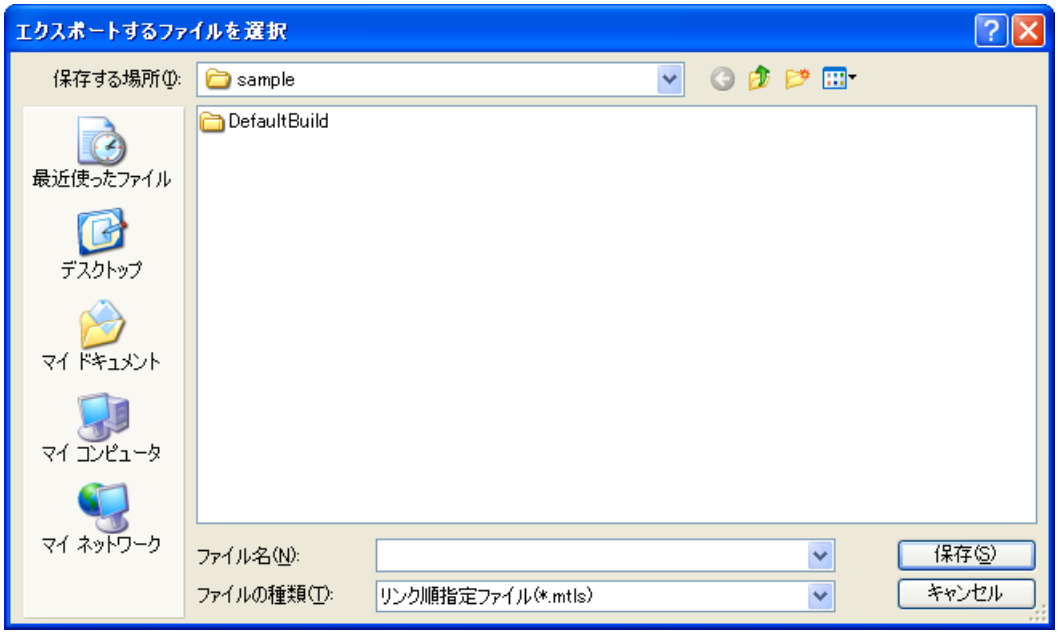
(2) ファイルの表示順の変更
ファイルの表示順を変更することにより，リンカへのファイルの入力順を設定することができます。
以下のいずれかの方法により，ファイルの表示順を変更します。

- [↑]，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 2.76　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
file02.obj
    :
```

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。

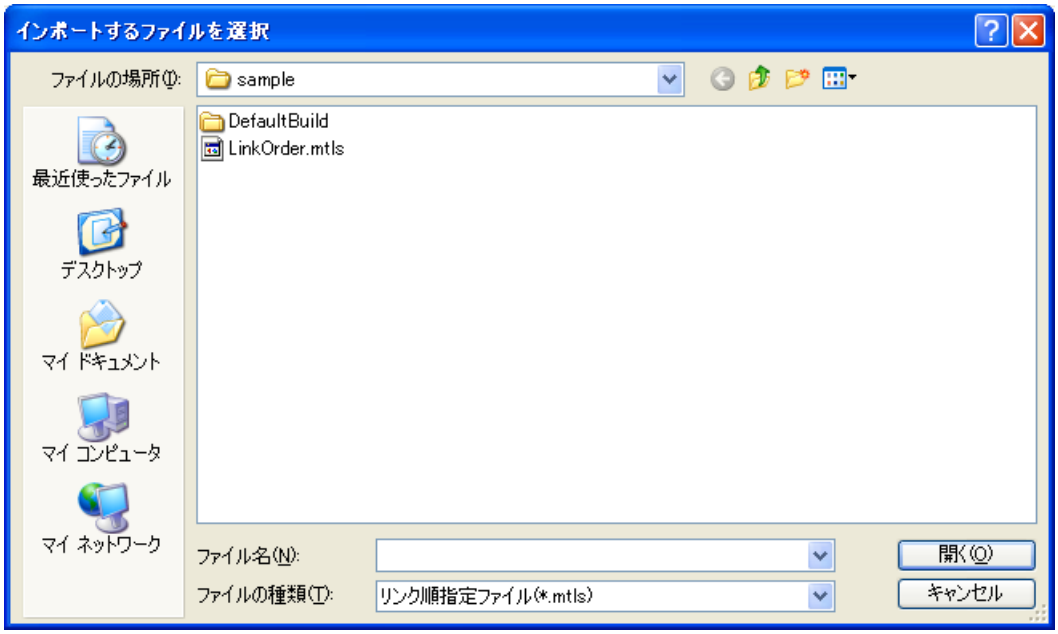
- ファイル名は 1 行に 1 つずつ指定してください。
- ファイル名の大文字／小文字は区別しません。

- “#” 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

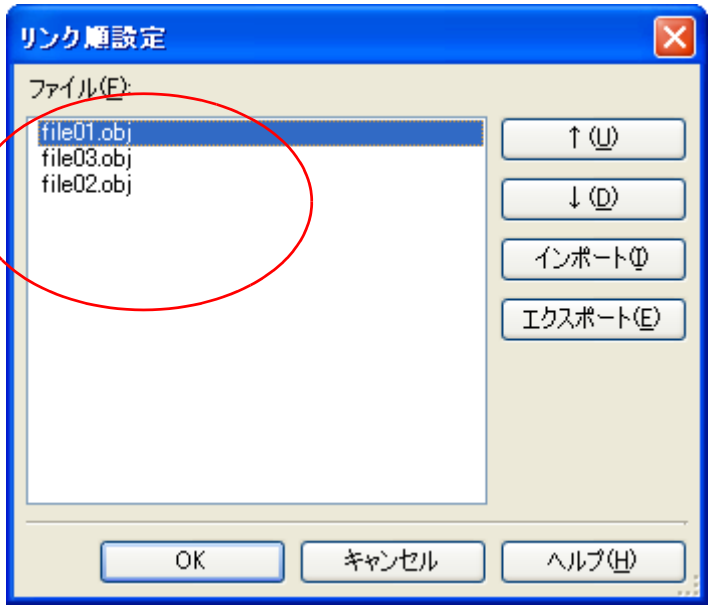
リンク順設定 ダイアログの［インポート］ボタンをクリックすると，インポートするファイルを選択 ダイアログがオープンします。

図 2.77 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 2.78 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。

注意 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。

注意 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。

2.12.3 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。

2.12.4 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が，出力 パネルの［ビルド・ツール］タブにビルド順に表示されます。

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 2.79 ［ビルド・オプション一覧表示フォーマット］プロパティ

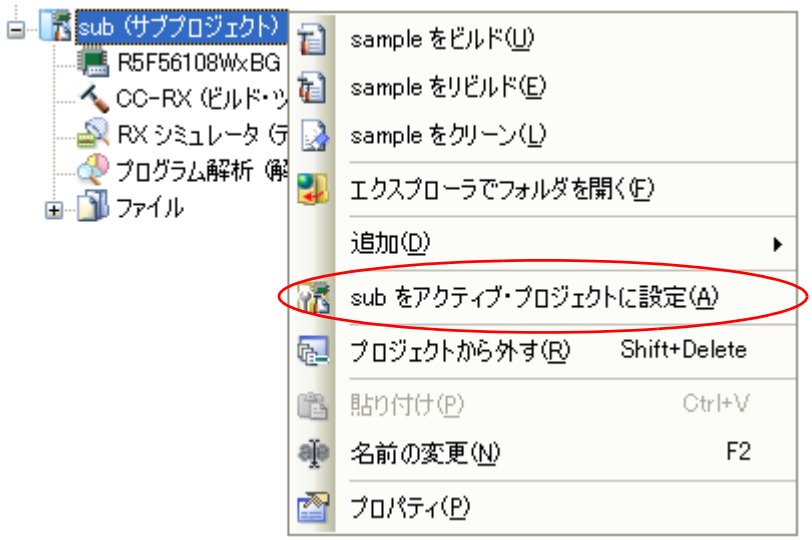
デフォルトでは，“%TargetFiles% : %Program% %Options%” が設定されています。

備考 “%TargetFiles%”，“%Program%”，“%Options%” は，プレースホルダで，それぞれビルド中のファイル名，実行中のプログラム名，ビルド時のコマンド・ライン・オプションに置換します。

2.12.5 ビルド対象プロジェクトを変更する

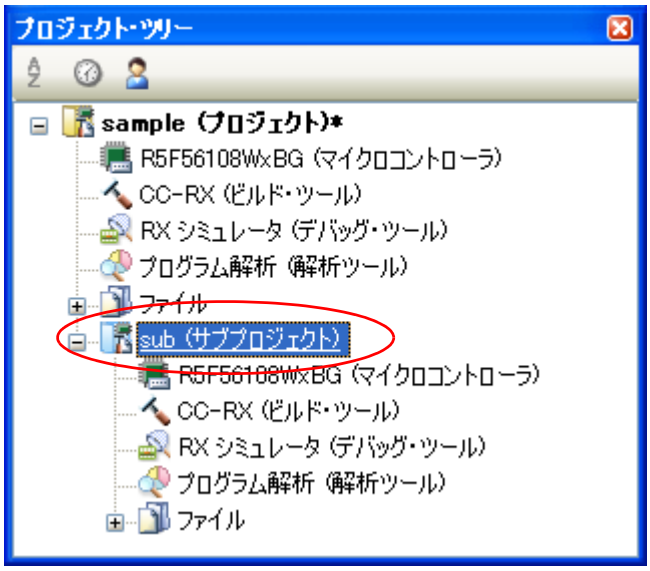
特定のプロジェクト（メイン・プロジェクト，またはサブプロジェクト）を対象にビルドを行う場合，そのプロジェクトを“アクティブ・プロジェクト”として設定する必要があります。
アクティブ・プロジェクトを設定するには，プロジェクト・ツリーでアクティブ・プロジェクトに設定するメイン・プロジェクト・ノード，またはサブプロジェクト・ノードを選択し，コンテキスト・メニューの［*選択しているプロジェクト*をアクティブ・プロジェクトに設定］を選択してください。

図 2.80　［*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が付加されます。

図 2.81　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。

備考 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。

2.12.6 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

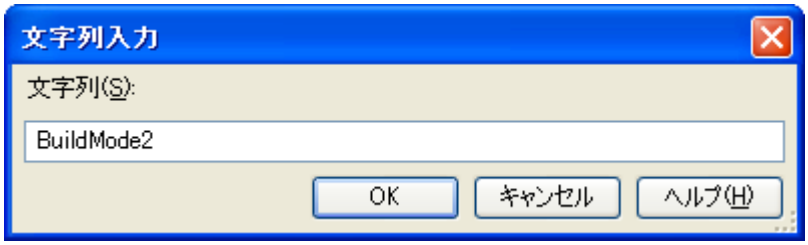
新規のビルド・モードの作成は，既存のビルド・モードの複製により行います。

［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 2.82　　ビルド・モード設定 ダイアログ

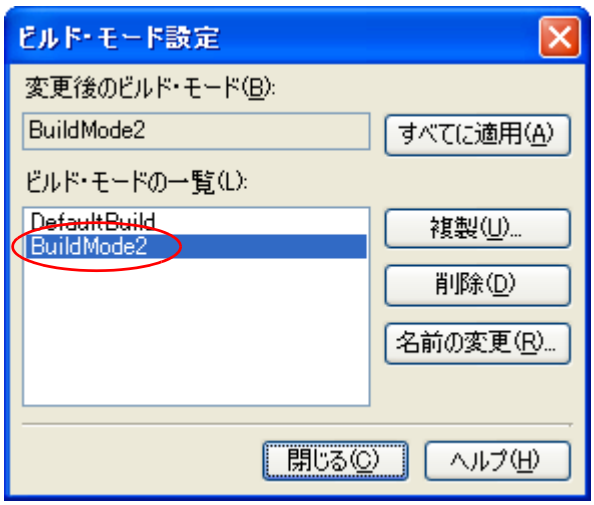
ビルド・モードの一覧から複製元のビルド・モードを選択したのち，［複製 ...］ボタンをクリックすると，文字列入力 ダイアログがオープンします。

図 2.83　　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・モードが追加されます。

図 2.84　　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（「2.12.7 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・オプションやマクロ定義の設定を変更します。

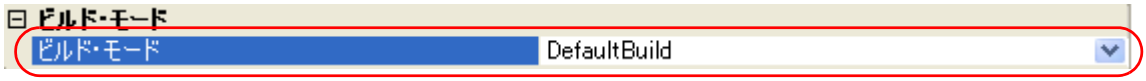
備考　　ビルド・モードの作成は，プロジェクトの変更とみなされます。プロジェクトを閉じる際に，ビルド・モードを保存するかどうかの確認を行います。

2.12.7 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。

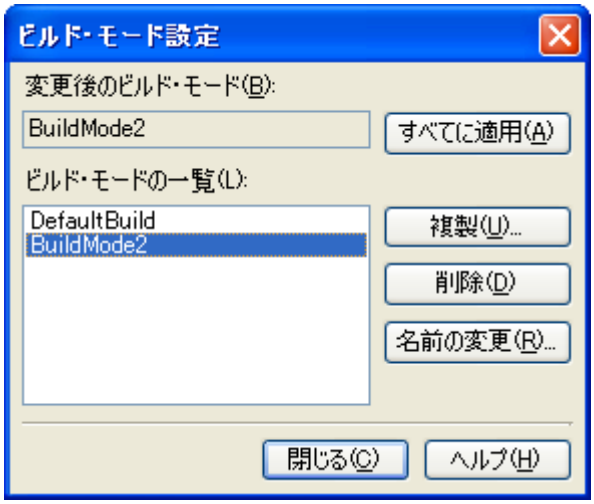
(1) メイン・プロジェクト，またはサブプロジェクトのビルド・モードを変更する場合
プロジェクト・ツリー で対象プロジェクトのビルド・ツール・ノードを選択したのち，プロパティ パネルの［共通オプション］タブを選択します。［ビルド・モード］カテゴリの［ビルド・モード］プロパティで変更するビルド・モードを選択してください。

図 2.85　［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 2.86　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・モードが表示されます。［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。

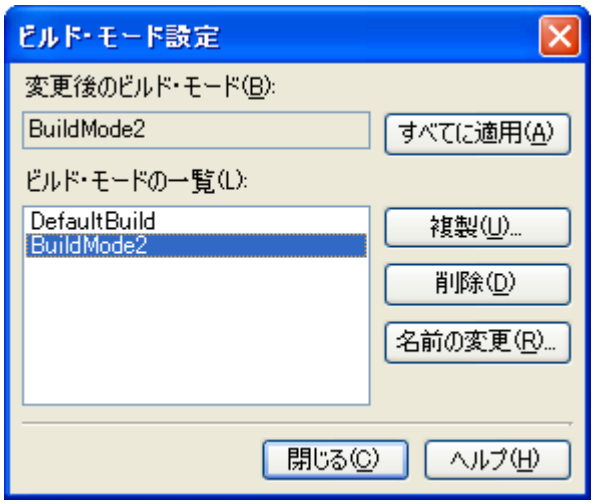
備考 1.　ビルド・モードは，デフォルトでは “DefaultBuild” のみ用意されています。ビルド・モードの追加方法については，「2.12.6 ビルド・モードを追加する」を参照してください。

備考 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。

2.12.8 ビルド・モードを削除する

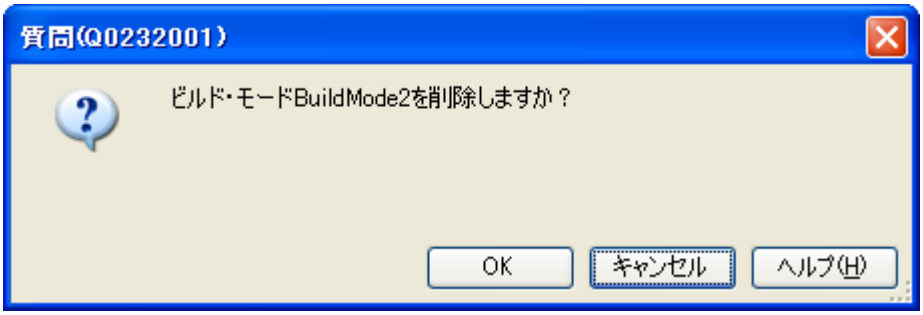
ビルド・モードの削除は，ビルド・モード設定 ダイアログで行います。
［ビルド］メニュー→［ビルド・モードの設定 ...］を選択すると，ダイアログがオープンします。

図 2.87　　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 2.88　　メッセージ ダイアログ



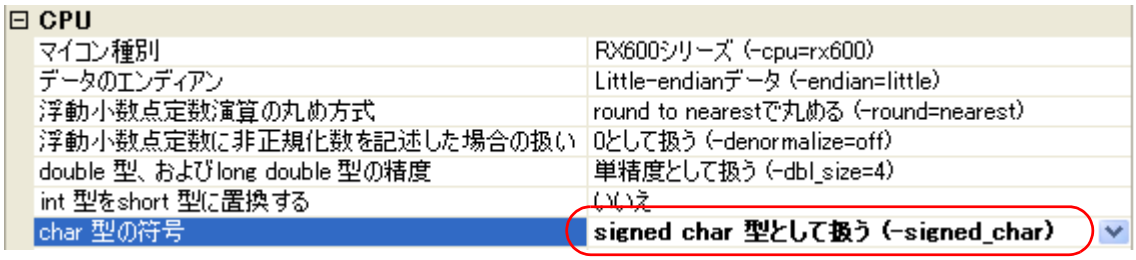
処理を継続するには，ダイアログ上で［OK］をクリックしてください。
選択したビルド・モードをプロジェクトから削除します。

注意　　“DefaultBuild” を削除することはできません。

2.12.9 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

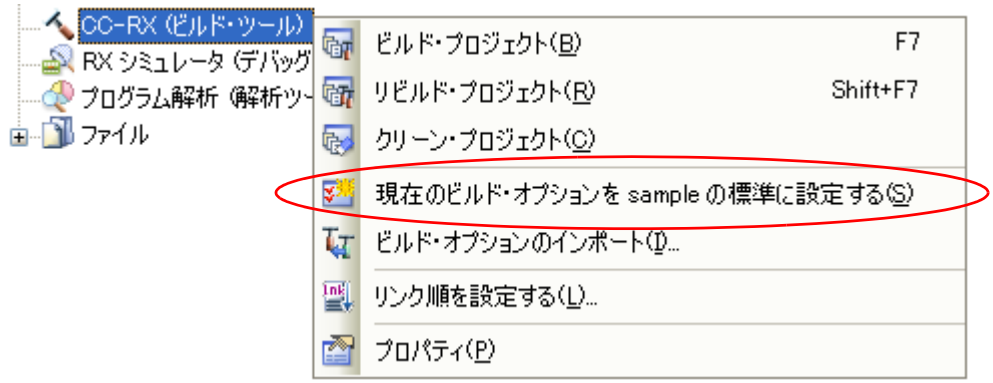
プロパティ パネルにおいて，標準ビルド・オプションの設定に変更を加えると，プロパティの値が太字表示されます。

図 2.89　　プロパティ パネル（標準ビルド・オプション変更後）



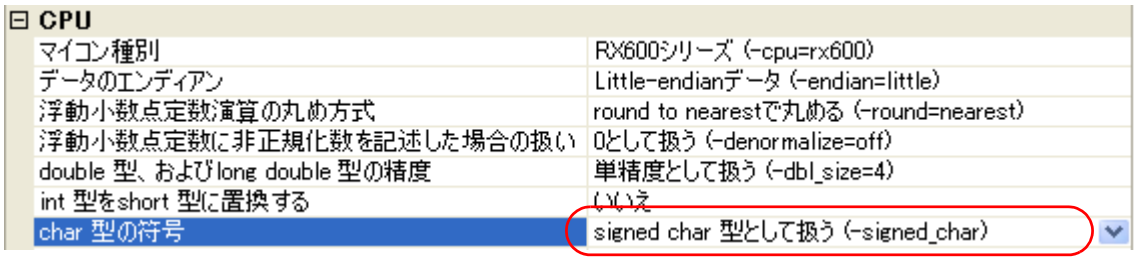
現在選択しているプロジェクト（メイン・プロジェクト，またはサブプロジェクト）のビルド・オプションを標準ビルド・オプションとするには，プロジェクト・ツリーでビルド・ツール・ノードを選択し，コンテキスト・メニューの［現在のビルド・オプションを*選択しているプロジェクト*の標準に設定する］を選択してください。

図 2.90　［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 2.91　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。

2.13 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，更新されたファイルのみビルドを実行します。 →「2.13.1 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。 →「2.13.2 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。 →「2.13.3 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。 →「2.13.4 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（「2.12.3 サブプロジェクトのビルド順を変更する」参照）。

備考 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されます。

- 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 2.92　ビルドの実行結果（ビルド，リビルド，バッチ・ビルドの場合）

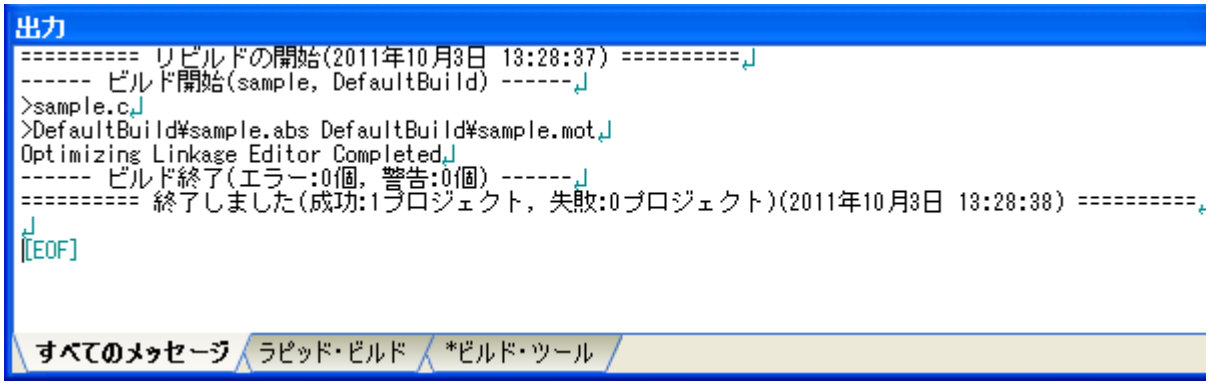
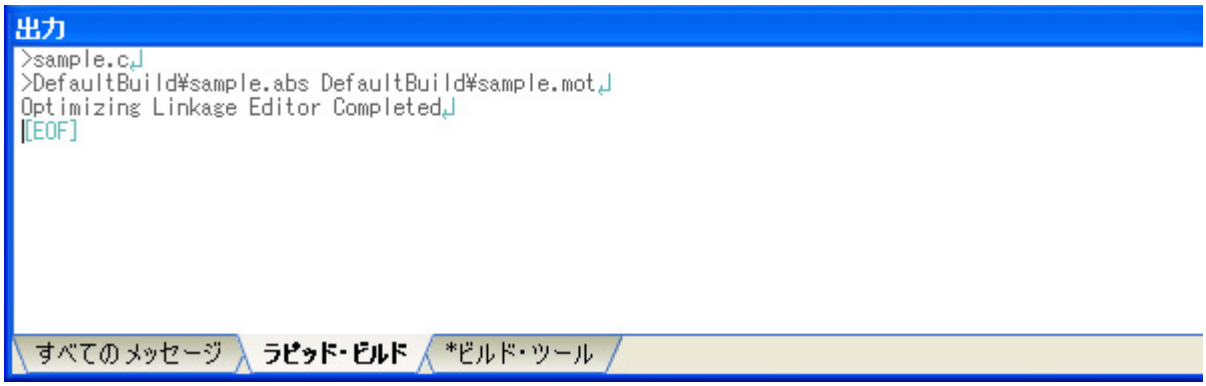


図 2.93　ビルドの実行結果（ラピッド・ビルドの場合）



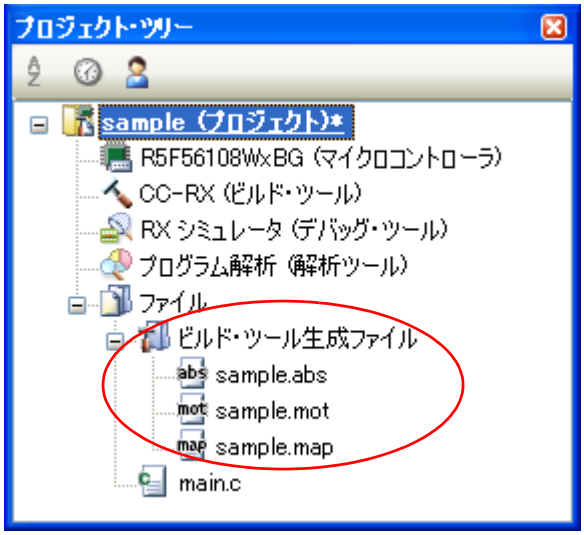
備考 1.　［ラピッド・ビルド］タブの表示文字列は，淡色表示になります。

備考 2.　出力され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。

備考 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・ノードに表示されます。

図 2.94　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・ノードに表示されるのは，以下のファイルです。

- アプリケーション用のプロジェクトの場合
  ロード・モジュール・ファイル（*.abs）
  ヘキサ・ファイル（*.hex）
  S レコード・ファイル（*.mot）
  バイナリ・データ・ファイル（*.bin）
  マップ・ファイル（*.map）

- 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)
  リロケータブル・モジュール・ファイル（*.rel）
  マップ・ファイル (*.map，*.lbp)

注意　　ビルド・ツール生成ファイル・ノードは，ビルド時に作成されるノードです。
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。

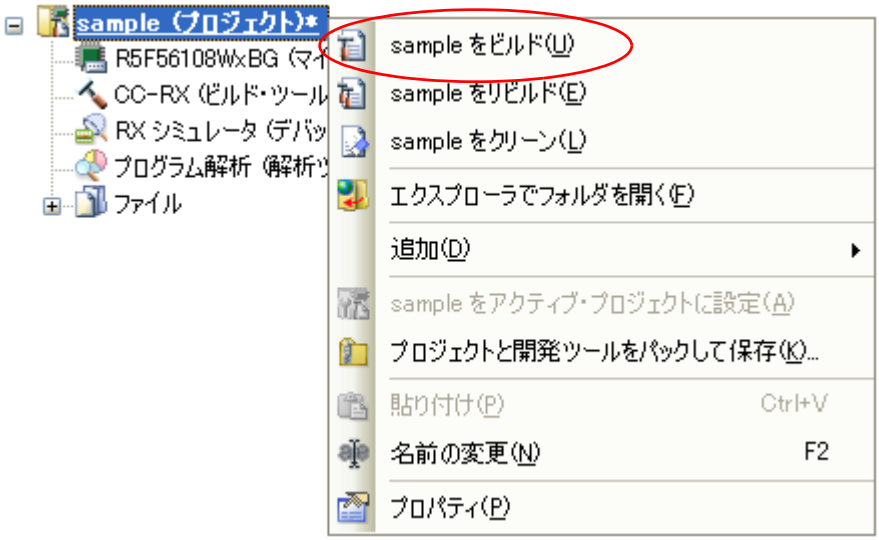
2.13.1 更新ファイルのビルドを実行する

ビルド対象ファイルのうち，更新され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（「2.12.5 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 2.95　　［*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（「2.3.6 ファイルの依存関係を更新する」参照）。

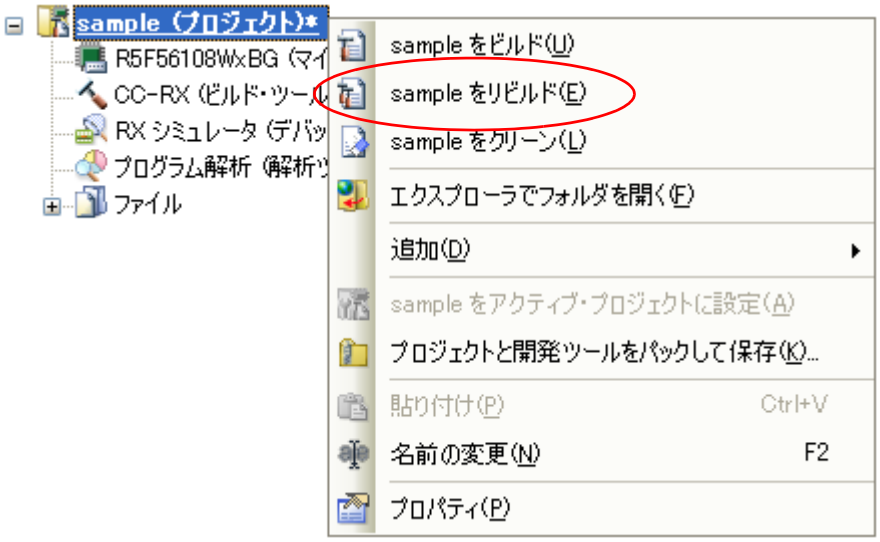
2.13.2 すべてのファイルのビルドを実行する

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）。
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（「2.12.5 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 2.96 ［*アクティブ・プロジェクト*をリビルド］項目



2.13.3 他の処理と平行してビルドを実行する

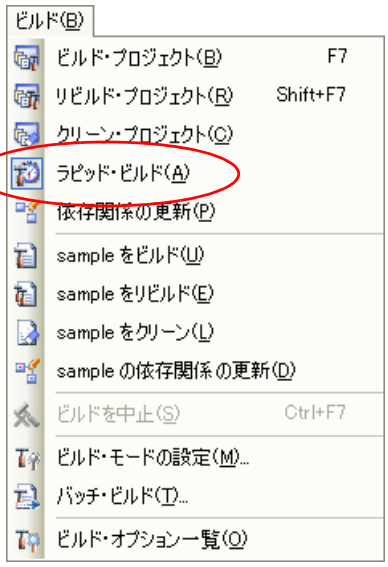
以下のタイミングでビルドを自動で開始する機能があります（以降，“ラピッド・ビルド”と呼びます）。

- プロジェクトに追加している C ソース・ファイル，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，オブジェクト・モジュール・ファイル，リロケータブル・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

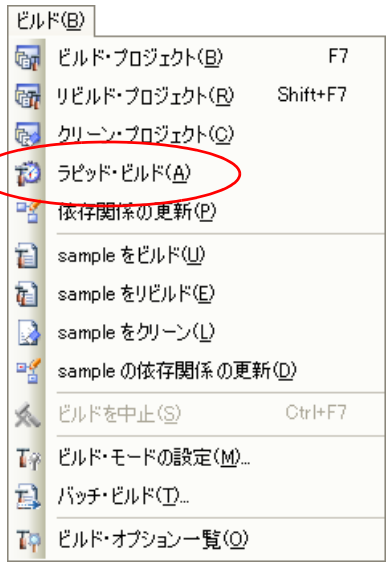
ラピッド・ビルドを有効にすることにより，上記の操作と平行してビルドを行うことができます。
ラピッド・ビルドの有効／無効は，［ビルド］メニュー→［ラピッド・ビルド］の選択により，切り替えます。デフォルトでは，有効となっています。

図 2.97 ［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・ファイルは、編集後の都度［Ctrl］＋［S］キーの押下により，定期的に上書き保存することを推奨します。

備考 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。

備考 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。

2.13.4 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→「2.13.1 更新ファイルのビルドを実行する」参照
- リビルド→「2.13.2 すべてのファイルのビルドを実行する」参照
- クリーン→「2.13.8 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 2.98 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・モードの組み合わせの一覧が表示されます。
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。

注意 本ビルド・ツールでは，［定義マクロ］の表示をサポートしていません。

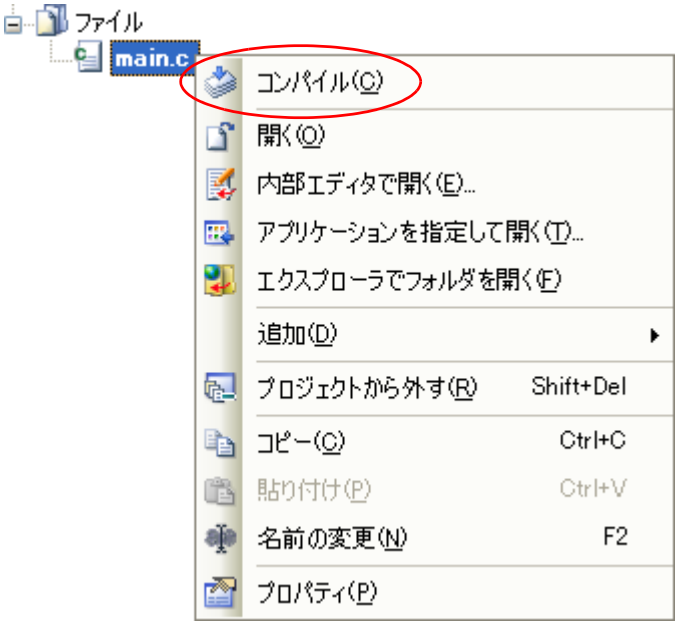
備考 バッチ・ビルド順は，プロジェクトのビルド順に従い，サブプロジェクト，メイン・プロジェクトの順となります。
1 つのメイン・プロジェクト，またはサブプロジェクトについて複数のビルド・モードを選択した場合は，サブプロジェクトで選択されているすべてのビルド・モードでビルドを行ったのち，次のサブプロジェクト，またはメイン・プロジェクトのビルドを行います。

2.13.5 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。

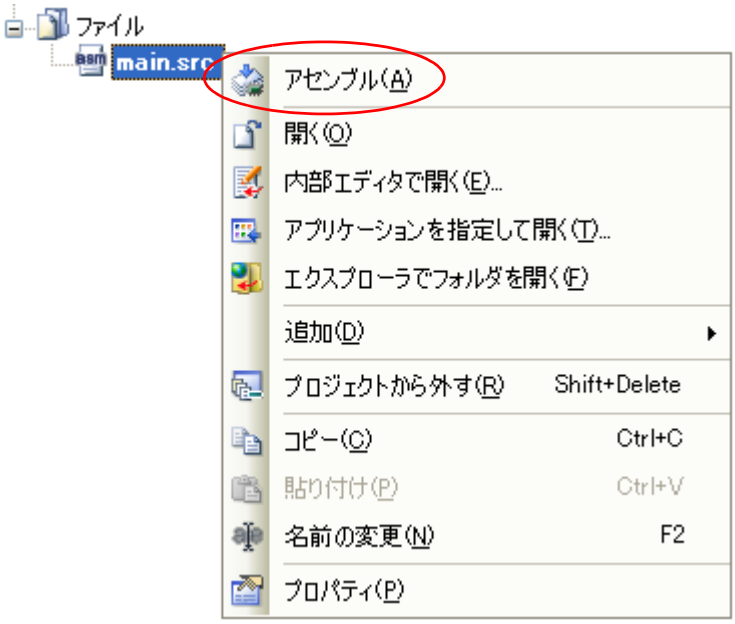
(1) C ソース・ファイル，または C++ ソース・ファイルをコンパイルする場合
プロジェクト・ツリー で C ソース・ファイルまたは C++ ソース・ファイルを選択し，コンテキスト・メニューの［コンパイル］を選択してください。

図 2.99 ［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 2.100 ［アセンブル］項目



2.13.6 ビルドの実行を中止する

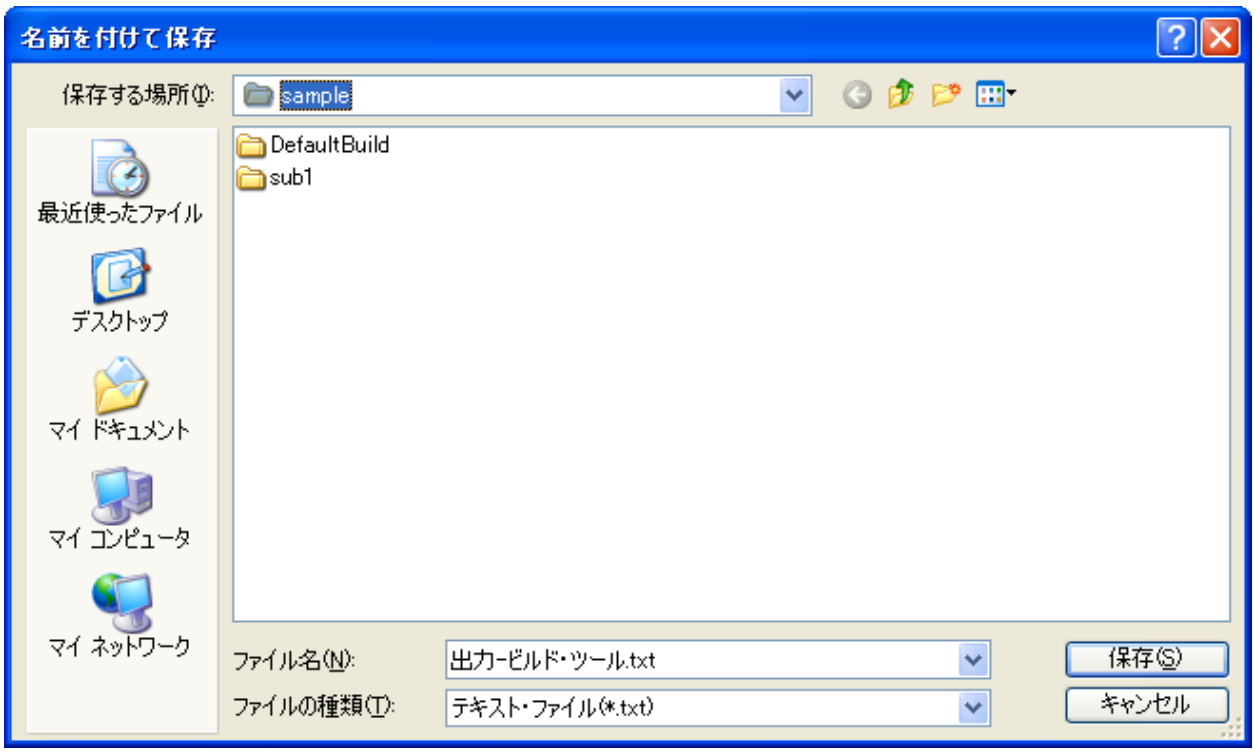
実行中のビルド，リビルド，バッチ・ビルドを中止するには，ツールバーの  ボタンをクリックしてください。

2.13.7 ビルド結果をファイルに保存する

出力 パネルに表示され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 2.101　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。

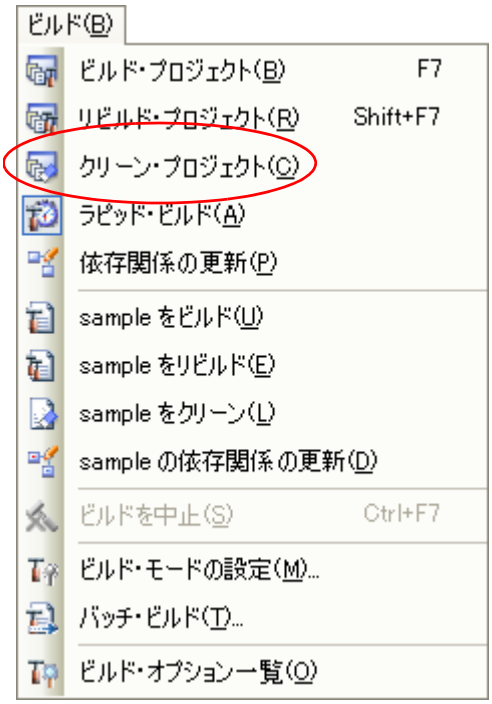
2.13.8 中間ファイル，生成ファイルを削除する

ビルドの実行により出力された中間ファイル，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（「2.12.5 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。

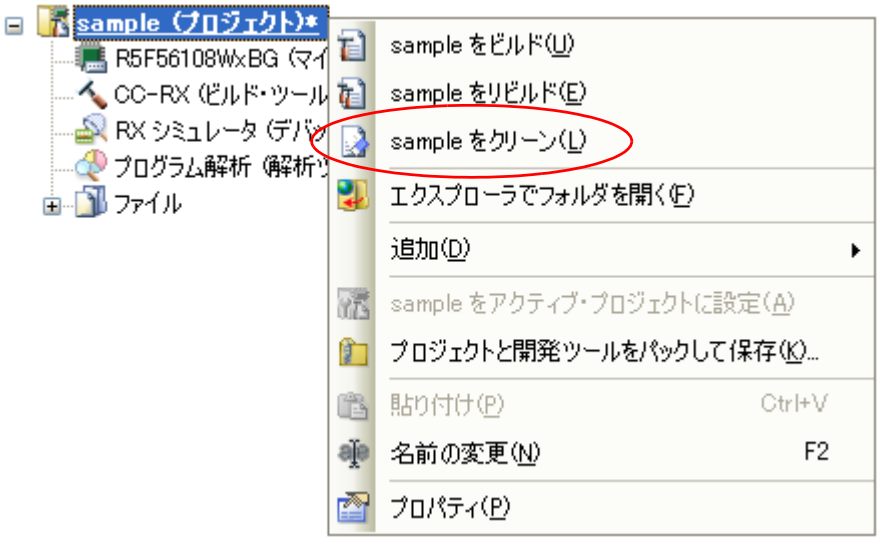
(1) プロジェクト全体のクリーンを実行する場合
［ビルド］メニュー→［クリーン・プロジェクト］を選択してください。

図 2.102 ［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 2.103 ［*アクティブ・プロジェクト*をクリーン］項目



2.14 スタックの使用量を見積もる

スタックの使用量を見積もるには，Call Walker を使用します。

Call Walker では，アブソリュートモジュールファイルの内容を静的に解析することにより，シンボルの呼び出し関係をツリー形式で表示すると共に，シンボル単位のスタック情報（シンボル名，属性，アドレス，サイズ，スタック・サイズ，ファイル名）をリスト形式で表示します。

Call Walker の起動は，Windows® の［スタート］→［すべてのプログラム］→［Renesas Electronics CubeSuite+］→［ユーティリティ］→［Call Walker］を選択することにより行います。

また，Call Walker の終了は，Call Walker の［File］メニュー→［Exit］を選択することにより行います。

操作手順については，CallWalker の［Help］メニュー→［Help Topics］を参照してください。

3. ビルドの出力リスト

この章では，ビルドにより各コマンドが出力する各種リスト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

アセンブラが出力するアセンブル・リスト・ファイルの内容と形式について説明します。
ソースリストファイルには，コンパイル結果およびアセンブル結果の情報を表示します。
ソースリストの構成と内容を示します。

表 3.1 ソースリストの構成と内容

No	リストファイルへ表示する情報	内容	サブオプション注	-show オプション省略時
1	ソース情報	アセンブリソースに対応して，C/C++ 言語ソースを表示	-show=source	出力しない
2	オブジェクト情報	オブジェクトプログラムの機械語，アセンブリソースコード	なし	出力する
3	統計情報	エラーの総数，ソースプログラムの行数，セクションサイズ	なし	出力する
4	コマンド指定情報	コマンドで指定されたファイル名とオプションを表示	なし	出力する

注 -listfile オプションを指定した場合に有効です。

3.1.1 ソース情報

ソース情報は，-show=source オプションを指定することでオブジェクト情報に埋め込まれる形で出力されます。出力例は次項を参照ください。

3.1.2 オブジェクト情報

オブジェクト情報の出力例を示します。

```
* RX FAMILY ASSEMBLER V2.00.00  [15 Feb 2013] *  SOURCE LIST  Mon Feb 18 20:15:19 2013
(1)       (2)                 (3)    (4)
LOC.      OBJ.                0XMDA SOURCE STATEMENT

                                      ;RX Family C/C++ Compiler (V2.00.00  [15 Feb 2013])  18-
Feb-2013 20:15:19

                                      ;*** CPU TYPE ***

                                      ;-ISA=RXV1

                                      ;*** COMMAND PARAMETER ***

                                      ;-output=src=sample.src
                                      ;-listfile
                                      ;-show=source
                                      ;sample.c

                                      .glb_x
                                      .glb_y
                                      .glb_func02
```

```
                                .glb_func03
                                .glb_func01
                                (5)       (6)
                                ;LineNo. C-SOURCE STATEMENT

                                .SECTIONP,CODE
00000000                        _func02:
                                .STACK_func02=12
                                ;       1 #include "include.h"
                                ;       2 int func01(int);
                                ;       3 int func03(int);
                                ;       4
                                ;       5 int func02(int z)
00000000 6E67                   PUSHM R6-R7
00000002 EF16                   MOV.L R1, R6
                                ;       6 {
                                ;       7     x = func01(z);
00000004 05rrrrrr             A BSR _func01
00000008 FB72rrrrrrrr           MOV.L #_x, R7
0000000E E371                   MOV.L R1, [R7]
                                ;       8     if (z == 2) {
00000010 6126                   CMP #02H, R6
00000012 18                   S BNE L12
00000013                        L11:; bb3
                                ;       9         x++;
00000013 6211                   ADD #01H, R1
00000015 08                   S BRA L13
00000016                        L12:; bb6
                                ;      10     } else {
                                ;      11         x = func03(x + 2);
00000016 6221                   ADD #02H, R1
00000018 39rrrr               W BSR _func03
0000001B                        L13:; bb13
0000001B E371                   MOV.L R1, [R7]
                                ;      12     }
                                ;      13     return x;
                                ;      14 }
0000001D 3F6702                 RTSD #08H, R6-R7
00000020                        _func03:
                                .STACK_func03=4
                                ;      15
                                ;      16 int func03(int p)
                                ;      17 {
                                ;      18     return p+1;
00000020 6211                   ADD #01H, R1
                                ;      19 }
00000022 02                     RTS
                                .SECTIOND,ROMDATA,ALIGN=4
00000000                        _y:
00000000 01000000               .lword00000001H
                                .END
```

項番	説明
(1)	ロケーション情報（LOC.） アセンブル時に決定できる範囲のオブジェクトコードのロケーションアドレスを出力します
(2)	オブジェクトコード情報（OBJ.） ニーモニックに対応するオブジェクトコードを出力します

<table>
<tr><th>項番</th><th>説明</th></tr>
<tr><td>(3)</td><td>行情報（0XMDA）
アセンブラがソースを処理した結果の情報を出力します。
各記号の意味を下記に示します。
<table>
<tr><th>0</th><th>X</th><th>M</th><th>S</th><th>D</th><th>内　容</th></tr>
<tr><td>0-30</td><td></td><td></td><td></td><td></td><td>インクルードファイルのネストレベルを示します。</td></tr>
<tr><td></td><td>X</td><td></td><td></td><td></td><td>-show=conditions 指定時，条件アセンブルで条件が偽となった行を示します。</td></tr>
<tr><td></td><td></td><td>M</td><td></td><td></td><td>-show=expansions 指定時，マクロ命令の展開行であることを示します。</td></tr>
<tr><td></td><td></td><td>D</td><td></td><td></td><td>-show=definitions 指定時，マクロ命令の定義行であることを示します。</td></tr>
<tr><td></td><td></td><td></td><td>S</td><td></td><td>分岐距離指定子 S を指定したことを示します。</td></tr>
<tr><td></td><td></td><td></td><td>B</td><td></td><td>分岐距離指定子 B を指定したことを示します。</td></tr>
<tr><td></td><td></td><td></td><td>W</td><td></td><td>分岐距離指定子 W を指定したことを示します。</td></tr>
<tr><td></td><td></td><td></td><td>A</td><td></td><td>分岐距離指定子 A を指定したことを示します。</td></tr>
<tr><td></td><td></td><td></td><td></td><td>*</td><td>条件分岐命令に対して代替命令を選択したことを示します。</td></tr>
</table></td></tr>
<tr><td>(4)</td><td>ソース情報（SOURCE STATEMENT）
アセンブリソースファイルの内容を表示します</td></tr>
<tr><td>(5)</td><td>C/C++ ソース行番号（LineNo.）</td></tr>
<tr><td>(6)</td><td>C/C++ ソース（C-SOURCE STATEMENT）
-show=source オプションを指定した場合，C/C++ ソースを出力します</td></tr>
</table>

3.1.3 統計情報

統計情報の出力例を示します。

```
Information List (1)
TOTAL ERROR(S)    00000
TOTAL WARNING(S)  00000
TOTAL LINE(S)     00071  LINES
Section List (2)
Attr          Size                Name
CODE     0000000047(0000002FH)  P
ROMDATA  0000000004(00000004H)  D
```

項番	説明
(1)	エラー，警告それぞれのメッセージ数と，ソース行の総数
(2)	セクション情報（セクション属性，サイズ，セクション名）

3.1.4 コンパイラのコマンド指定情報

コンパイラを起動したときのコマンドで指定されたファイル名とオプションを表示します。
コンパイラのコマンド指定情報は，リストファイルの先頭に出力されます。
コマンド指定情報の出力例を示します。

```
;*** CPU TYPE *** (1)
;-ISA=RXV1
;*** COMMAND PARAMETER *** (2)
;-output=src=C:￥tmp￥elp1894￥sample.src
;-nologo
;-show=source
;sample.c
```

項番	説明
(1)	選択されているマイコン
(2)	コンパイラに渡したファイル名とオプション]

3.1.5 アセンブラのコマンド指定情報

アセンブラを起動したときのコマンドで指定されたファイル名とオプションを表示します。
アセンブラのコマンド指定情報は，リストファイルの最後に出力されます。
コマンド指定情報の出力例を示します。

```
Cpu Type (1)
-ISA=RXV1
Command Parameter (2)
-output=sample.obj
-nologo
-listfile=sample.lst
```

項番	説明
(1)	アセンブラで選択されているマイコン
(2)	アセンブラに渡したファイル名とオプション

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。

3.2.1 リンケージリストの構成

リンケージリストの構成と内容を表 3.3 に示します。

表 3.2 リンケージリストの構成と内容

No	リストファイルへ表示する情報	内容	-show オプション[注]指定	-show オプション省略時
1	オプション情報	コマンドライン，サブコマンドで指定したオプション列を表示	なし	出力する
2	エラー情報	エラーメッセージを表示	なし	出力する
3	リンケージマップ情報	セクション名，先頭／最終アドレス，サイズ，種別を表示	なし	出力する
4	シンボル情報	静的定義シンボル名，アドレス，サイズ，種別をアドレス順に表示 -show=reference を指定した場合は，各シンボルの参照回数，最適化実行有無も表示	-show=symbol -show=reference	出力しない 出力しない
5	シンボル削除最適化情報	最適化で削除したシンボルを表示	-show=symbol	出力しない
6	クロスリファレンス情報	シンボルの参照情報を表示	-show=xreference	出力しない
7	合計セクションサイズ	RAM,ROM, およびプログラムセクションの合計サイズを表示	-show=total_size	出力しない
8	ベクタ情報	ベクタ番号とアドレスの情報を表示	-show=vector	出力しない
9	CRC 情報	CRC の演算結果および出力位置アドレスを表示	なし	CRC オプション指定時は常に出力

注 -show オプションは list オプションを指定した場合に有効です。

3.2.2 オプション情報

コマンドライン，サブコマンドファイルで指定したオプション列を出力します。
オプション情報の出力例を示します（rlink -subcommand=test.sub -list -show 指定時）。

```
（test.sub の内容）
INPUT  test.obj
```

```
*** Options ***
-sub=test.sub
INPUT test.obj
-list
-show
```

(1)
(2)

項番	説明
(1)	コマンドライン，サブコマンドで指定したオプション列を，指定順に出力します。
(2)	サブコマンドファイル test.sub 内のサブコマンドです。

3.2.3 エラー情報

エラーメッセージを出力します。エラー情報の出力例を示します。

```
*** Error Information ***
** E0562310 Undefined external symbol "strcmp" referred to in "test.obj" (1)
```

項番	説明
(1)	エラーメッセージを出力します

3.2.4 リンケージマップ情報

各セクションの先頭／最終アドレス，サイズ，種別をアドレス順に出力します。
リンケージマップ情報の出力例を示します。

```
*** Mapping List ***
(1)                                (2)        (3)          (4)    (5)
SECTION                             START      END          SIZE   ALIGN
P
                                   00001000  00001000          1   1
C
                                   00001004  00001007          4   4
D_2
                                   00001008  000014dd        4d6   2
B_2
                                   000014de  000050b3       3bd6   2
```

項番	説明
(1)	セクション名を表示します
(2)	先頭アドレスを表示します
(3)	最終アドレスを表示します
(4)	セクションサイズを表示します
(5)	セクションのアライメント数を表示します

3.2.5 シンボル情報

-show=symbol を指定した場合，外部定義シンボルまたは静的内部定義シンボルのアドレス，サイズ，種別をアドレス順に出力します。また，-show=reference を指定した場合は，各シンボルの参照回数，最適化実行の有無も出力します。シンボル情報の出力例を図 3.8 に示します。

```
*** Symbol List ***
SECTION= (1)
  (2)          (3)          (4)               (5)
FILE=         START         END                SIZE
   (6)           (7)          (8)               (9)            (10)     (11)
   SYMBOL        ADDR         SIZE               INFO          COUNTS   OPT
SECTION=P
FILE=test.obj
                 00000000     00000428              428
   _main
                 00000000            2         func ,g             0
   _malloc
                 00000000           32         func ,l             0
FILE=mvn3
                 00000428     00000490               68
   $MVN#3
                 00000428            0         none ,g             0
```

項番	説明
(1)	セクション名を表示します。
(2)	ファイル名を表示します
(3)	(2) のファイルに含まれる該当セクションの先頭アドレスを表示します
(4)	(2) のファイルに含まれる該当セクションの最終アドレスを表示します
(5)	(2) のファイルに含まれる該当セクションのセクションサイズを表示します
(6)	シンボル名を表示します
(7)	シンボルアドレスを表示します
(8)	シンボルサイズを表示します
(9)	シンボル種別を次のように表示します データ種別 func: 関数名 data: 変数名 entry: エントリ関数名 none: 未設定（ラベル，アセンブラシンボル） 宣言種別 g: 外部定義 l: 内部定義
(10)	シンボル参照回数を表示します。 -show=reference を指定した場合のみ表示します。 参照回数を表示しないときは，* を表示します
(11)	最適化有無を次のように表示します ch: 最適化によって変更されたシンボル cr: 最適化によって生成されたシンボル mv: 最適化によって移動されたシンボル

3.2.6 シンボル削除最適化情報

シンボル削除最適化（-optimize=symbol_delete）によって削除されたシンボルのサイズ，種別を出力します。
シンボル削除最適化情報の出力例を示します。

```
*** Delete Symbols ***
(1)                              (2)       (3)
SYMBOL                            SIZE      INFO
  _Version
                                     4      data ,g
```

項番	説明
(1)	削除シンボル名を表示します
(2)	削除シンボル名を表示します
(3)	削除シンボルの種別を以下のように表示します データ種別 func: 関数名 data: 変数名 宣言種別 g: 外部定義 l: 内部定義

3.2.7 クロスリファレンス情報

-show=xreference を指定した場合，シンボルの参照情報（クロスリファレンス情報）を出力します。
クロスリファレンス情報の出力例を示します。

```
*** Cross Reference List ***
(1)  (2)         (3)             (4)        (5)
No   Unit Name  Global.Symbol  Location  External Information
0001 a
     SECTION=P  _func
                                00000100
                _func1
                                00000116
                _main
                                0000012c
                _g
                                00000136
     SECTION=B
                _a
                                00000190  0001(00000140:P)
                                          0002(00000178:P)
                                          0003(0000018c:P)
0002 b
     SECTION=P
                _func01
                                00000154  0001(00000148:P)
                _func02
                                00000166  0001(00000150:P)
0003 c
     SECTION=P
                _func03
                                00000184
```

項番	説明
(1)	Unit 番号。オブジェクト単位の識別番号
(2)	オブジェクト名。 リンク時の入力指定順になる
(3)	シンボル名。セクションごとに配置アドレスの昇順に出力される

項番	説明
(4)	シンボルの配置アドレス。 -form=relocate 指定時は，セクション先頭からの相対値となる
(5)	参照している外部シンボルのアドレスを表す。 出力形式は以下のようになる。 <Unit 番号 >(< アドレス or セクション内オフセット >:< セクション名 >)

3.2.8 合計セクションサイズ

ROM セクション，RAM セクション，およびプログラムセクションの合計サイズを出力します。
合計の出力例を示します。

```
*** Total Section Size ***
RAMDATA SECTION:         00000660 Byte(s) (1)
ROMDATA SECTION:         00000174 Byte(s) (2)
PROGRAM SECTION:         000016d6 Byte(s) (3)
```

項番	説明
(1)	RAM データセクションの合計サイズ
(2)	ROM データセクションの合計サイズ
(3)	プログラムセクションの合計サイズ

3.2.9 ベクタ情報

-show=vector を指定した場合，可変ベクタテーブルの内容を表示します。
合計の出力例を示します。

```
*** Variable Vector Table List ***
(1)    (2)
NO.    SYMBOL/ADDRESS
  0    $fdummy
  1    $fa
  2    00ff8800
  3    $fdummy
    :
  < 省略 >
```

項番	説明
(1)	ベクタ番号
(2)	シンボルを表示します シンボルが定義されていない場合はアドレスで表示します

3.2.10 CRC 情報

CRC オプション指定時に CRC の演算結果および出力位置アドレスを出力します。

```
*** CRC Code ***
CODE    : cb0b      (1)
ADDRESS : 00007ffe (2)
```

項番	説明
(1)	CRC 演算結果
(2)	CRC の演算結果の出力位置アドレス

3.3 ライブラリ・リスト

本節では，最適化リンケージエディタが出力するライブラリリストの内容と形式について説明します。

3.3.1 ライブラリリストの構成

ライブラリリストの構成と内容を示します。

表 3.3 ライブラリリストの構成と内容

No	リストの作成	内容	サブオプション注	-show オプション省略時
1	オプション情報	コマンドライン，サブコマンドで指定したオプション列を表示	-	出力する
2	エラー情報	エラーメッセージを表示	-	出力する
3	ライブラリ情報	ライブラリ情報を表示	-	出力する
4	ライブラリ内モジュール，セクション，シンボル情報	ライブラリ内モジュールを表示	-	出力する
		-show=symbol を指定した場合は，モジュール内シンボル名一覧も表示	-show=symbol	出力しない
		-show=section を指定した場合は，各モジュール内セクション名，シンボル名一覧も表示	-show=section	出力しない

注 すべてのオプションは，-list オプションを指定した場合に有効です。

3.3.2 オプション情報

コマンドライン，サブコマンドファイルで指定したオプション列を出力します。
オプション情報の出力例を示します（rlink -subcommand=test.sub -list -show を指定した場合）。

```
（test.sub の内容）
form    library
in      adhry.obj
output  test.lib
```

```
*** Options ***
-sub=test.sub
form      library
in        adhry.obj
output    test.lib
-list
-show
```

(1)
(2)

項番	説明
(1)	コマンドライン，サブコマンドで指定したオプション列を，指定順に出力します。

項番	説明
(2)	サブコマンドファイル test.sub 内のサブコマンドです。

3.3.3 エラー情報

エラー，ウォーニングなどのメッセージを出力します。エラー情報の出力例を示します。

```
*** Error Information ***
** W0561200 Backed up file "main.lib" into "main.lbk"  (1)
```

項番	説明
(1)	ウォーニングメッセージを出力します。

3.3.4 ライブラリ情報

ライブラリの種別を出力します。ライブラリ情報の出力例を示します。

```
*** Library Information ***
LIBRARY NAME=test.lib    (1)
CPU=RX600                (2)
ENDIAN=Big               (3)
ATTRIBUTE=system         (4)
NUMBER OF MODULE=1       (5)
```

項番	説明
(1)	ライブラリ名を表示します。
(2)	マイコン名を表示します。
(3)	エンディアン種別を表示します。
(4)	ライブラリファイルの属性がシステムライブラリかユーザライブラリかを表示します。
(5)	ライブラリ内モジュール数を表示します。

3.3.5 ライブラリ内モジュール，セクション，シンボル情報

ライブラリ内のモジュール一覧を出力します。
-show=symbol を指定した場合はモジュール内シンボル名一覧を，-show=section を指定した場合はモジュール内セクション名，シンボル名一覧を出力します。
ライブラリ内モジュール，セクション，シンボル情報の出力例を示します。

```
*** Library List ***
(1)              (2)
MODULE            LAST UPDATE
  (3)
  SECTION
    (4)
    SYMBOL

adhry
                  29-Feb-2000 12:34:56
  P
    _main
    _Proc0
    _Proc1
  C
  D
    _Version
  B
    _IntGlob
    _CharGlob
```

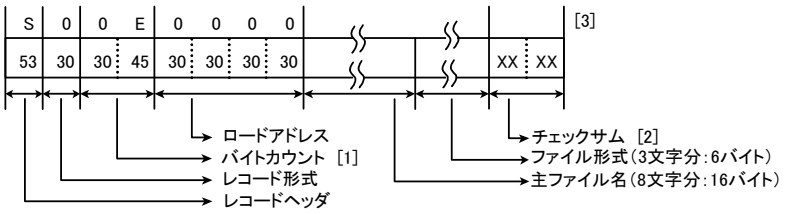
項番	説明
(1)	モジュール名を表示します。
(2)	モジュールを登録した日付を表示します。 モジュールが更新された場合は，最新の更新日付を表示します。
(3)	モジュール内セクション名を表示します。
(4)	セクション内をシンボル表示します。

3.4 モトローラ S 形式，インテル HEX 形式ファイル

本節では，最適化リンケージエディタによって出力されるモトローラ S 形式ファイル，およびインテル HEX 形式ファイルについて説明します。

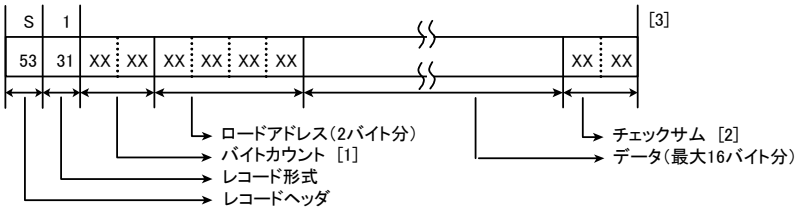
3.4.1 モトローラ S 形式ファイル

(a) ヘッダレコード(S0レコード)

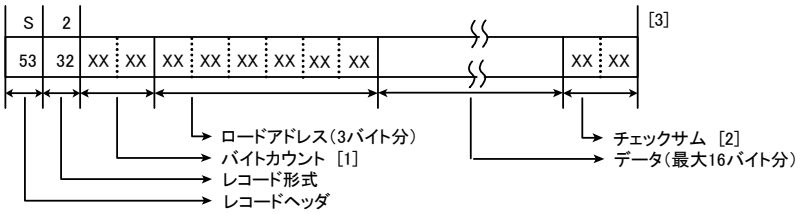


(b) データレコード(S1, S2, S3レコード)

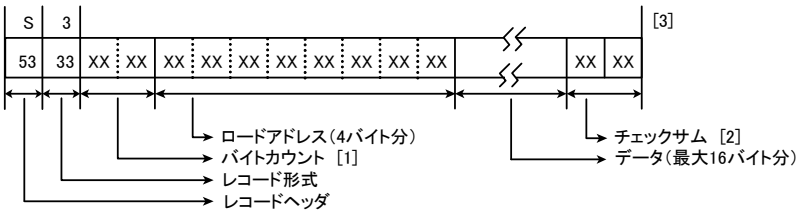
(i) ロードアドレスが0～FFFFの場合



(ii) ロードアドレスが10000～FFFFFFの場合

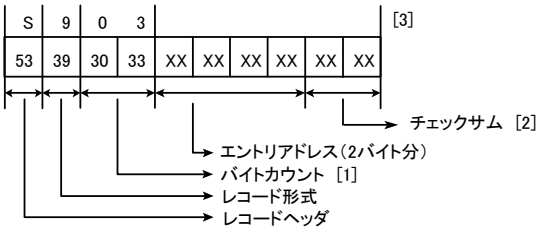


(iii) ロードアドレスが1000000～FFFFFFFFの場合

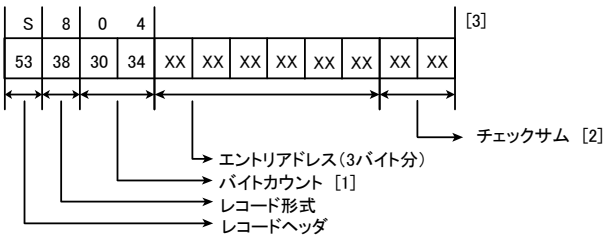


(c) エンドレコード(S9, S8, S7レコード)

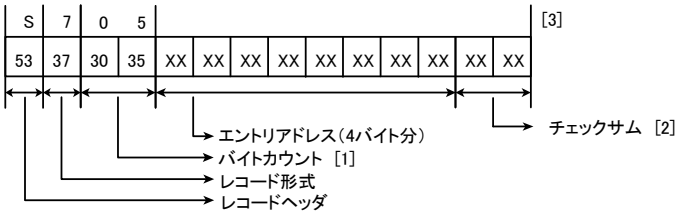
(i) エントリアドレスが0～FFFFの場合



(ii) エントリアドレスが10000～FFFFFFの場合



(iii) エントリアドレスが1000000～FFFFFFFFの場合



【注】 [1] ロードアドレス(またはエントリアドレス)からチェックサムまでのバイト数
[2] バイトカウンタからチェックサムの前までのデータ値をバイト単位に加算した結果の1の補数
[3] チェックサムの直後に改行コードが付加される

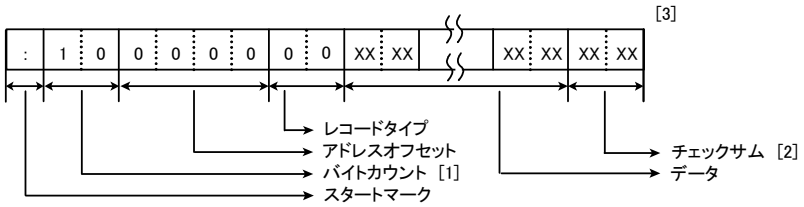
3.4.2 インテル HEX 形式ファイル

各データレコードの実行アドレスは以下のように求めます。

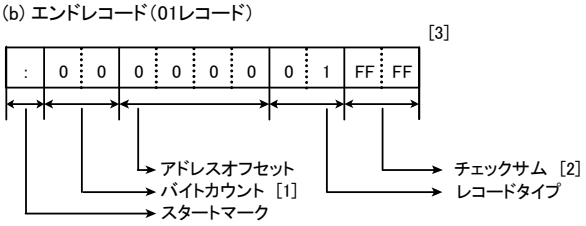
(1) セグメントアドレスの場合
(セグメントベースアドレス << 4) + (データレコードのアドレスオフセット)

(2) リニアアドレスの場合
(リニアベースアドレス << 16) + (データレコードのアドレスオフセット)

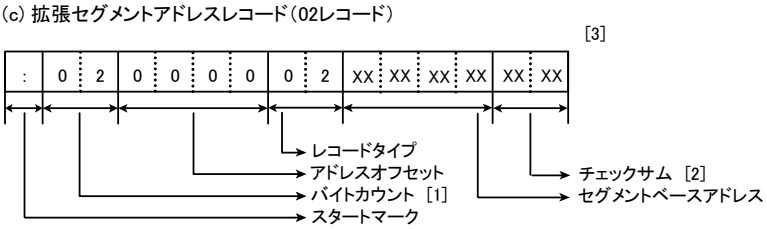
(a) データレコード(00レコード)



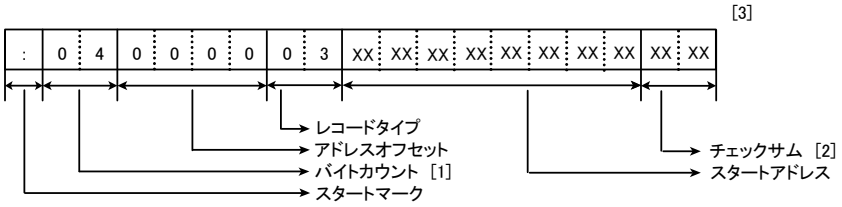
(b) エンドレコード(01レコード)



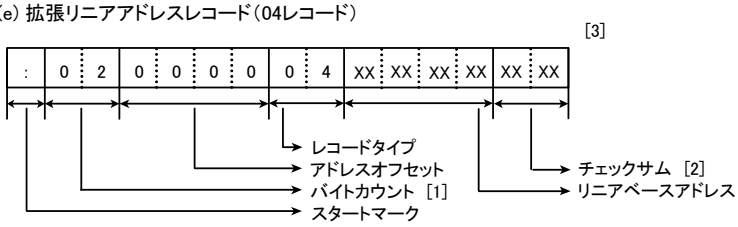
(c) 拡張セグメントアドレスレコード(02レコード)



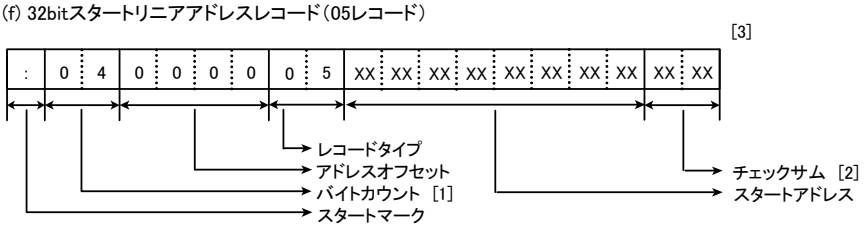
(d) スタートアドレスレコード(03レコード)



(e) 拡張リニアアドレスレコード(04レコード)



(f) 32bitスタートリニアアドレスレコード(05レコード)



【注】 [1] レコードタイプの次のデータから、チェックサムまでのバイト数
[2] バイトカウンタからチェックサムの前までのデータを、16進数で加算した結果の2の補数(下位8bitが有効)
[3] チェックサムの直後に改行コードが付加される

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

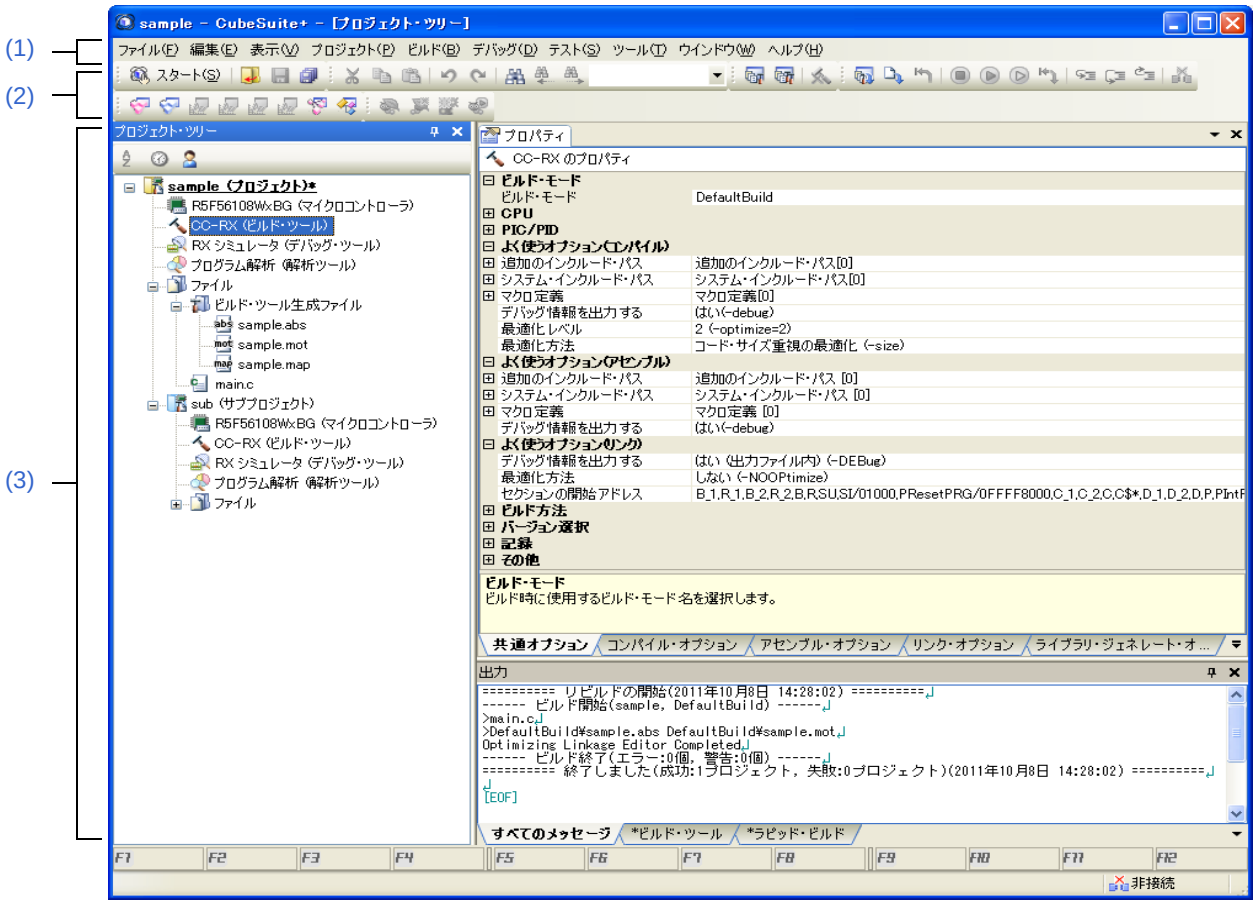
ウインドウ／パネル／ダイアログ名	機能概要
メイン・ウインドウ	CubeSuite+ を起動した際，最初にオープンするウインドウ
プロジェクト・ツリー パネル	プロジェクトの構成要素をツリー表示
プロパティ パネル	プロジェクト・ツリー パネルで選択しているビルド・ツール・ノード，ファイル，カテゴリ・ノードについて，詳細情報を表示，および設定を変更
エディタ パネル	テキスト・ファイル／ソース・ファイルを表示／編集
出力 パネル	ビルド・ツールから出力するメッセージを表示
ファイル追加 ダイアログ	新規ファイルを作成，およびプロジェクトに追加
フォルダとファイル追加 ダイアログ	既存のファイル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
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
対象外ファイルの追加 ダイアログ	本ダイアログの呼び出し元に設定する MISRA-C:2004 ルールのチェック対象外のファイルを選択
セクション設定 ダイアログ	セクションを追加，変更，削除
セクション追加，セクション編集，オーバーレイ配置セクションの追加 ダイアログ	セクションを追加，編集，または複数割り付け時のセクション名を設定
セクションのアドレス ダイアログ	セクションを追加，変更時のアドレスを設定

ウインドウ／パネル／ダイアログ名	機能概要
削除するセクション ダイアログ	セクションを削除
無効化するマクロの指定 ダイアログ	本ダイアログの呼び出し元に設定する無効化するプリデファインド・マクロを選択
ルール番号の指定 ダイアログ	本ダイアログの呼び出し元に設定する MISRA-C: 2004 ルール番号を選択
Misra2004 ルール・ファイルの指定 ダイアログ	本ダイアログの呼び出し元に設定する MISRA-C: 2004 ルール・ファイルファイルを選択
対象外ファイルの追加 ダイアログ	本ダイアログの呼び出し元に設定する MISRA-C: 2004 ルール・チェック対象外のファイルを選択
使用するライブラリ・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ライブラリ・ファイルの選択
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

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 ” が表示されます。
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 ” が表示されます。
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 ” が表示されます。
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 ” が表示されます。
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されます。
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・システム上からは削除されません。
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，一つのフォルダにまとめて保存します。

(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。

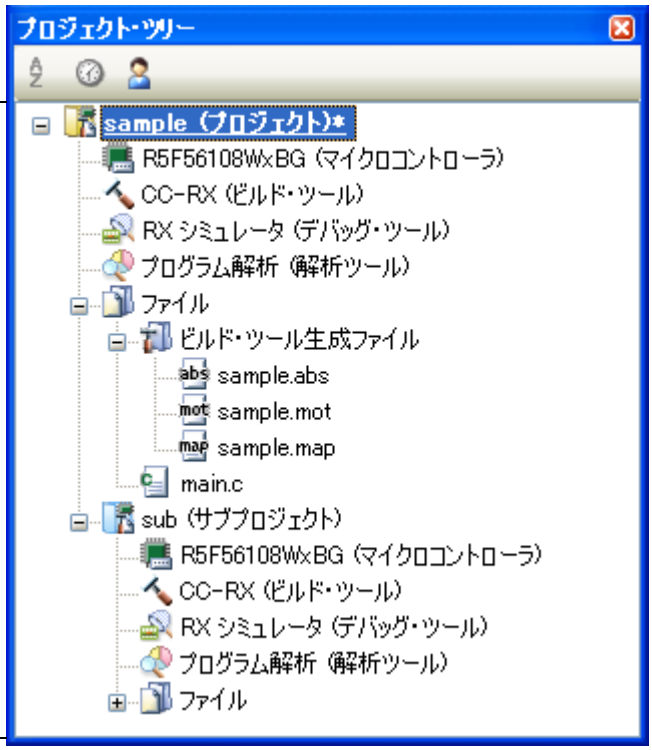
- プロジェクト・ツリー パネル
- プロパティ パネル
- エディタ パネル
- 出力 パネル

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（コンパイラ，アセンブラ等）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が，直下に表示されます。 - C ソース・ファイル（*.c） - C++ ソース・ファイル（*.cpp，*.cp，*.cc） - アセンブラ・ソース・ファイル（*.src） - ヘッダ・ファイル（*.h，*.hpp，*.inc） - ライブラリ・ファイル（*.lib） - 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。)	ビルド時に生成されるノードで，ビルド・ツールによって生成されたファイルのうち，以下のものが直下に表示されます。 - 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 (*.abs) ヘキサ・ファイル (*.hex) S レコード・ファイル (*.mot) バイナリ・データ・ファイル (*.bin) マップ・ファイル (*.map) - 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) リロケータブル・モジュール・ファイル（*.rel） マップ・ファイル (*.map，*.lbp) 本ノードに表示されているファイルは，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・ノード以下に生成されます。 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・ノード以下に表示されます。
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 (「2.3.4 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・ノード以下に作成されます。
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）がプロパティ パネルに表示され，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されます。
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。

<1> 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されます。

<2> 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，空のファイルが作成されます。

(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・システム上からは削除されません。

- 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は，ファイルに設定した個別オプションは保持されます。

備考 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は，ファイルは移動ではなく，コピーされます。なお，ファイルに設定した個別オプションは保持されません。

(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。

備考 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は，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されます。

備考 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は，カテゴリ・ノードは移動ではなく，コピーされます。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されません。

(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されます。

- 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されます。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，プロパティの値が太字表示されます。
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  が選択されます。

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
	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

ボタン	説明
	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。

(j) 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，“*” は消去されます。

通常のファイル	 main.c
編集後，未保存のファイル	 main.c*

(k) 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されます。

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

(l) 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，イタリック表示されます。

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

(m) 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，グレー表示され，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

(n) 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が検出されたファイルは，以下の例のように強調表示されます。

エラーもワーニングも検出されなかったファイル	 main.c
エラーが検出されたファイル	 main.c
ワーニングが検出されたファイル	 main.c

備考 1. エラーとワーニングの両方が検出されたファイルは，赤色表示されます。

備考 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されます。

- 以下のファイルは，太字表示されます。
  - 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は解除されます。

(o) 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されます。

通常のファイル	 main.c
ビルド対象外ファイル	 main.c

(p) 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 “*” が付加され，太字表示されます。
強調表示は，プロジェクトを保存すると解除されます。

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)***

(q) 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，下線が付加されます。

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)***

(r) 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されません。

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）
- C++ ソース・ファイル（*.cpp，*.cp，*.cc）
- ヘッダ・ファイル（*.h，*.hpp，*.inc）
- アセンブラ・ソース・ファイル（*.src）
- マップ・ファイル（*.map，*.lbp）
- ヘキサ・ファイル（*.hex）
- アセンブル・リスト・ファイル（*.lst）
- S レコード・ファイル（*.mot）
- 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，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されます。 ファイルを選択し，そのファイルを他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されます。 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，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。
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されます。
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，一つのフォルダにまとめて保存します。
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。
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されます。
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・システム上からは削除されません。 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されます。
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。
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，「2.12.1 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されます。
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されます。
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。
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されます。 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されます。 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（「(r) 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。
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されます。
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・システム上からは削除されません。 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されます。 選択しているファイルを他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されます。
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されます。
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されます。
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。
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

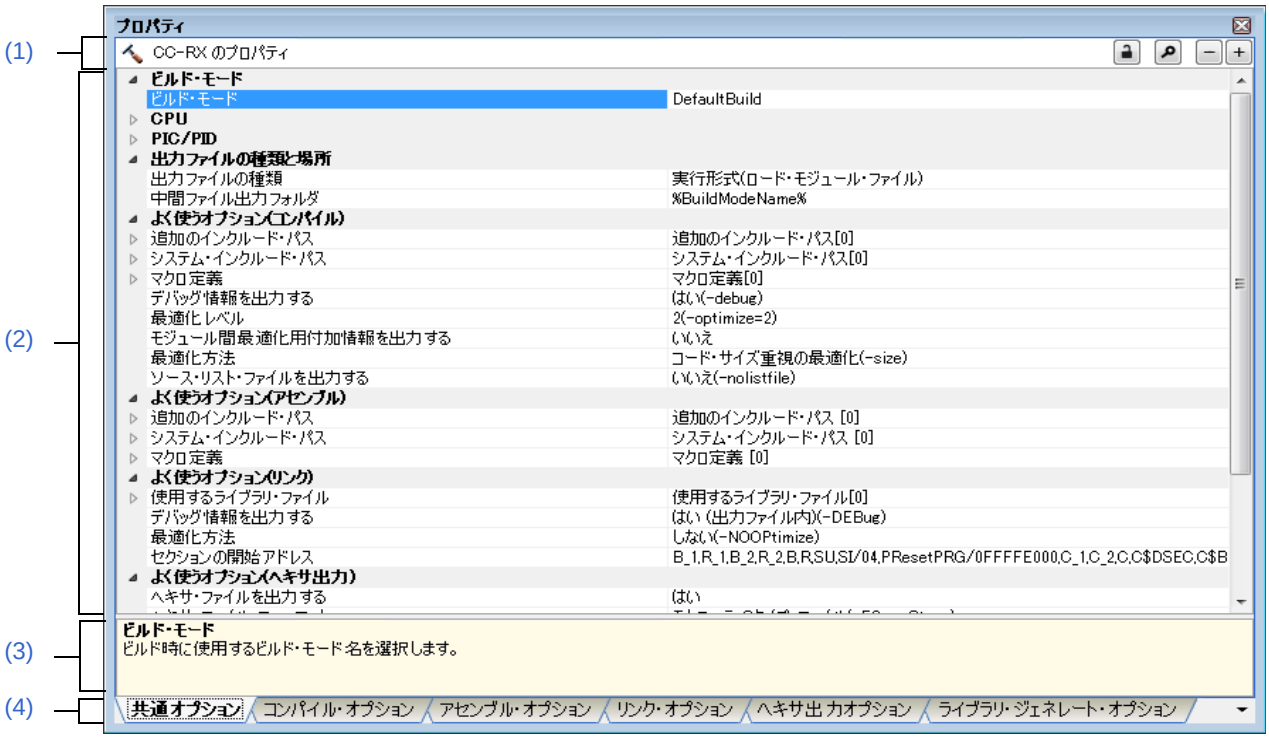
追加	プロジェクトにファイル，カテゴリ・ノードを追加するためのカスケード・メニューを表示します。
既存のファイルを追加 ...	既存のファイルを追加 ダイアログをオープンし，選択したファイルをプロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンされます。
新しいファイルを追加 ...	ファイル追加 ダイアログをオープンし，選択した種類でファイルを作成し，プロジェクトに追加します。追加先は，本ノードの直下です。 追加したファイルはファイルの拡張子に割り当てられたアプリケーションでオープンされます。
新しいカテゴリを追加	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。
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ノードの種類について，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。また，コード生成パネルでクリックした［コード生成］ボタンの種類，コード生成プレビュー パネルで選択したファイルに対応した情報の表示，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，以下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］
- ［プロパティ パネルからオープンするダイアログ］
- ［［編集］メニュー（プロパティ パネル専用部分）］
- ［［ヘルプ］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，プロジェクト・ノード，サブプロジェクト・ノード，マイクロコントローラ・ノード，設計ツール・ノード，ビルド・ツール・ノード，デバッグ・ツール・ノード，解析ツール・ノード，ファイル・ノード，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
- コード生成パネルにおいて，［コード生成］ボタンをクリックしたのち，［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
- コード生成プレビュー パネルにおいて，ファイルを選択したのち，［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，プロジェクト・ノード，サブプロジェクト・ノード，マイクロコントローラ・ノード，設計ツール・ノード，ビルド・ツール・ノード，デバッグ・ツール・ノード，解析ツール・ノード，ファイル・ノード，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) 対象名エリア，およびボタン群

(a) 対象名エリア

プロジェクト・ツリー パネルで選択しているノードの名称を表示します。
複数のノードを選択している場合，このエリアは空欄となります。

(b) ボタン群

	マウス，またはキーボードの誤った操作によるプロパティ値の変更を防ぐためにプロパティをロックします。  ：プロパティをロックします。  ：プロパティのロックを解除します。 なお，本ボタンは以下のタブのみに表示されます。 - ビルド・ツール・ノード選択時に表示されるタブ - C ソース・ファイル，C++ ソース・ファイル【CC-RX】，アセンブリ・ソース・ファイル【CC-RH】，アセンブラ・ソース・ファイル【CC-RX】【CA850】【CX】【NC30】【CA78K0R】【CA78K0】，オブジェクト・ファイル【CC-RH】，オブジェクト・モジュール・ファイル【CC-RX】【CA850】【CX】【NC30】【CA78K0R】【CA78K0】，リンク・ディレクティブ・ファイル【CA850】【CX】【CA78K0R】【CA78K0】，シンボル情報ファイル【CX】，ライブラリ・ファイル選択時に表示されるタブ（[ファイル情報] タブを除く）
	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。 プロパティ名，プロパティ値，アクティブヘルプ文字列を検索することができます。
	現在表示している詳細情報表示／変更エリアをすべて閉じた状態にします。
	現在表示している詳細情報表示／変更エリアをすべて展開した状態にします。

(2) 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プロジェクト・ノード，サブプロジェクト・ノード，マイクロコントローラ・ノード，ビルド・ツール・ノード，デバッグ・ツール・ノード，解析ツール・ノード，ファイル・ノード，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が展開表示されていることを示し，また，マークは，カテゴリ内の項目が折りたたみ表示され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。
 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。

(3) プロパティの説明エリア

詳細情報表示／変更エリアで選択したカテゴリや項目の簡単な説明を表示します。

(4) 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。
このパネルには，以下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）。

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されます。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されます。

[プロパティ パネルからオープンするダイアログ]

プロパティ パネルからオープンするダイアログには，次のものがあります。

- 文字列入力 ダイアログ
  詳細は，文字列入力 ダイアログ，「CubeSuite+ 統合開発環境 ユーザーズマニュアル ビルド編」，「CubeSuite+ 統合開発環境 ユーザーズマニュアル デバッグ編」を参照してください。
- テキスト編集 ダイアログ
  詳細は，テキスト編集 ダイアログ，「CubeSuite+ 統合開発環境 ユーザーズマニュアル ビルド編」，「CubeSuite+ 統合開発環境 ユーザーズマニュアル デバッグ編」を参照してください。

- パス編集 ダイアログ
  詳細は，「CubeSuite+ 統合開発環境 ユーザーズマニュアル ビルド編」を参照してください。

[[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。
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。

[[ヘルプ] メニュー（プロパティ パネル専用部分)]

プロパティ パネルのヘルプを開く	このパネルのヘルプを表示します。

[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

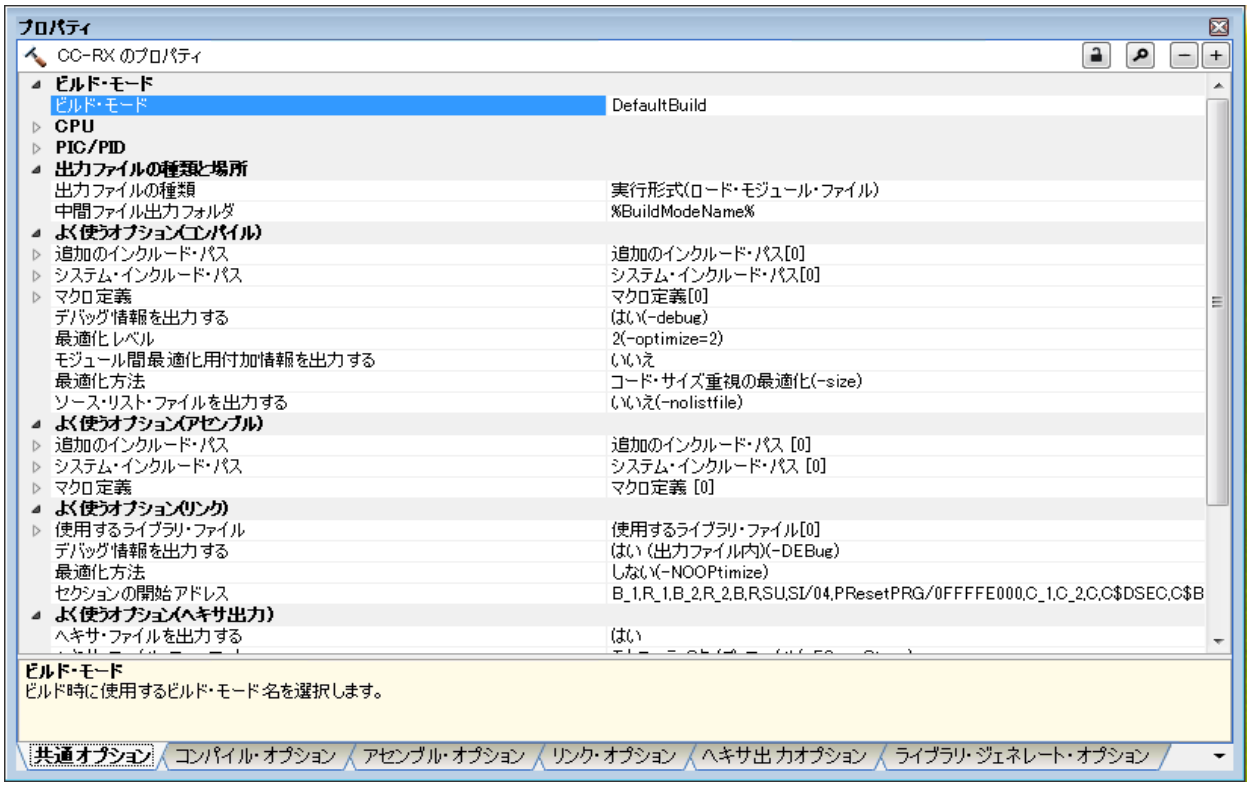
本タブでは，ビルド・ツールに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

(1) ［ビルド・モード］
(2) ［CPU］
(3) ［PIC/PID］
(4) ［出力ファイルの種類と場所］
(5) ［よく使うオプション（コンパイル）］
(6) ［よく使うオプション（アセンブル）］
(7) ［よく使うオプション（リンク）］
(8) ［よく使うオプション（ヘキサ出力）］
(9) ［よく使うオプション（ライブラリアン）］
(10) ［ビルド方法］
(11) ［バージョン選択］
(12) ［記録］
(13) ［その他］

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されます。

［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	［ライブラリアン・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，コンテキスト・メニューの［すべてデフォルトに戻す］は，本プロパティには適用されません。		
	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されるビルド・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（DefaultBuild 以外）	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) ［CPU］

CPU に関する詳細情報の表示，および設定の変更を行います。

命令セット・アーキテクチャ	命令セット・アーキテクチャを選択します。 コンパイラ、アセンブラおよびライブラリ・ジェネレータのオプション -isa に相当します。		
	デフォルト	プロジェクト作成時に選択した CPU の品種に従います。	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	RXv1 アーキテクチャ (-isa=rxv1)	RXv1 命令セットアーキテクチャ向けの命令コードを生成します。
		RXv2 アーキテクチャ (-isa=rxv2)	RXv2 命令セットアーキテクチャ向けの命令コードを生成します。
		なし	マイコン種別の選択を用います。
マイコン種別	マイコン種別を選択します。 コンパイラ、アセンブラおよびライブラリ・ジェネレータのオプション -cpu に相当します。 命令セット・アーキテクチャの項目で、「なし」を選択した場合だけ選択できます。		
	デフォルト	RX600 シリーズ (-cpu=rx600)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	RX600 シリーズ (-cpu=rx600)	RX600 向けの命令コードを生成します。
		RX200 または RX100 シリーズ (-cpu=rx200)	RX200 向けの命令コードを生成します。

<table>
<tr><td rowspan="5">浮動小数点演算命令を使用する</td><td colspan="3">浮動小数点演算命令を使用するかどうかを選択します。
コンパイラのオプション -fpu，-nofpu およびアセンブラのオプション -fpu, -nofpu に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">・[命令セット・アーキテクチャ] プロパティで，「なし」を選択した場合
　マイコン種別オプションに依存する
・[命令セット・アーキテクチャ] プロパティで，「なし」以外を選択した場合
　ターゲット・デバイス固有の値</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>マイコン種別オプションに依存する</td><td>マイコン種別オプションに従います。
[命令セット・アーキテクチャ] プロパティで，「なし」以外を選択した場合は非表示となります。</td></tr>
<tr><td>はい (-fpu)</td><td>FPU 命令を使用したコード生成を行います。
なお，本項目は，［マイコン種別］プロパティで［RX200 (-cpu=rx200)］を選択した場合は，選択できません。</td></tr>
<tr><td>いいえ (-nofpu)</td><td>FPU 命令を使用せず，実行時ルーチン呼び出しによるコード生成を行います。</td></tr>
<tr><td rowspan="4">データのエンディアン</td><td colspan="3">データのエンディアンを選択します。
コンパイラおよびライブラリ・ジェネレータのオプション -endian，アセンブラのオプション -endian に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">Little-endian データ (-endian=little)</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>Big-endian データ (-endian=big)</td><td>データのバイト並びが big endian になります。</td></tr>
<tr><td rowspan="5">浮動小数点定数演算の丸め方式</td><td colspan="3">浮動小数点定数演算の丸め方式を選択します。
本オプションでは，実行時の浮動小数点演算における丸め方式を変更することはできません。
コンパイラ，およびライブラリ・ジェネレータのオプション -round に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">round to nearest で丸める (-round=nearest)</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>round to zero で丸める (-round=zero)</td><td>round to zero で丸めます。</td></tr>
<tr><td>round to nearest で丸める (-round=nearest)</td><td>round to nearest で丸めます。</td></tr>
</table>

浮動小数点定数に非正規化数を記述した場合の扱い	浮動小数点定数に非正規化数を記述した場合の扱いを選択します。 本オプションでは，実行時の浮動小数点演算における非正規化数の扱いを変更することはできません。 コンパイラ，およびライブラリ・ジェネレータのオプション -denormalize に相当します。		
	デフォルト	0 として扱う (-denormalize=off)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	0 として扱う (-denormalize=off)	非正規化数を 0 として扱います。
		非正規化数として扱う (-denormalize=on)	非正規化数を非正規化数として扱います。
double 型、および long double 型の精度	double 型，および long double 型の精度を選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -dbl_size に相当します。		
	デフォルト	単精度として扱う (-dbl_size=4)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	単精度として扱う (-dbl_size=4)	double 型，および long double 型を単精度として扱います。
		倍精度として扱う (-dbl_size=8)	double 型，および long double 型を倍精度として扱います。
int 型を short 型に置換する	int 型を short 型に置換するかどうかを選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -int_to_short 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 (-int_to_short)	int を short に，unsigned int を unsigned short に置換します。
		いいえ	int を short に，unsigned int を unsigned short に置換しません。
char 型の符号	符号指定のない char 型の符号を選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -signed_char，-unsigned_char に相当します。		
	デフォルト	unsigned char 型として扱う (-unsigned_char)	
	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 char 型として扱う (-signed_char)	char 型を signed char 型として扱います。
		unsigned char 型として扱う (-unsigned_char)	char 型を unsigned char 型として扱います。

ビットフィールド型の符号	符号指定のないビットフィールド型の符号を選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション，-signed_bitfield，-unsigned_bitfield に相当します。		
	デフォルト	符号なし型として扱う (-unsigned_bitfield)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	符号付き型として扱う (-signed_bitfield)	ビットフィールドの符号を符号付き型として扱います。
		符号なし型として扱う (-unsigned_bitfield)	ビットフィールドの符号を符号なし型として扱います。
列挙型のサイズを自動選択する	列挙型データのサイズを自動選択するかどうかを選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -auto_enum 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 (-auto_enum)	enum 宣言した列挙型のデータを，列挙値が収まる最小型として処理します。
		いいえ	列挙型サイズを signed long 型として処理します。
ビットフィールドメンバの並び順	ビットフィールドメンバの並び順を選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -bit_order に相当します。		
	デフォルト	右から割り付け (-bit_order=right)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	左から割り付け (-bit_order=left)	上位ビットからメンバを割り付けます。
		右から割り付け (-bit_order=right)	下位ビットからメンバを割り付けます。
構造体メンバのアライメントを 1 とする	構造体メンバ，クラスメンバのアライメント数を 1 とするかどうかを選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -pack オプション，-unpack に相当します。		
	デフォルト	いいえ (-unpack)	
	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 (-pack)	構造体メンバ，クラスメンバのアライメント数を 1 とします。
		いいえ (-unpack)	データのアライメントに従います。
C++ 例外処理機能 (try, catch, throw) を有効にする	C++ 例外処理機能 (try, catch, throw) を有効にするかどうかを選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -exception，-noexception に相当します。		
	デフォルト	いいえ (-noexception)	
	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 (-exception)	例外処理機能を有効にします。
		いいえ (-noexception)	例外処理機能を無効にします。

C++ 実行時型情報 (dynamic_cast, typeid) を有効にする	C++ 実行時型情報 (dynamic_cast，typeid) を有効にするかどうかを選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -rtti に相当します。		
	デフォルト	いいえ (-rtti=off)	
	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 (-rtti=on)	dynamic_cast，typeid を有効にします。
		いいえ (-rtti=off)	dynamic_cast，typeid を無効にします。
高速割り込み関数でのみ使用する汎用レジスタ	高速割り込み関数でのみ使用する汎用レジスタを選択します。 ［ROM 用ベースレジスタ］プロパティ，［RAM 用ベースレジスタ］プロパティ，［アドレス値を設定するベースレジスタのレジスタ］プロパティで指定したレジスタを本オプションで指定した場合，エラーとなります。 コンパイラおよびライブラリ・ジェネレータのオプション -fint_register，アセンブラのオプション -fint_register に相当します。		
	デフォルト	なし (-fint_register=0)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	なし (-fint_register=0)	高速割り込み専用のレジスタを使用しません。
		R13 (-fint_register=1)	R13 を高速割り込み専用で使用します。
		R12, R13 (-fint_register=2)	R13 ～ R12 を高速割り込み専用で使用します。
		R11 ～ R13 (-fint_register=3)	R13 ～ R11 を高速割り込み専用で使用します。
		R10 ～ R13 (-fint_register=4)	R13 ～ R10 を高速割り込み専用で使用します。
分岐幅	分岐幅を選択します。 コンパイラ，およびライブラリ・ジェネレータのオプション -branch オプションに相当します。		
	デフォルト	24bit 以内であるとしてコンパイル (-branch=24)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	16bit 以内であるとしてコンパイル (-branch=16)	分岐幅のサイズが 16bit 以内であるとしてコンパイルします。
		24bit 以内であるとしてコンパイル (-branch=24)	分岐幅のサイズが 24bit 以内であるとしてコンパイルします。
		指定しない (-branch=32)	分岐幅のサイズを限定しません。

ROM 用ベースレジスタ	プログラム全体で，ベースアドレスとして固定で使用する汎用レジスタを指定します。 「base=rom= *レジスタ A*」を指定した場合は，const 変数のアクセスはすべて指定した「*レジスタ A*」相対で行います。 ただし，定数領域セクション全体の大きさが 64KB ～ 256KB 以内でなければなりません。 コンパイラおよびライブラリ・ジェネレータのオプション -base，アセンブラのオプション -base に相当します。 また，本プロパティは，［PIC/PID］カテゴリの［PID 機能を有効にする］プロパティが［いいえ］のときに表示します。		
	デフォルト	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	なし	ROM 用ベースレジスタを指定しません。
		R8 (-base=rom=R8)	R8 を ROM 用ベースレジスタに指定します。
		R9 (-base=rom=R9)	R9 を ROM 用ベースレジスタに指定します。
		R10 (-base=rom=R10)	R10 を ROM 用ベースレジスタに指定します。
		R11 (-base=rom=R11)	R11 を ROM 用ベースレジスタに指定します。
		R12 (-base=rom=R12)	R12 を ROM 用ベースレジスタに指定します。
		R13 (-base=rom=R13)	R13 を ROM 用ベースレジスタに指定します。
RAM 用ベースレジスタ	プログラム全体で，ベースアドレスとして固定で使用する汎用レジスタを指定します。 「base=ram= *レジスタ B*」を指定した場合は，初期化変数および未初期化変数のアクセスはすべて指定した「*レジスタ B*」相対で行います。 ただし，RAM データ全体の大きさが 64KB ～ 256KB 以内でなければなりません。 コンパイラおよびライブラリ・ジェネレータのオプション -base，アセンブラのオプション -base に相当します。		
	デフォルト	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	なし	RAM 用ベースレジスタを指定しません。
		R8 (-base=ram=R8)	R8 を RAM 用ベースレジスタに指定します。
		R9 (-base=ram=R9)	R9 を RAM 用ベースレジスタに指定します。
		R10 (-base=ram=R10)	R10 を RAM 用ベースレジスタに指定します。
		R11 (-base=ram=R11)	R11 を RAM 用ベースレジスタに指定します。
		R12 (-base=ram=R12)	R12 を RAM 用ベースレジスタに指定します。
		R13 (-base=ram=R13)	R13 を RAM 用ベースレジスタに指定します。

アドレス値を設定するベースレジスタのアドレス	アドレス値を設定するベースレジスタのアドレスを指定します。 コンパイラおよびライブラリ・ジェネレータのオプション -base，アセンブラのオプション -base に相当します。		
	デフォルト	00000000（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）	
アドレス値を設定するベースレジスタのレジスタ	プログラム全体で，ベースアドレスとして固定で使用する汎用レジスタを指定します。 「*アドレス値 = レジスタ C*」を指定した場合は，コンパイル時に割り付けアドレスが確定している領域のうち，アドレス値から 64KB ～ 256KB 以内の領域のアクセスを，指定した「*レジスタ C*」相対で行います。 コンパイラおよびライブラリ・ジェネレータのオプション -base，アセンブラのオプション -base に相当します。 なお，本プロパティは，［PIC/PID］カテゴリの［PID 機能を有効にする］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	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	なし	アドレス値を設定するベースレジスタを指定しません。
		R8 (-base=< アドレス値 >=R8)	R8 をアドレス値を設定するベースレジスタに指定します。
		R9 (-base=< アドレス値 >=R9)	R9 をアドレス値を設定するベースレジスタに指定します。
		R10 (-base=< アドレス値 >=R10)	R10 をアドレス値を設定するベースレジスタに指定します。
		R11 (-base=< アドレス値 >=R11)	R11 をアドレス値を設定するベースレジスタに指定します。
		R12 (-base=< アドレス値 >=R12)	R12 をアドレス値を設定するベースレジスタに指定します。
		R13 (-base=< アドレス値 >=R13)	R13 をアドレス値を設定するベースレジスタに指定します。
CPU タイプ特有の問題を回避する	CPU タイプ特有の問題を回避するかどうかを選択します。 コンパイラおよびライブラリ・ジェネレータのオプション -patch，アセンブラのオプション -patch 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	いいえ	組み込み関数 set_ipl の呼び出しに対する生成コードは，MVTIPL 命令を含んだものとなります。
		はい (RX610 グループ向け) (-patch=rx610)	MVTIPL 命令を生成コードに使用しません (RX610 グループ向け)。

割り込み関数で ACC を退避・回復する	割り込み関数でアキュムレータ (ACC) を退避・回復するかどうかを選択します。 生成される退避・回復コードは，#pragma interrupt に acc を選択したときに生成されるコードと同じものです。 コンパイラ，およびライブラリ・ジェネレータのオプション -save_acc 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 (-save_acc)	割り込み関数で ACC を退避・回復します。
		いいえ	割り込み関数で ACC を退避・回復しません。

(3) ［PIC/PID］

PIC/PID 機能に関する詳細情報の表示，および設定の変更を行います。

PIC 機能を有効にする	PIC(位置独立コード) 機能を有効にするかどうかを選択します。 PIC においては，関数呼び出しはすべて BSR または BRA 命令を用いて行い，また関数のアドレスを取得するときは PC からの相対アドレスを用いるようにします。これによって，PIC はリンク後に任意のアドレスに配置することができます。 コンパイラおよびライブラリ・ジェネレータのオプション -pic，アセンブラのオプション -pic 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 (-pic)	PIC 機能を有効にします。
		いいえ	PIC 機能を無効にします。
PID 機能を有効にする	PID(位置独立データ) 機能を有効にするかどうかを選択します。 定数領域セクション C, C_2 および C_1，リテラルセクション L と switch 文分岐テーブルセクション W, W_2 および W_1 を PID(位置独立データ) とします。 PID のアクセスは，すべて PID レジスタからの相対アドレスで行います。 コンパイラおよびライブラリ・ジェネレータのオプション -pid，アセンブラのオプション -pid に相当します。 なお，本プロパティは，［CPU］カテゴリの［ROM 用ベースレジスタ］プロパティで［なし］，および［PID レジスタをコード生成に使用する］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	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 (オフセットの最大幅：16 ビット) (-pid=16)	PID 機能を有効にします。 16 ビット (64KB ～ 256KB) のアドレッシングに対応します。
		はい (オフセットの最大幅：制限なし) (-pid=32)	PID 機能を有効にします。 32 ビット (4GB) のアドレッシングに対応します。
		いいえ	PID 機能を無効にします。
PID レジスタをコード生成に使用する	PID レジスタをコード生成に使用するかどうかを選択します。 PID 機能が有効なアプリケーションプログラムから呼び出されるマスタプログラムは，本オプションでコンパイル / アセンブルする必要があります。 コンパイラおよびライブラリ・ジェネレータのオプション -nouse_pid_register，アセンブラのオプション -nouse_pid_register に相当します。 なお，本プロパティは，［PID 機能を有効にする］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	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	PID レジスタをコード生成に使用します。
		いいえ (-nouse_pid_register)	PID レジスタをコード生成に使用しません。

(4) ［出力ファイルの種類と場所］
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。

<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になります。
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（ヘキサ・ファイル)］のみを表示します。ただし、［よく使うオプション (ヘキサ出力)］カテゴリの［ヘキサ・ファイルを出力する］プロパティで［いいえ］を選択した場合、［実行形式 (ヘキサ・ファイル)］を表示しません。
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="3">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>ロード・モジュール・ファイルをデバッグ対象にします。</td></tr>
<tr><td>実行形式 (ヘキサ・ファイル)</td><td>ヘキサ・ファイルをデバッグ対象にします。</td></tr>
<tr><td>ライブラリ形式</td><td>デバッグ対象に設定しません。</td></tr>
<tr><td rowspan="4">中間ファイル出力フォルダ</td><td colspan="3">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。
%ProjectName% ：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。
空欄の場合は，プロジェクトのフォルダを指定したものとみなします。</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</td></tr>
<tr><td>指定可能値</td><td colspan="2">247 文字までの文字列</td></tr>
</table>

(5) ［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。

<table>
<tr><td rowspan="4">追加のインクルード・パス</td><td colspan="3">インクルード・ファイルの存在するパス名を指定します。
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。
なお，パスはプロジェクト・フォルダを基点とします。
コンパイラのオプション -include 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。</td></tr>
<tr><td>デフォルト</td><td colspan="2">追加のインクルード・パス [定義数]</td></tr>
<tr><td>変更方法</td><td colspan="2">[...] 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 colspan="2">247 文字までの文字列
65536 個まで指定可能です。</td></tr>
<tr><td rowspan="3">システム・インクルード・パス</td><td colspan="3">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。
コンパイラのオプション -include に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">システム・インクルード・パス [定義数]</td></tr>
<tr><td>変更方法</td><td colspan="2">[...] 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 rowspan="3">マクロ定義</td><td colspan="3">定義したいマクロ名を指定します。
「マクロ名 = 文字列」の形式で 1 行に 1 つずつ指定します。
「= 文字列」の部分は省略可能で，省略した場合，そのマクロ名が定義されたものと仮定します。
コンパイラのオプション -define 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>デフォルト</td><td colspan="2">マクロ定義 [定義数]</td></tr>
<tr><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 rowspan="4">デバッグ情報を出力する</td><td colspan="3">デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。コンパイラのオプション -debug，-nodebug に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ (-nodebug)</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 (-debug)</td><td>デバッグ情報をオブジェクト・モジュール・ファイルに出力します。</td></tr>
<tr><td>いいえ (-nodebug)</td><td>デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。</td></tr>
</table>

最適化レベル	最適化レベルを選択します。 コンパイラのオプション -optimize に相当します。		
	デフォルト	2 (-optimize=2)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	0 (-optimize=0)	最適化を実施しません。
		1 (-optimize=1)	自動変数のレジスタ割り付け，関数出口ブロックの統合，統合可能な複数命令の統合など，一部最適化を実施します。
		2 (-optimize=2)	全般的に最適化を実施します。
		Max (-optimize=max)	実施可能な最適化を最大限に行います。
モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 コンパイラのオプション -goptimize 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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。
最適化方法	最適化方法を選択します。 コンパイラのオプション -speed， -size に相当します。		
	デフォルト	コード・サイズ重視の最適化 (-size)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	実行性能重視の最適化 (-speed)	実行性能重視の最適化を実施します。
		コード・サイズ重視の最適化 (-size)	コードサイズ重視の最適化を実施します。
ソース・リスト・ファイルを出力する	ソース・リスト・ファイルを出力するかどうかを選択します。 コンパイラのオプション -listfile， -nolistfile 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 (-listfile)	ソース・リスト・ファイルを出力します。
		いいえ (-nolistfile)	ソース・リスト・ファイルを出力しません。

C/C++ ソースを出力する	ソース・リスト・ファイルの内容の設定を行います。 C/C++ ソースを出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=source)	C/C++ ソースを出力します。
		いいえ	C/C++ ソースを出力しません。
条件アセンブルで偽の行を出力する	ソース・リスト・ファイルの内容の設定を行います。 条件アセンブルで偽の行を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=conditionals)	条件アセンブルで偽の行を出力します。
		いいえ	条件アセンブルで偽の行を出力しません。
.DEFINE 置換前の情報を出力する	ソース・リスト・ファイルの内容の設定を行います。 .DEFINE 置換前の情報を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=definitions)	.DEFINE 置換前の情報を出力します。
		いいえ	.DEFINE 置換前の情報を出力しません。
アセンブラ・マクロ記述展開行を出力する	ソース・リスト・ファイルの内容の設定を行います。 アセンブラ・マクロ記述展開行を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=expansions)	アセンブラ・マクロ記述展開行を出力します。
		いいえ	アセンブラ・マクロ記述展開行を出力しません。

(6) ［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。

<table>
<tr><td>追加のインクルード・パス</td><td colspan="3">インクルード・ファイルの存在するパス名を指定します。
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。
なお，パスはプロジェクト・フォルダを基点とします。
アセンブラのオプション -include 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。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">追加のインクルード・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">[...] 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 colspan="2">247 文字までの文字列
65536 個まで指定可能です。</td></tr>
<tr><td>システム・インクルード・パス</td><td colspan="3">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。
アセンブラのオプション -include に相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">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 colspan="2">[...] 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 colspan="2">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>マクロ定義</td><td colspan="3">定義したいマクロ名を指定します。
「マクロ名 = 文字列」の形式で 1 行に 1 つずつ指定します。
アセンブラのオプション -define 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">マクロ定義 [定義数]</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">32767 文字までの文字列
65536 個まで指定可能です。</td></tr>
<tr><td>デバッグ情報を出力する</td><td colspan="3">デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。アセンブラのオプション -debug，-nodebug に相当します。
なお、本プロパティは、［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］を選択した場合に表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">いいえ (-nodebug)</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 (-debug)</td><td>デバッグ情報をオブジェクト・モジュール・ファイルに出力します。</td></tr>
<tr><td></td><td>いいえ (-nodebug)</td><td>デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。</td></tr>
</table>

モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 アセンブラのオプション -goptimize 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］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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。
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。 アセンブラのオプション -listfile，-nolistfile 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］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。		
	デフォルト	いいえ (-nolistfile)	
	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 (-listfile)	アセンブル・リスト・ファイルを出力します。
		いいえ (-nolistfile)	アセンブル・リスト・ファイルを出力しません。
条件アセンブルで偽の行を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 条件アセンブルで偽の行を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］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 (-show=conditionals)	条件アセンブルで条件が偽となった行を出力します。
		いいえ	条件アセンブルで条件が偽となった行を出力しません。
.DEFINE 置換前の情報を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 .DEFINE 置換前の情報を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］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 (-show=definitions)	.DEFINE 置換前の情報を出力します。
		いいえ	.DEFINE 置換前の情報を出力しません。

アセンブラ・マクロ記述展開行を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 アセンブラ・マクロ記述展開行を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］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 (-show=expansions)	アセンブラ・マクロ記述展開行を出力します。
		いいえ	アセンブラ・マクロ記述展開行を出力しません。

(7) ［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。

使用するライブラリ・ファイル	ライブラリ・ファイルを指定します。 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。 %ProjectName%: 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。 リンカのオプション -library 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	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	32767 文字までの文字列 65536 個まで指定可能です。	
デバッグ情報を出力する	デバッグ情報を出力するかどうかを選択します。 リンカのオプション -debug，-sdebug，および -nodebug に相当します。		
	デフォルト	はい (出力ファイル内) (-DEBug)	
	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 (出力ファイル内) (-DEBug)	デバッグ情報を出力します（出力ファイル内）。
		はい (デバッグ情報ファイル出力) (-SDebug)	デバッグ情報ファイルを出力します。
		いいえ (-NODEBug)	デバッグ情報を出力しません。
最適化方法	最適化方法を選択します。 リンカのオプション -nooptimize，および -optimize に相当します。		
	デフォルト	しない (-NOOPtimize)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	しない (-NOOPtimize)	モジュールの最適化を行いません。
		すべて (-OPtimize)	すべての最適化を行います。
		スピード重視 (-OPtimize=SPeed)	実行速度優先の最適化を行います。
		安全な最適化 (-OPtimize=SAFe)	安全な最適化を行います。
		カスタム	指定した項目の最適化を行います。
未参照シンボルを削除する	未参照シンボルを削除するかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=Symbol_delete)	未参照シンボルを削除します。
		いいえ	未参照シンボルを削除しません。

複数の同一命令列をサブルーチン化する	複数の同一命令列をサブルーチン化するかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=SAMe_code)	複数の同一命令列をサブルーチン化します。
		いいえ	複数の同一命令列をサブルーチン化しません。
最小コードサイズ	共通コード統合最適化で，最適化対象となる最小コードサイズを指定します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［複数の同一命令列をサブルーチン化する］プロパティで［はい (-OPtimize=SAMe_code)］を選択した場合のみ表示します。		
	デフォルト	1E（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	8 ～ 7FFF（16 進数）	
コードサイズがより小さくなる命令に置き換える	コードサイズがより小さくなる命令に置き換えるかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=SHort_format)	コードサイズがより小さくなる命令に置き換えます。
		いいえ	コードサイズがより小さくなる命令に置き換えません。
分岐命令サイズを最適化する	プログラムの配置情報に基づいて，分岐命令サイズを最適化するかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=Branch)	プログラムの配置情報に基づいて，分岐命令サイズを最適化します。
		いいえ	分岐命令サイズを最適化しません。
セクションの開始アドレス	セクションの開始アドレスを指定します。 リンカのオプション -start に相当します。		
	デフォルト	ターゲット・デバイス固有の値	
	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，セクション設定 ダイアログによる編集	
	指定可能値	1022 文字までの文字列	

(8) ［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・ファイルを出力する	ヘキサ・ファイルを出力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	ヘキサ・ファイルを出力します。
		いいえ	ヘキサ・ファイルを出力しません。
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。 リンカのオプション -form に相当します。		
	デフォルト	モトローラ・S タイプ・ファイル (-Form=Stype)	
	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・ファイル (-Form=Hexadecimal)	インテル拡張ヘキサ・ファイルを出力します。
		モトローラ・S タイプ・ファイル (-Form=Stype)	モトローラ・S タイプ・ファイルを出力します。
		バイナリ・ファイル (-Form=Binary)	バイナリ・ファイルを出力します。
出力フォルダ	ヘキサ・ファイルの出力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。 %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。 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。 リンカのオプション -output に相当します。 なお、本プロパティは、［ヘキサ・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	%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	

出力ファイル名	ヘキサ・ファイル名をを指定します。 拡張子を省略した場合は、［ヘキサ・ファイル・フォーマット］プロパティの選択に依存して、自動的に付加します。 ［インテル拡張ヘキサ・ファイル (-Form=Hexadecimal)］を選択している場合：.hex ［モトローラ・S タイプ・ファイル (-Form=Stype)］を選択している場合：.mot ［バイナリ・ファイル (-Form=Binary)］を選択している場合：.bin 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。 %ProjectName% ：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。 リンカのオプション -output に相当します。 なお、本プロパティは、［ヘキサ・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	%ProjectName%.abs
	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
分割出力ファイル	分割出力ファイルを指定します。 「*ファイル名 = 先頭アドレス - 終了アドレス*」、または「*ファイル名 = セクション名*」の形式で 1 行に 1 つずつ指定します。 セクション名を複数指定する場合は、「*ファイル名 = セクション名* : *セクション名*」のように、コロンで区切って指定します (例：file1.mot=sec1:sec2)。 アドレスは 16 進数で指定します (例：file2.mot=400-4ff) 拡張子を省略した場合は、［ヘキサ・ファイル・フォーマット］プロパティの選択に依存して、自動的に付加します。 ［インテル拡張ヘキサ・ファイル (-Form=Hexadecimal)］を選択している場合：.hex ［モトローラ・S タイプ・ファイル (-Form=Stype)］を選択している場合：.mot ［バイナリ・ファイル (-Form=Binary)］を選択している場合：.bin 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。 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。 リンカのオプション -output に相当します。 分割出力ファイル名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	分割出力ファイル［*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	255 文字までの文字列 65536 個まで指定可能

(9) ［よく使うオプション（ライブラリアン）］

ライブラリ生成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，ライブラリ用プロジェクトの場合で、［ライブラリアン・オプション］タブの［出力］カテゴリの［出力ファイル形式］プロパティが［リロケータブル・モジュール・ファイル (-Form=Relocate)］の時のみ表示します。

<table>
<tr><td rowspan="5">デバッグ情報を出力する</td><td colspan="3">デバッグ情報を出力するかどうかを選択します。
リンカのオプション -nodebug，-debug に相当します。
なお、本プロパティは、［ライブラリアン・オプション］タブの［出力］カテゴリの［出力ファイル形式］プロパティが［リロケータブル・モジュール・ファイル (-Form=Relocate)］の時のみ表示します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">はい (出力ファイル内) (-DEBug)</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 (出力ファイル内) (-DEBug)</td><td>デバッグ情報を出力します（出力ファイル内）。</td></tr>
<tr><td>いいえ (-NODEBug)</td><td>デバッグ情報を出力しません。</td></tr>
</table>

(10) ［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「2.12.5 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。。
		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，ソース・ファイルを再コンパイル／アセンブルしません。
パス、リンク順の互換性を保つ	High-performance Embedded Workshop のパス指定、リンク順と互換性を保つかどうかを選択します。		
	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	High-performance Embedded Workshop の仕様に則った、パス指定、リンク順になります。
		いいえ	本製品の仕様に則った、パス指定、リンク順になります。

(11) ［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・パッケージが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	

(12) ［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。

(13) ［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。

<table>
<tr><td rowspan="5">出力メッセージ・フォーマット</td><td colspan="3">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。
%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%:ビルド中の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。</td></tr>
<tr><td>デフォルト</td><td colspan="2">%TargetFiles%</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（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</td></tr>
<tr><td rowspan="3">指定可能値</td><td>%TargetFiles%</td><td>出力メッセージにファイル名を表示します。</td></tr>
<tr><td>%TargetFiles%:
%Options%</td><td>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。</td></tr>
<tr><td>%Program% %Options%</td><td>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。</td></tr>
<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="3">ビルド・オプション一覧（「2.12.4 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。
%Program%:実行中のプログラム名に置換します。
%Options%: ビルド時のコマンド・ライン・オプションに置換します。
%TargetFiles%:ビルド中の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 colspan="2">%TargetFiles% : %Program% %Options%</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，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 colspan="2">256 文字までの文字列</td></tr>
</table>

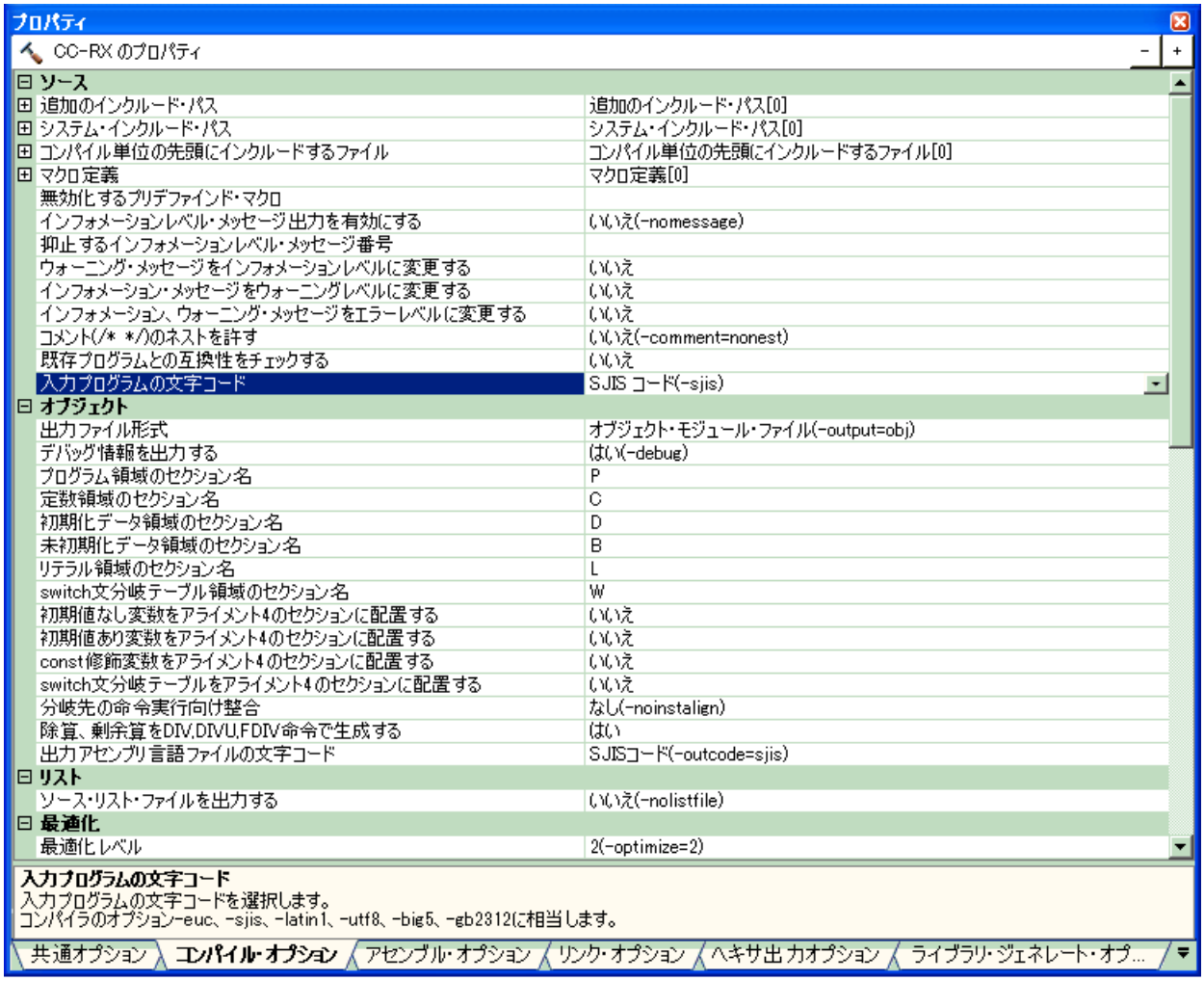
<table>
<tr><td>ビルド前に実行するコマンド</td><td>ビルド処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir%:アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName%:アクティブ・プロジェクト名に置換します。
%BuildModeName%:ビルド・モード名に置換します。
%MainProjectDir%:メイン・プロジェクト・フォルダの絶対パスに置換します。
%MainProjectName%: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>
<tr><td>ビルド後に実行するコマンド</td><td>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。
<table><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></table></td></tr>
</table>

その他の追加オプション	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、コンパイラのオプション群の最後に付加します。	
	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

［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) ［ソース］
ソー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	インクルード・ファイルの存在するパス名を指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 コンパイラのオプション -include 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。	
	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	247 文字までの文字列 65536 個まで指定可能です。
システム・インクルード・パス	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。 コンパイラのオプション -include に相当します。	
	デフォルト	システム・インクルード・パス [*定義数*]
	変更方法	[...] 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
	指定可能値	変更不可（インクルード・パスの設定順の変更のみ可能）
コンパイル単位の先頭にインクルードするファイル	コンパイル単位の先頭にインクルードするファイルを指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 コンパイラのオプション -preinclude に相当します。	
	デフォルト	コンパイル単位の先頭にインクルードするファイル [*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	259 文字までの文字列 65536 個まで指定可能です。

マクロ定義	定義したいマクロ名を指定します。 「*マクロ名* = *文字列*」の形式で 1 行に 1 つずつ指定します。 「= *文字列*」の部分は省略可能で，省略した場合，そのマクロ名が定義されたものと仮定します。 コンパイラのオプション -define 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	マクロ定義 [*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	32767 文字までの文字列 65536 個まで指定可能です。	
無効化するプリデファインド・マクロ	無効化するプリデファインド・マクロを指定します。 複数指定する場合は，マクロ名をカンマで区切って指定します（例：__DBL4,__SCHAR）。 コンパイラのオプション -undefine に相当します。		
	デフォルト	空欄	
	変更方法	[...] ボタンをクリックし，無効化するマクロの指定 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
インフォメーションレベル・メッセージ出力を有効にする	インフォメーションレベル・メッセージ出力を有効にするかどうかを選択します。 コンパイラのオプション -message，および -nomessage に相当します。		
	デフォルト	いいえ (-nomessage)	
	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 (-message)	インフォメーション・レベル・メッセージを有効にします。
		いいえ (-nomessage)	インフォメーション・レベル・メッセージを無効にします。
抑止するインフォメーションレベル・メッセージ番号	抑止するインフォメーション・レベル・メッセージ番号を指定します。 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：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -nomessage に相当します。 なお，本プロパティは，［インフォメーション・レベル・メッセージを抑止する］プロパティで［いいえ (-nomessage)］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	32767 文字までの文字列	

ウォーニング・メッセージをインフォメーションレベルに変更する	ウォーニング・メッセージをインフォメーションレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=information)	すべてのウォーニング・メッセージをインフォメーションレベルに変更します。
		はい (エラー番号指定) (-change_message=information=< エラー番号 >)	ウォーニングレベルの指定エラー番号のみインフォメーションレベルに変更します。
		いいえ	ウォーニング・メッセージをインフォメーションレベルに変更しません。
ウォーニングレベルのエラー番号	ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［ウォーニング・メッセージをインフォメーションレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=information=< エラー番号 >)］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	32767 文字までの文字列	
インフォメーション・メッセージをウォーニングレベルに変更する	インフォメーション・メッセージをウォーニングレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=warning)	すべてのインフォメーション・メッセージをウォーニングレベルに変更します。
		はい (エラー番号指定) (-change_message=warning=< エラー番号 >)	インフォメーションレベルの指定エラー番号のみウォーニングレベルに変更します。
		いいえ	インフォメーション・メッセージをウォーニングレベルに変更しません。

インフォメーションレベルのエラー番号	インフォメーションレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション・メッセージをウォーニングレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=warning=< エラー番号 >)］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	32767 文字までの文字列	
インフォメーション、ウォーニング・メッセージをエラーレベルに変更する	インフォメーション，ウォーニング・メッセージをエラーレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=error))	すべてのインフォメーション，ウォーニング・メッセージをエラーレベルに変更します。
		はい (エラー番号指定) (-change_message=error=< エラー番号 >)	インフォメーション，ウォーニングレベルの指定エラー番号のみエラーレベルに変更します。
		いいえ	インフォメーション，ウォーニング・メッセージをエラーレベルに変更しません。
インフォメーション、ウォーニングレベルのエラー番号	インフォメーション，ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション、ウォーニング・メッセージをエラーレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=error=< エラー番号 >)］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	32767 文字までの文字列	
コメント (/* */) のネストを許す	コメント (/* */) のネストを許すかどうかを選択します。 コンパイラのオプション -comment に相当します。		
	デフォルト	いいえ (-comment=nonest)	
	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 (-comment=nest)	コメント (/* */) のネストを許します。
		いいえ (-comment=nonest)	コメント (/* */) のネストを許しません。

既存プログラムとの互換性をチェックする	既存プログラムとの互換性をチェックするかどうかを選択します。 コンパイラのオプション -check 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 (NC コンパイラ) (-check=nc)	R8C,M16C ファミリ用 C コンパイラとの互換性をチェックします。
		はい (H8 コンパイラ) (-check=ch38)	H8,H8S,H8S ファミリ用 C/C++ コンパイラとの互換性をチェックします。
		はい (SH コンパイラ) (-check=sh)	SuperH ファミリ用 C/C++ コンパイラとの互換性をチェックします。
		いいえ	既存プログラムとの互換性をチェックしません。
入力プログラムの文字コード	入力プログラムの文字コードを選択します。 コンパイラのオプション -euc，-sjis，-latin1，-utf8，-big5，-gb2312 に相当します。		
	デフォルト	SJIS コード (-sjis)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	EUC コード (-euc)	文字列，文字定数，およびコメント内の文字を EUC で扱います。
		SJIS コード (-sjis)	文字列，文字定数，およびコメント内の文字を SJIS で扱います。
		ISO-Latin1 コード (-latin1)	文字列，文字定数，およびコメント内の文字を ISO-Latin1 で扱います。
		ISO-Latin1 コード (-latin1)	文字列，文字定数，およびコメント内の文字を ISO-Latin1 で扱います。
		繁体中国語 (-big5))	文字列，文字定数，およびコメント内の文字を繁体中国語で扱います。
		簡体中国語 (-gb2312)	文字列，文字定数，およびコメント内の文字を簡体中国語で扱います。

(2) ［オブジェクト］

オブジェク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。

出力ファイル形式	出力ファイル形式を選択します。 コンパイラのオプション -output に相当します。		
	デフォルト	オブジェクト・モジュール・ファイル (-output=obj)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	オブジェクト・モジュール・ファイル (-output=obj)	オブジェクト・モジュール・ファイルを出力します。
		プリプロセッサ展開後のソース・ファイル (-output=prep)	プリプロセッサ展開後のソースファイルを出力します。
		プリプロセッサ展開後のソース・ファイル (#line 出力抑止) (-output=prep -noline)	プリプロセッサ展開時に #line 出力を抑止します。

デバッグ情報を出力する	デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。コンパイラのオプション -debug, -nodebug に相当します。		
	デフォルト	はい (-debug)	
	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 (-debug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力します。
		いいえ (-nodebug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。
プログラム領域のセクション名	プログラム領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	P	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
定数領域のセクション名	定数領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	C	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
初期化データ領域のセクション名	初期化データ領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	D	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
未初期化データ領域のセクション名	未初期化データ領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	B	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
リテラル領域のセクション名	リテラル領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	L	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
switch 文分岐テーブル領域のセクション名	switch 文分岐テーブル領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	W	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	

初期値なし変数をアライメント 4 のセクションに配置する	初期値なし変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=B)	初期値なし変数をアライメント 4 のセクションに配置します。
		いいえ	初期値なし変数をアライメント 4 のセクションに配置しません。
初期値あり変数をアライメント 4 のセクションに配置する	初期値あり変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=D)	初期値あり変数をアライメント 4 のセクションに配置します。
		いいえ	初期値あり変数をアライメント 4 のセクションに配置しません。
const 修飾変数をアライメント 4 のセクションに配置する	const 修飾変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=C)	const 修飾変数をアライメント 4 のセクションに配置します。
		いいえ	const 修飾変数をアライメント 4 のセクションに配置しません。
switch 文分岐テーブルをアライメント 4 のセクションに配置する	switch 文分岐テーブルをアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=W)	switch 文分岐テーブルをアライメント 4 のセクションに配置します。
		いいえ	switch 文分岐テーブルをアライメント 4 のセクションに配置しません。

分岐先の命令実行向け整合	分岐先の命令実行向け整合を選択します。 コンパイラのオプション -noinstalign , -instalign4, -instalign8 に相当します。		
	デフォルト	なし (-noinstalign)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	なし (-noinstalign)	分岐先を命令実行向け整合をしません。
		4 バイト整合 (-instalign4)	分岐先を 4 バイトで命令実行向け整合します。
		4 バイト整合 (各ループの先頭含む) (-instalign4=loop)	分岐先を 4 バイトで命令実行向け整合します (各ループの先頭含む)。
		4 バイト整合 (各最内周ループの先頭含む) (-instalign4=inmostloop)	分岐先を 4 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
		8 バイト整合 (-instalign8)	分岐先を 8 バイトで命令実行向け整合します。
		8 バイト整合 (各ループの先頭含む) (-instalign8=loop)	分岐先を 8 バイトで命令実行向け整合します (各ループの先頭含む)。
		8 バイト整合 (各最内周ループの先頭含む) (-instalign8=inmostloop)	分岐先を 8 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
除算、剰余算を DIV,DIVU,FDIV 命令で生成する	除算，剰余算を DIV,DIVU,FDIV 命令で生成するかどうかを選択します。 コンパイラのオプション -nouse_div_inst 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	除算，剰余算に DIV,DIVU,FDIV 命令を使ったコードを生成します。
		いいえ (-nouse_div_inst)	除算，剰余算に DIV,DIVU,FDIV 命令を使わないコードを生成します。

出力アセンブリ言語ファイルの文字コード	出力アセンブリ言語ファイルの文字コードを選択します。 コンパイラのオプション -outcode に相当します。		
	デフォルト	SJIS コード (-outcode=sjis)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	EUC コード (-outcode=euc)	文字列，文字定数内の文字を EUC で出力します。
		SJIS コード (-outcode=sjis)	文字列，文字定数内の文字を SJIS で出力します。
		UTF-8 コード (-outcode=utf8)	文字列，文字定数内の文字を UTF-8 で出力します。 なお，本項目は，［ソース］カテゴリの［C ソース・ファイルの言語］プロパティで［C(C89) (-lang=c)］を選択した場合は選択できません。
		繁体中国語 (-outcode=big5)	文字列，文字定数内の文字を繁体中国語で出力します。
		簡体中国語 (-outcode=gn2312)	文字列，文字定数内の文字を簡体中国語で出力します。

(3) ［リスト］

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。

ソース・リスト・ファイルを出力する	ソース・リスト・ファイルを出力するかどうかを選択します。 コンパイラのオプション -listfile ， -nolistfile 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 (-listfile)	ソース・リスト・ファイルを出力します。
		いいえ (-nolistfile)	ソース・リスト・ファイルを出力しません。
C/C++ ソースを出力する	ソース・リスト・ファイルの内容の設定を行います。 C/C++ ソースを出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=source)	C/C++ ソースを出力します。
		いいえ	C/C++ ソースを出力しません。

<table>
<tr><td rowspan="5">条件アセンブルで偽の行を出力する</td><td colspan="3">- ソース・リスト・ファイルの内容の設定を行います。
- 条件アセンブルで偽の行を出力するかどうかを選択します。
- コンパイラのオプション -show に相当します。
- なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=conditionals)</td><td>- 条件アセンブルで偽の行を出力します。</td></tr>
<tr><td>- いいえ</td><td>- 条件アセンブルで偽の行を出力しません。</td></tr>
<tr><td rowspan="5">.DEFINE 置換前の情報を出力する</td><td colspan="3">- ソース・リスト・ファイルの内容の設定を行います。
- .DEFINE 置換前の情報を出力するかどうかを選択します。
- コンパイラのオプション -show に相当します。
- なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=definitions)</td><td>- .DEFINE 置換前の情報を出力します。</td></tr>
<tr><td>- いいえ</td><td>- .DEFINE 置換前の情報を出力しません。</td></tr>
<tr><td rowspan="5">アセンブラ・マクロ記述展開行を出力する</td><td colspan="3">- ソース・リスト・ファイルの内容の設定を行います。
- アセンブラ・マクロ記述展開行を出力するかどうかを選択します。
- コンパイラのオプション -show に相当します。
- なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=expansions)</td><td>- アセンブラ・マクロ記述展開行を出力します。</td></tr>
<tr><td>- いいえ</td><td>- アセンブラ・マクロ記述展開行を出力しません。</td></tr>
</table>

(4) ［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	最適化レベルを選択します。 コンパイラのオプション -optimize に相当します。		
	デフォルト	2 (-optimize=2)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	0 (-optimize=0)	最適化を実施しません。
		1 (-optimize=1)	自動変数のレジスタ割り付け，関数出口ブロックの統合，統合可能な複数命令の統合など，一部最適化を実施します。
		2 (-optimize=2)	全般的に最適化を実施します。
		Max (-optimize=max)	実施可能な最適化を最大限に行います。
モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 コンパイラのオプション -goptimize 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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。
最適化方法	最適化方法を選択します。 コンパイラのオプション -speed ， -size に相当します。		
	デフォルト	コード・サイズ重視の最適化 (-size)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	実行性能重視の最適化 (-speed)	実行性能重視の最適化を実施します。
		コード・サイズ重視の最適化 (-size)	コードサイズ重視の最適化を実施します。
ループ展開	ループ文 (for, while, do-while) を展開するかどうかを選択します。 コンパイラのオプション -loop に相当します。		
	デフォルト	最適化レベル，最適化方法オプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		展開する (-loop=< 数値 >)	ループ文 (for, while, do-while) を展開します。

最大展開数	最大で何倍の展開を行うかを指定します。 コンパイラのオプション -loop に相当します。 なお，本プロパティは，［ループ展開］プロパティで［展開する (-loop=< 数値 >)］を選択した場合のみ表示します。		
	デフォルト	2（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	1 ～ 32（10 進数）	
自動インライン展開を行う	自動インライン展開を行うかどうかを選択します。 コンパイラのオプション -inline，-noinline に相当します。		
	デフォルト	最適化レベル，最適化方法オプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		はい (-inline=< 数値 >)	自動インライン展開を行います。
		いいえ (-noinline)	自動インライン展開を行いません。
関数サイズの最大増加率	関数サイズの最大増加率を指定します。 たとえば，100 を指定した場合，関数サイズが 100% 増加するまで (2 倍まで) インライン展開を行います。 コンパイラのオプション -inline に相当します。 なお，本プロパティは，［自動インライン展開を行う］プロパティで［はい (-inline=< 数値 >)］を選択した場合のみ表示します。		
	デフォルト	100（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 ～ 65535（10 進数）	
switch 文のコード展開方式	switch 文のコード展開方式を選択します。 コンパイラのオプション -case に相当します。		
	デフォルト	コンパイラが自動選択 (-case=auto)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	if_then 方式 (-case=ifthen)	switch 文を if_then 方式で展開します。
		テーブル・ジャンプ方式 (-case=table)	switch 文をテーブル方式で展開します。
		コンパイラが自動選択 (-case=auto)	if_then 方式，テーブル方式いずれかをコンパイラが自動的に選択します。
外部変数を volatile 化する	すべての外部変数を volatile 宣言したものとして扱うかどうかを選択します。 コンパイラのオプション -volatile，-novolatile に相当します。		
	デフォルト	いいえ (-novolatile)	
	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 (-volatile)	すべての外部変数を volatile 宣言したものとして扱います。
		いいえ (-novolatile)	volatile 修飾のない外部変数に対して最適化を行います。

const 宣言された外部変数の定数伝播を実施する	const 宣言された外部変数の定数伝播を実施するかどうかを選択します。 C++ 言語ソースファイルの const 修飾型変数については，本オプションで制御することはできません (常に定数伝播されます)。 コンパイラのオプション -const_copy，-noconst_copy に相当します。		
	デフォルト	最適化レベルオプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-const_copy)	const 修飾型外部変数についても定数伝播を行います。
		いいえ (-noconst_copy)	const 修飾型外部変数の定数伝播を抑止します。
整数型定数による除算および剰余算の変換方法	整数型定数による除算および剰余算の変換方法を選択します。 コンパイラのオプション -const_div，-noconst_div に相当します。		
	デフォルト	最適化方法オプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化方法オプションに依存する	最適化方法オプションに従います。
		乗算を用いた命令列に変換 (-const_div)	ソースファイル中の整数型定数による除算および剰余算を，乗算を用いた命令列に変換します。
		除算を用いた命令列に変換 (-noconst_div)	ソースファイル中の整数型定数による除算および剰余算を，除算を用いた命令列に変換します。
ライブラリ関数の展開方法	ライブラリ関数の展開方法を選択します。 コンパイラのオプション -library に相当します。		
	デフォルト	一部のライブラリ関数を命令展開 (-library=intrinsic)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	すべて関数呼び出し (-library=function)	ライブラリ関数をすべて関数呼び出しします。
		一部のライブラリ関数を命令展開 (-library=intrinsic)	abs()，absf() およびストリング操作命令が使用できるライブラリ関数を命令展開します。
最適化範囲を複数に分割してコンパイルする	サイズの大きい関数について，最適化範囲を複数に分割してコンパイルするかどうかを指定します。 コンパイラのオプション -scope，-noscope に相当します。		
	デフォルト	最適化レベルオプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-scope)	コンパイルの前にサイズの大きい関数について，最適化範囲を複数に分割します。
		いいえ (-noscope)	コンパイルの前に最適化範囲を分割しません。

<table>
<tr><td rowspan="5">パイプライン処理を考慮した命令並べ替えを行う</td><td colspan="3">パイプライン処理を考慮した命令並べ替えを行うかどうかを選択します。
コンパイラのオプション -schedule，-noschedule 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="3">指定可能値</td><td>最適化レベルオプションに依存する</td><td>最適化レベルオプションに従います。</td></tr>
<tr><td>はい (-schedule)</td><td>パイプライン処理を考慮した命令並べ替えを行います。</td></tr>
<tr><td>いいえ (-noschedule)</td><td>命令並べ替えを行いません。</td></tr>
<tr><td rowspan="5">外部変数アクセス最適化を行う</td><td colspan="3">外部変数アクセス最適化を行うかどうかを選択します。
コンパイラのオプション -nomap，-smap，-map に相当します。
ライブラリ・プロジェクトの場合、選択肢「はい (モジュール間で最適化)(-map)」は表示されません。</td></tr>
<tr><td>デフォルト</td><td colspan="2">［最適化レベル］プロパティで，［Max (-optimize=max)］を選択した場合
はい (モジュール間で最適化) (-map)
上記以外の場合
いいえ (-nomap)</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>はい (モジュール内で最適化) (-smap)</td><td>コンパイル対象ファイル内で定義された外部変数もしくは静的変数についてベースアドレスを設定し，アクセスをベースアドレス相対で行うコードを生成します。</td></tr>
<tr><td>はい (モジュール間で最適化) (-map)</td><td>最適化リンケージエディタが生成する外部シンボル割り付け情報を元にベースアドレスを設定し，外部変数もしくは静的変数のアクセスをベースアドレス相対で行うコードを生成します。</td></tr>
<tr><td>いいえ (-nomap)</td><td>外部変数アクセス最適化を行いません。</td></tr>
</table>

<table>
<tr><td rowspan="5">大域最適化を行う</td><td colspan="3">大域最適化 (関数の統合など) を行うレベルを指定します。
［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］を選択した場合は［はい (レベル 1)(最適化を行う (-ip_optimize))］、および［いいえ］のみ表示します。
コンパイラのオプション -whole_program、-merge_files、-ip_optimize 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="3">指定可能値</td><td>はい (レベル 3)(入力ソースがプログラム全体として最適化する)(-whole_program)</td><td>ソース・ファイルがプログラム全体であることを仮定して、大域最適化を行います。
ただし、前提条件を満たさなかった場合の動作は保証しません。
前提条件の詳細については、[-whole_program］を参照してください。</td></tr>
<tr><td>はい (レベル 2)(複数ファイルをマージして最適化する)(-merge_files, -ip_optimize)</td><td>複数の C ソース・ファイルをマージした上で、大域最適化を行います。</td></tr>
<tr><td>はい (レベル 1)(最適化を行う)(-ip_optimize)</td><td>ソースファイル毎に、大域最適化を行います。</td></tr>
<tr><td rowspan="4">浮動小数点定数除算の乗算化を行う</td><td colspan="3">浮動小数点定数除算を，定数の逆数の乗算に変換するかどうかを選択します。
コンパイラのオプション -approxdiv 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>指定可能値</td><td>はい (-approxdiv)</td><td>浮動小数点定数除算を，定数の逆数の乗算に変換します。</td></tr>
<tr><td></td><td>いいえ</td><td>浮動小数点定数除算を，定数の逆数の乗算に変換しません。</td></tr>
<tr><td rowspan="4">浮動小数点型 <-> 符号無し整数型の範囲チェックを省略する</td><td colspan="3">浮動小数点型 <-> 符号無し整数型の範囲チェックを省略するかどうかを選択します。
“ はい ” を選択した時，該当する型変換の処理に対するコード性能は向上しますが，変換結果が C,C++ 言語規格と異なる場合がありますので，ご注意ください。
コンパイラのオプション -simple_float_conv 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>指定可能値</td><td>はい (-simple_float_conv)</td><td>浮動小数点型の型変換処理の一部を省略します。</td></tr>
<tr><td></td><td>いいえ</td><td>浮動小数点型の型変換処理の一部を省略しません。</td></tr>
</table>

ポインタ指示先の型を考慮した最適化を実施する	ポインタ指示先の型を考慮した最適化を実施するかどうかを選択します。 “ はい ” を指定した場合，一般には，alias=noansi を指定した場合よりもオブジェクト性能が向上しますが，alias=ansi と alias=noansi とで実行結果が異なる場合があります。 コンパイラのオプション -alias に相当します。		
	デフォルト	いいえ (-alias=noansi)	
	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 (-alias=ansi)	ANSI 規格に基づき，ポインタ指示先の型を考慮した最適化を行います。
		いいえ (-alias=noansi)	ANSI 規格に基づくポインタ指示先の型を考慮した最適化を行いません。
浮動小数点式の演算順序変更の最適化を行う	浮動小数点演算式の演算順序変更の最適化を行うかどうかを選択します。 “ はい ” を指定した場合，一般には，-float_order を指定しない場合よりもオブジェクト性能が向上しますが，演算の精度が -float_order を指定しなかった場合と異なることがあります。 コンパイラのオプション -float_order に相当します。 また，本プロパティは，［最適化レベル］プロパティが［2 (-optimize=2)］または［Max (-optimize=max)］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 (-float_order)	浮動小数点演算式の演算順序変更の最適化を行います。
		いいえ	浮動小数点演算式の演算順序変更の最適化を行いません。

(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。

<table>
<tr><td rowspan="5">アセンブリ・ソース・ファイルを出力する</td><td colspan="3">C ソースのコンパイル結果のアセンブリ・ソース・ファイルを出力するかどうかを選択します。
コンパイラのオプション -output=src 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 (-output=src)</td><td>C ソースのコンパイル結果のアセンブリ・ソース・ファイルを出力します。</td></tr>
<tr><td>いいえ</td><td>C ソースのコンパイル結果のアセンブリ・ソース・ファイルを出力しません。</td></tr>
<tr><td rowspan="5">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。
コンパイラのオプション -output，-noline 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="3">指定可能値</td><td>はい (-output=prep)</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>はい (#line 出力抑止)(-output=prep -noline)</td><td>ソースファイルに対して、プリプロセス処理 (#line 出力抑止) を実行した結果をファイルに出力します。</td></tr>
<tr><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>

(6) ［MISRA C ルール検査］

MISRA C ルール検査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="11">適用するルール</td><td colspan="3">適用する MISRA C ルールを選択します。
コンパイラのオプション -misra2004 に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">適用ルールなし</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="8">指定可能値</td><td>すべてのルールを適用 (-misra2004=all)</td><td>サポートしているすべてのルールをチェック対象とします。</td></tr>
<tr><td>指定したルール番号を適用 (-misra2004=apply)</td><td>サポートしているルールのうち，指定されたルール番号をチェック対象とします。</td></tr>
<tr><td>指定したルール番号を除外 (-misra2004=ignore)</td><td>サポートしているルールのうち，指定されたルール番号以外のルールをチェック対象とします。</td></tr>
<tr><td>必須ルールを適用 (-misra2004=required)</td><td>サポートしているルールのうち，ルールの分類が “required” になっているルールをチェック対象とします。</td></tr>
<tr><td>必須ルールと指定したルールを適用 (-misra2004=required_add)</td><td>サポートしているルールのうち，ルールの分類が “required” になっているルールと指定されたルール番号をチェック対象とします。</td></tr>
<tr><td>必須ルールから指定したルール番号を除外 (-misra2004=required_remove)</td><td>サポートしているルールのうち，ルールの分類が “required” になっているルールから指定されたルール番号を除いたルール番号をチェック対象とします。</td></tr>
<tr><td>指定ファイルに記載されたルール番号を適用 (-misra2004=< ファイル名 >)</td><td>サポートしているルールのうち，指定されたファイル名に記載されたルール番号をチェック対象とします。</td></tr>
<tr><td>適用ルールなし</td><td>MISRA C ルールを適用しません。</td></tr>
<tr><td rowspan="4">ルール番号記載ファイル</td><td colspan="3">ルール番号記載ファイル (misra2004 ルール・ファイル) を指定します。
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。
%ProjectName%: 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。
コンパイラのオプション -misra2004 に相当します。
なお，本プロパティは，［適用するルール］プロパティで［指定ファイルに記載されたルール番号を適用 (-misra2004=< ファイル名 >)］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，または［...］ボタンをクリックし，Misra2004 ルール・ファイルの指定 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td colspan="2">259 文字までの文字列</td></tr>
</table>

ルール番号	ルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［適用するルール］プロパティで［指定したルール番号を適用 (-misra2004=apply］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
除外するルール番号	除外するルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［適用するルール］プロパティで［指定したルール番号を除外 (-misra2004=ignore)］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
必須ルールの他にチェックするルール番号	必須ルールの他にチェックするルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［必須ルールと指定したルールを適用 (-misra2004=required_add)］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
必須ルールから除外するルール番号	必須ルールから除外したいルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［適用するルール］プロパティで［必須ルールから指定したルール番号を除外 (-misra2004=required_remove)］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列

ルール・チェック対象外のファイル	MISRA-C2004 のルール・チェック対象外のファイルを指定します。 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。 %ProjectName%: 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。 コンパイラのオプション -ignore_files_misra に相当します。 なお，本プロパティは，［適用するルール］プロパティで［適用ルールなし］を選択した場合は表示されません。		
	デフォルト	空欄	
	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	259 文字までの文字列 65536 個まで指定可能	
拡張キーワードや拡張仕様をメッセージ出力する	拡張キーワードや拡張仕様をメッセージ出力するかどうかを選択します。 コンパイラのオプション -check_language_extension に相当します。 なお，本プロパティは，［適用するルール］プロパティで［適用ルールなし］を選択した場合は表示されません。		
	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 (-check_language_extension)	C/C++ 言語規格から独自に拡張した言語仕様が有効な場合に一部抑止される，MISRA2004 ルールチェックを有効にします。
		いいえ	C/C++ 言語規格から独自に拡張した言語仕様が有効な場合に一部抑止される，MISRA2004 ルールチェックを無効にします。

(7) ［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。

コピーライトを出力する	コピーライトを出力するかどうかを選択します。 コンパイラのオプション -nologo に相当します。		
	デフォルト	いいえ (-nologo)	
	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 (-logo)	コピーライトを出力します。
		いいえ (-nologo)	コピーライトの出力を抑止します。
クロス・リファレンス情報を出力する	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。 本オプションを変更するには，「プログラム解析」のプロパティの設定を変更する必要があります。		
	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	クロス・リファレンス情報を出力します。
		いいえ	クロス・リファレンス情報を出力しません。

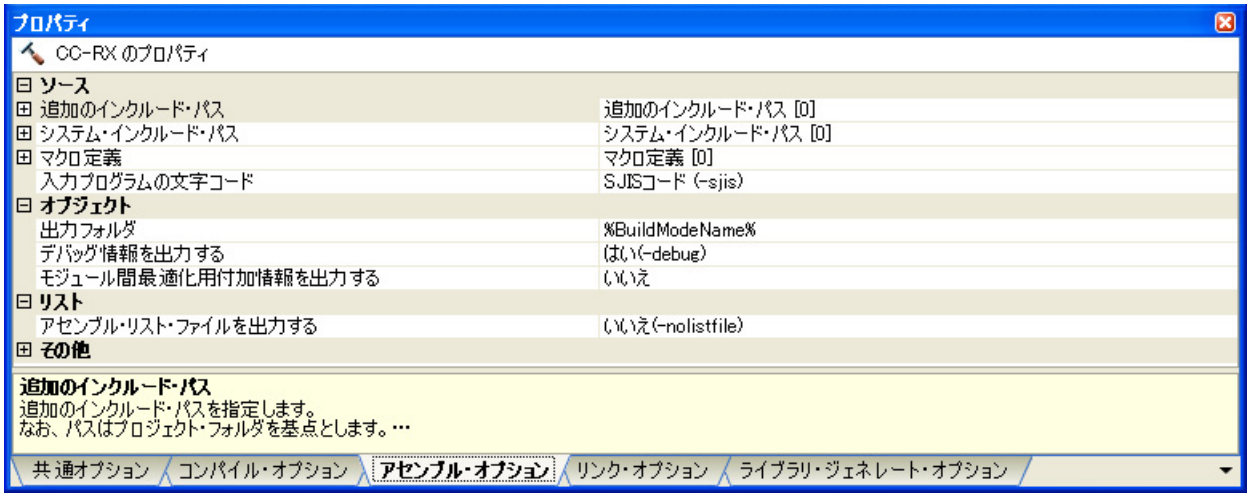
<table>
<tr><td>コンパイル前に実行するコマンド</td><td>コンパイル処理前に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir%:アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName%:アクティブ・プロジェクト名に置換します。
%BuildModeName%:ビルド・モード名に置換します。
%CompiledFile%:コンパイル時の出力ファイルの絶対パスに置換します。
%InputFile% ：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。
%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・ファイル名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，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>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。
%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・ファイル名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，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></td><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></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>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td>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。
なお，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されます。</td></tr>
<tr><td></td><td>デフォルト</td><td>空欄</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，文字列入力 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>259 文字までの文字列</td></tr>
<tr><td>コマンド・ライン</td><td colspan="2">指定されているオプションを表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>コマンド・ライン [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>変更不可</td></tr>
</table>

［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。

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) ［ソース］

ソー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	インクルード・ファイルの存在するパス名を指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 アセンブラのオプション -include 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。	
	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	247 文字までの文字列 65536 個まで指定可能です。

システム・インクルード・パス	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。 アセンブラのオプション -include に相当します。		
	デフォルト	システム・インクルード・パス [*定義数*]	
	変更方法	[...] 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	
	指定可能値	変更不可（インクルード・パスの設定順の変更のみ可能）	
マクロ定義	定義したいマクロ名を指定します。 「*マクロ名* = *文字列*」の形式で 1 行に 1 つずつ指定します。 アセンブラのオプション -define 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	マクロ定義 [*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	32767 文字までの文字列 65536 個まで指定可能です。	
入力プログラムの文字コード	入力プログラムの文字コードを選択します。 アセンブラのオプション -euc，-sjis，-latin1，-big5，-gb2312 に相当します。		
	デフォルト	SJIS コード (-sjis)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	EUC コード (-euc)	文字列，文字定数，およびコメント内の文字を EUC で扱います。
		SJIS コード (-sjis)	文字列，文字定数，およびコメント内の文字を SJIS で扱います。
		ISO-Latin1 コード (-latin1)	文字列，文字定数，およびコメント内の文字を ISO-Latin1 で扱います。
		繁体中国語 (-big5)	文字列，文字定数，およびコメント内の文字を繁体中国語で扱います。
		簡体中国語 (-gb2312)	文字列，文字定数，およびコメント内の文字を簡体中国語で扱います。

(2) ［オブジェクト］

オブジェク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。
なお、本カテゴリは、［共通オプション］タブの［ビルド方法］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。

出力フォルダ	出力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。 %BuildModeName%: ビルド・モード名に置換します。 %ProjectName%: 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。 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。 アセンブラのオプション -output 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］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。	
	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	

デバッグ情報を出力する	デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。アセンブラのオプション -debug，-nodebug 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］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。		
	デフォルト	はい (-debug)	
	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 (-debug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力します。
		いいえ (-nodebug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。

(3) ［リスト］

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。
なお、本カテゴリは、［共通オプション］タブの［ビルド方法］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。

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。 アセンブラのオプション -listfile，-nolistfile 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］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。		
	デフォルト	いいえ (-nolistfile)	
	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 (-listfile)	アセンブル・リスト・ファイルを出力します。
		いいえ (-nolistfile)	アセンブル・リスト・ファイルを出力しません。
条件アセンブルで偽の行を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 条件アセンブルで偽の行を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］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 (-show=conditionals)	条件アセンブルで条件が偽となった行を出力します。
		いいえ	条件アセンブルで条件が偽となった行を出力しません。
.DEFINE 置換前の情報を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 .DEFINE 置換前の情報を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］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 (-show=definitions)	.DEFINE 置換前の情報を出力します。
		いいえ	.DEFINE 置換前の情報を出力しません。

アセンブラ・マクロ記述展開行を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 アセンブラ・マクロ記述展開行を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］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 (-show=expansions)	アセンブラ・マクロ記述展開行を出力します。
		いいえ	アセンブラ・マクロ記述展開行を出力しません。

(4) ［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。
なお、本カテゴリは、［共通オプション］タブの［ビルド方法］の [一括ビルドを行う] プロパティで [いいえ] を選択した場合に表示します。

モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 アセンブラのオプション -goptimize 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］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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。

(5) ［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。

特権命令をチェックする	特権命令をチェックするかどうかを選択します。 アセンブラのオプション -chkpm 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 (-chkpm)	特権命令をチェックします。
		いいえ	特権命令をチェックしません。
浮動小数点演算命令をチェックする	浮動小数点演算命令をチェックするかどうかを選択します。 アセンブラのオプション -chkfpu 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 (-chkfpu)	浮動小数点演算命令をチェックします。
		いいえ	浮動小数点演算命令をチェックしません。

<table>
<tr><td rowspan="4">DSP 機能命令をチェックする</td><td colspan="3">DSP 機能命令をチェックするかどうかを選択します。
アセンブラのオプション -chkdsp 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>指定可能値</td><td>はい (-chkdsp)</td><td>DSP 機能命令をチェックします。</td></tr>
<tr><td></td><td>いいえ</td><td>DSP 機能命令をチェックしません。</td></tr>
<tr><td rowspan="4">コピーライトを出力する</td><td colspan="3">コピーライトを出力するかどうかを選択します。
アセンブラのオプション -logo，-nologo に相当します。</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ (-nologo)</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 (-logo)</td><td>コピーライトを出力します。</td></tr>
<tr><td></td><td>いいえ (-nologo)</td><td>コピーライトの出力を抑止します。</td></tr>
<tr><td rowspan="4">アセンブル前に実行するコマンド</td><td colspan="3">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。
%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・ファイル名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，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。</td></tr>
<tr><td>デフォルト</td><td colspan="2">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 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>指定可能値</td><td colspan="2">1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>

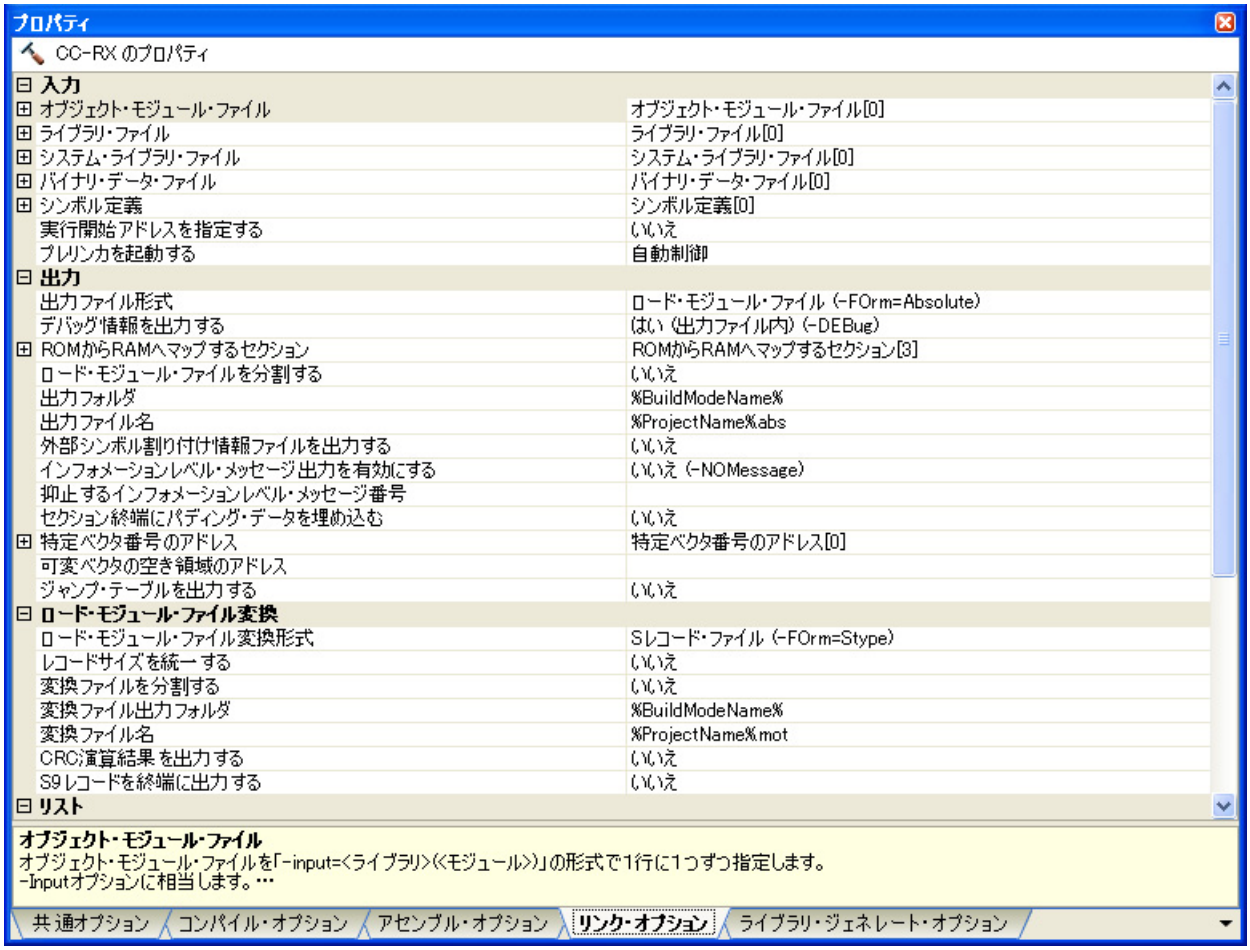
<table>
<tr><td>アセンブル後に実行するコマンド</td><td colspan="2">アセンブル処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir%:アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName%:アクティブ・プロジェクト名に置換します。
%AssembledFile%:アセンブル時の出力ファイルの絶対パスに置換します。
%BuildModeName%:ビルド・モード名に置換します。
%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。
%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・ファイル名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，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></td><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></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>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td>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ラの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，アセンブラのオプション群の最後に付加されます。</td></tr>
<tr><td></td><td>デフォルト</td><td>空欄</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，文字列入力 ダイアログによる編集</td></tr>
<tr><td></td><td>指定可能値</td><td>259 文字までの文字列</td></tr>
<tr><td>コマンド・ライン</td><td colspan="2">指定されているオプションを表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>コマンド・ライン [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td>変更不可</td></tr>
</table>

［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。

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) [入力]

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。

オブジェクト・モジュール・ファイル	オブジェクト・モジュール・ファイルを指定します。 1 行に 1 ファイルずつ指定します。 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。 %ProjectName%: 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。 リンカのオプション -Input に相当します。 指定した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・モジュール・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	32767 文字までの文字列 65536 個まで指定可能です。
使用するライブラリ・ファイル	ライブラリ・ファイルを指定します。 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。 %ProjectName%: 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。 リンカのオプション -library 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	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 65536 個まで指定可能です。
システム・ライブラリ・ファイル	リンク時にシステムが設定するシステム・ライブラリ・ファイルを表示します。 リンカのオプション -library に相当します。	
	デフォルト	システム・ライブラリ・ファイル [*定義数*]
	指定可能値	変更不可
バイナリ・データ・ファイル	バイナリ・データ・ファイルを指定します。 「*ファイル名* (*セクション名* : *アライメント数* / *セクション属性* , *シンボル名*)」の形式で 1 行に 1 つずつ指定します。 「: *アライメント数*」，「/ *セクション属性*」，「, *シンボル名*」の部分は省略可能です。 「*アライメント数*」は，1，2，4，8，16，または 32 になります。 省略した場合は，定義値を 1 とします。 「*セクション属性*」は，CODE または DATA になります。 省略した場合は，書き込み，読み取り，実行，すべての属性が有効になります。 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。 %ProjectName%: 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。 リンカのオプション -binary に相当します。 バイナリ・データ・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	バイナリ・データ・ファイル指定 [*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	32767 文字までの文字列 65536 個まで指定可能です。

シンボル定義	シンボルを定義します。 「*シンボル名 = シンボル名*」，または「*シンボル名 = 数値*」の形式で 1 行に 1 つずつ指定します。 数値は 16 進数で指定します。 リンカのオプション -define に相当します。		
	デフォルト	シンボル定義 [*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	32767 文字までの文字列 65536 個まで指定可能です。	
実行開始アドレスを指定する	実行開始アドレスを外部定義シンボルまたはアドレスで指定するかどうかを選択します。リンカのオプション -entry 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 (-ENTry)	実行開始アドレスを外部定義シンボルまたはアドレスで指定します。
		いいえ	実行開始アドレスを外部定義シンボルまたはアドレスで指定しません。
実行開始アドレス	実行開始アドレスを指定します。 「*シンボル名*」または「*アドレス*」の形式で指定します。 アドレスは 16 進数で指定します。 リンカのオプション -entry に相当します。 なお，本プロパティは，［実行開始アドレスを指定する］プロパティで［はい (-ENTry)］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	32767 文字までの文字列	
プレリンカを起動する	プレリンカ（C++ テンプレート・インスタンスの自動生成）を起動するかどうかを選択します。 リンカのオプション -noprelink に相当します。		
	デフォルト	自動制御	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	自動制御	リンカの入力ファイルに ii ファイルがなければ，プレリンカの起動を抑止します。
		はい	プレリンカを起動します。
		いいえ (-NOPRElink)	プレリンカの起動を抑止します。

(2) ［出力］

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。

出力ファイル形式	出力ファイル形式を表示します。 リンカのオプション -form に相当します。	
	デフォルト	ロード・モジュール・ファイル (-FOrm=Absolute)
	変更方法	変更不可

デバッグ情報を出力する	デバッグ情報を出力するかどうかを選択します。 リンカのオプション -debug，-sdebug，および -nodebug に相当します。		
	デフォルト	はい (出力ファイル内) (-DEBug)	
	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 (出力ファイル内) (-DEBug)	デバッグ情報を出力します（出力ファイル内）。
		はい (デバッグ情報ファイル出力) (-SDebug)	デバッグ情報ファイルを出力します。
		いいえ (-NODEBug)	デバッグ情報を出力しません。
ロード・モジュール・ファイルを分割する	ロード・モジュール・ファイルを分割するかどうかを選択します。 リンカのオプション -output 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	ロード・モジュール・ファイルを分割します。
		いいえ	ロード・モジュール・ファイルを分割しません。
出力フォルダ	出力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。 %BuildModeName%: ビルド・モード名に置換します。 %ProjectName%: 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。 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。 リンカのオプション -output 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	
出力ファイル名	出力ファイル名を指定します。 拡張子を省略した場合は，.abs 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。 %ProjectName%: プロジェクト名に置換します。 空欄の場合は，%ProjectName%.abs を指定したものとみなします。 リンカのオプション -output に相当します。		
	デフォルト	%ProjectName%.abs	
	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	

外部シンボル割り付け情報ファイルを出力する	外部シンボル割り付け情報ファイルを出力するかどうかを選択します。 リンカのオプション -map に相当します。		
	デフォルト	［コンパイル・オプション］タブの［最適化］カテゴリの［外部変数アクセス最適化を行う］プロパティで，［はい (モジュール間で最適化) (-map)］を選択した場合 はい (-Map) 上記以外の場合 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 (-Map)	外部シンボル割り付け情報ファイルを出力します。
		いいえ	外部シンボル割り付け情報ファイルを出力しません。
インフォメーションレベル・メッセージ出力を有効にする	インフォメーションレベル・メッセージの出力を有効にするかどうかを選択します。 リンカのオプション -message，-msg_unused，-nomessage に相当します。		
	デフォルト	いいえ (-NOMessage)	
	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 (-Message)	インフォメーション・レベル・メッセージを出力します。
		はい (参照されない定義シンボルの通知) (-Message -MSg_unused)	参照されない定義シンボルを通知します。
		いいえ (-NOMessage)	インフォメーション・レベル・メッセージを抑止します。
抑止するインフォメーションレベル・メッセージ番号	抑止するインフォメーションレベル・メッセージ番号を指定します。 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：4,200）。 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：4,200-203,1300）。 リンカのオプション -nomessage に相当します。 なお，本プロパティは，［インフォメーションレベル・メッセージ出力を有効にする］プロパティで［いいえ (-NOMessage)］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	32767 文字までの文字列	
セクション終端にパディング・データを埋め込む	セクション終端にパディング・データを埋め込むかどうかを選択します。 リンカのオプション -padding 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 (-PADDING)	セクション終端にパディング・データを埋め込みます。
		いいえ	セクション終端にパディング・データを埋め込みません。

特定ベクタ番号のアドレス	特定ベクタ番号のアドレスを指定します。 「*ベクタ番号* = *シンボル*」または「*ベクタ番号* = *アドレス*」の形式で 1 行に 1 つずつ指定します。 「*ベクタ番号*」は，10 進数で 0 ～ 255 の範囲で指定します。 「*シンボル*」は，対象関数の外部名で指定します。 アドレスは 16 進数で指定します。 リンカのオプション -vectn に相当します。		
	デフォルト	特定ベクタ番号のアドレス [*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	32767 文字までの文字列 65536 個まで指定可能です。	
可変ベクタの空き領域のアドレス	特定のベクタ番号の空き領域のアドレスを指定します。 「*シンボル*」または「*アドレス*」の形式で指定します。 「*シンボル*」は，対象関数の外部名で指定します。 アドレスは 16 進数で指定します。 リンカのオプション -vect 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	32767 文字までの文字列	
ジャンプ・テーブルを出力する	ジャンプ・テーブルを出力するかどうかを選択します。 リンカのオプション -jump_entries_for_pic に相当します。		
	デフォルト	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 (-JUMP_ENTRIES_FOR_PIC)	ジャンプ・テーブルを出力します。
		いいえ	ジャンプ・テーブルを出力しません。
外部定義シンボルへ分岐するジャンプ・テーブルを出力するセクション	外部定義シンボルへ分岐するジャンプ・テーブルを出力するセクションを指定します。 「*セクション名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -jump_entries_for_pic に相当します。 なお，本プロパティは，［ジャンプ・テーブルを出力する］プロパティで［はい (-JUMP_ENTRIES_FOR_PIC)］を選択した場合のみ表示します。		
	デフォルト	外部定義シンボルへ分岐するジャンプテーブルを出力するセクション［*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	32767 文字までの文字列 65536 個まで指定可能です。	

(3) ［リスト］

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。

リンケージ・リスト・ファイルを出力する	リンケージ・リスト・ファイルを出力するかどうかを選択します。 リンカのオプション -list，-show に相当します。		
	デフォルト	はい (リスト内容 = 選択) (-LISt)	
	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 (リスト内容 = 指定なし) (-LISt -SHow)	リンケージ・リスト・ファイルに出力形式に従った情報を出力します。
		はい (リスト内容 = すべて) (-LISt -SHow=ALL)	リンケージ・リスト・ファイルに出力形式に従ったすべての情報を出力します。
		はい (リスト内容 = 選択) (-LISt)	リンケージ・リスト・ファイルに指定した情報を出力します。
		いいえ	リンケージ・リスト・ファイルを出力しません。
シンボル情報を出力する	シンボル情報を出力するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］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 (-SHow=SYmbol)	シンボル情報を出力します。
		いいえ	シンボル情報を出力しません。
シンボルの参照回数を出力する	シンボルの参照回数を出力するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］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 (-SHow=Reference)	シンボルの参照回数を出力します。
		いいえ	シンボルの参照回数を出力しません。
クロスリファレンス情報を出力する	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。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］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 (-SHow=Xreference)	クロスリファレンス情報を出力します。
		いいえ	クロスリファレンス情報を出力しません。

セクションの合計サイズを表示する	セクションの合計サイズを表示するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］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 (-SHow=Total_size)	ROM 配置対象，RAM 配置対象ごとに，セクションの合計サイズを表示します。
		いいえ	セクションの合計サイズを表示しません。
ベクタ情報を出力する	ベクタ情報を出力するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］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 (-SHow=VECTOR)	ベクタ情報を出力します。
		いいえ	ベクタ情報を出力しません。

(4) ［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。

最適化方法	最適化方法を選択します。 リンカのオプション -nooptimize，および -optimize に相当します。		
	デフォルト	しない (-NOOPtimize)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	しない (-NOOPtimize)	モジュールの最適化を行いません。
		すべて (-OPtimize)	すべての最適化を行います。
		スピード重視 (-OPtimize=SPeed)	実行速度優先の最適化を行います。
		安全な最適化 (-OPtimize=SAFe)	安全な最適化を行います。
		カスタム	指定した項目の最適化を行います。
未参照シンボルを削除する	未参照シンボルを削除するかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=Symbol_delete)	未参照シンボルを削除します。
		いいえ	未参照シンボルを削除しません。
複数の同一命令列をサブルーチン化する	複数の同一命令列をサブルーチン化するかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=SAMe_code)	複数の同一命令列をサブルーチン化します。
		いいえ	複数の同一命令列をサブルーチン化しません。
最小コードサイズ	最適化対象となる最小コードサイズを指定します。 リンカのオプション -samesize に相当します。 なお，本プロパティは，［複数の同一命令列をサブルーチン化する］プロパティで［はい (-OPtimize=SAMe_code)］を選択した場合のみ表示します。		
	デフォルト	1E（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	8 ～ 7FFF（16 進数）	

コードサイズがより小さくなる命令に置き換える	コードサイズがより小さくなる命令に置き換えるかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=SHort_format)	コードサイズがより小さくなる命令に置き換えます。
		いいえ	コードサイズがより小さくなる命令に置き換えません。
分岐命令サイズを最適化する	プログラムの配置情報に基づいて，分岐命令サイズを最適化するかどうかを選択します。 リンカのオプション -optimize に相当します。 なお，本プロパティは，［最適化方法］プロパティで［カスタム］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 (-OPtimize=Branch)	プログラムの配置情報に基づいて，分岐命令サイズを最適化します。
		いいえ	分岐命令サイズを最適化しません。
最適化による削除を抑止する未参照シンボル	最適化による削除を抑止する未参照シンボルを指定します。 「*シンボル名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -symbol_forbid に相当します。 なお，本プロパティは，［最適化方法］プロパティで［しない (-NOOPtimize)］を選択した場合は表示しません。		
	デフォルト	最適化による削除を抑止する未参照シンボル [*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	32767 文字までの文字列 65536 個まで指定可能です。	
最適化で統合を抑止する共通コード	最適化で統合を抑止する共通コードを指定します。 「*関数名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -samecode_forbid に相当します。 なお，本プロパティは，［最適化方法］プロパティで［しない (-NOOPtimize)］を選択した場合は表示しません。		
	デフォルト	最適化で統合を抑止する共通コード [*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	32767 文字までの文字列 65536 個まで指定可能です。	

最適化を抑止するセクション	最適化を抑止するセクションを指定します。 「*ファイル名* (*セクション名 [,...]*)」，または「*モジュール名* (*セクション名 [,...]*)」の形式で 1 行に 1 つずつ指定します。 「*ファイル名*」または「*モジュール名*」の部分は省略可能です。 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。 %ProjectName%: 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。 リンカのオプション -section_forbid に相当します。 なお，本プロパティは，［最適化方法］プロパティで［しない (-NOOPtimize)］を選択した場合は表示しません。	
	デフォルト	最適化を抑止するセクション [*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	32767 文字までの文字列 65536 個まで指定可能です。
最適化を抑止するアドレス範囲	最適化を抑止するアドレス範囲を指定します。 「*アドレス* + *サイズ*」の形式で 1 行に 1 つずつ指定します。 「+ *サイズ*」の部分は省略可能です。 アドレスとサイズは 16 進数で指定します。 リンカのオプション -absolute_forbid に相当します。 なお，本プロパティは，［最適化方法］プロパティで［しない (-NOOPtimize)］を選択した場合は表示しません。	
	デフォルト	最適化を抑止するアドレス範囲 [*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	32767 文字までの文字列 65536 個まで指定可能です。

(5) ［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。

セクションの開始アドレス	セクションの開始アドレスを指定します。 リンカのオプション -start に相当します。	
	デフォルト	ターゲット・デバイス固有の値
	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，セクション設定 ダイアログによる編集
	指定可能値	1022 文字までの文字列
外部定義シンボルをファイル出力するセクション	外部定義シンボルをファイル出力するセクションを指定します。 「*セクション名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -fsymbol に相当します。	
	デフォルト	外部定義シンボルをファイル出力するセクション [*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	32767 文字までの文字列 65536 個まで指定可能です。

セクション・アライメント	アライメント数を 0x10 bytes に変更するセクション名を指定します。 「*セクション名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -aligned_section に相当します。	
	デフォルト	セクション・アライメント [*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	32767 文字までの文字列 65536 個まで指定可能です。
ROM から RAM へマップするセクション	初期化データ領域の ROM 用，RAM 用領域を確保し，ROM セクション内定義シンボルを RAM セクション内アドレスになるようリロケーションします。 「*ROM セクション名 =RAM セクション名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -rom に相当します。	
	デフォルト	ROM から RAM へマップするセクション [*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	32767 文字までの文字列 65535 個まで指定可能です。

(6) ［ベリファイ］

ベリファイに関する詳細情報の表示，および設定の変更を行います。

セクションの割り付けアドレスをチェックする	セクションの割り付けアドレスをチェックするかどうかを選択します。 リンカのオプション -cpu 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 (-CPu)	セクションの割り付けアドレスをチェックします。
		いいえ	セクションの割り付けアドレスをチェックしません。
メモリ種別のアドレス範囲	メモリ種別のアドレス範囲を指定します。 「*メモリ種別 = 先頭アドレス - 終了アドレス*」の形式で 1 行に 1 つずつ指定します。 「*メモリ種別*」は，ROm，RAm，または FIX を指定します。 アドレスは 16 進数で指定します。 リンカのオプション -cpu に相当します。 なお，本プロパティは，［セクションの割り付けアドレスをチェックする］プロパティで［はい (-CPu)］を選択した場合のみ表示します。		
	デフォルト	メモリ種別のアドレス範囲 [*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	32767 文字までの文字列 65536 個まで指定可能です。	

次の同メモリ種別に配置、または、分割して配置する	同メモリ種別の領域にセクションを分割するかどうかを選択します。 リンカのオプション -cpu に相当します。 なお，本プロパティは，［メモリ種別のアドレス範囲］プロパティでメモリ種別のアドレス範囲を指定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 (-CPu=stride)	同メモリ種別の領域にセクションを分割します。
		いいえ	同メモリ種別の領域にセクションを分割しません。
分割対象外セクション	分割対象外セクションを指定します。 「*セクション名*」の形式で 1 行に 1 つずつ指定します。 リンカのオプション -contiguous_section に相当します。 なお，本プロパティは，［次の同メモリ種別に配置，または，分割して配置する］プロパティで［はい (-CPu=stride)］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	［...］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	32767 文字までの文字列 65536 個まで指定可能です。	

(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。

スタック使用量情報ファイルを出力する	スタック使用量情報ファイルを出力するかどうかを選択します。 リンカのオプション -stack 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 (-STACk)	スタック使用量情報ファイルを出力します。
		いいえ	スタック使用量情報ファイルを出力しません。
デバッグ情報を圧縮する	デバッグ情報を圧縮するかどうかを選択します。 リンカのオプション -compress，および -nocompress に相当します。		
	デフォルト	いいえ (-NOCOmpress)	
	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 (-Compress)	デバッグ情報を圧縮します。
		いいえ (-NOCOmpress)	デバッグ情報を圧縮しません。

<table>
<tr><td rowspan="4">メモリ使用量を削減する</td><td colspan="3">メモリ使用量を削減するかどうかを選択します。
リンカのオプション -memory に相当します。
なお，本プロパティは，以下を選択した場合は表示しません。
［出力］カテゴリの［外部シンボル割り付け情報ファイルを出力する］プロパティで［はい (-Map)］
［リスト］カテゴリの［シンボルの参照回数を出力する］プロパティで［はい (-SHow=Reference)］，［クロスリファレンス情報を出力する］プロパティで［はい (-SHow=Xreference)］
［ベリファイ］カテゴリの［次の同メモリ種別に配置，または，分割して配置する］プロパティで［はい (-CPu=stride)］，［スタック使用量情報ファイル出力する］プロパティで［はい (-STACk)］，［デバッグ情報を圧縮する］プロパティで［はい (-Compress)］</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ (-MEMory=High)</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 (-MEMory=Low)</td><td>メモリ使用量を削減します。</td></tr>
<tr><td>いいえ (-MEMory=High)</td><td>メモリ使用量を削減しません。</td></tr>
<tr><td rowspan="5">ウォーニング・メッセージをインフォメーションレベルに変更する</td><td colspan="3">ウォーニング・メッセージをインフォメーションレベルに変更するかどうかを選択します。
リンカのオプション -change_message 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="3">指定可能値</td><td>はい (すべて) (-CHange_message=information)</td><td>すべてのウォーニング・メッセージをインフォメーションレベルに変更します。</td></tr>
<tr><td>はい (エラー番号指定) (-CHange_message=information=< エラー番号 >)</td><td>ウォーニングレベルの指定エラー番号のみインフォメーションレベルに変更します。</td></tr>
<tr><td>いいえ</td><td>ウォーニング・メッセージをインフォメーションレベルに変更しません。</td></tr>
<tr><td rowspan="4">ウォーニングレベルのエラー番号</td><td colspan="3">ウォーニングレベルのエラー番号を指定します。
複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。
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：4,200-203,1300）。
リンカのオプション -change_message に相当します。
なお，本プロパティは，［ウォーニング・メッセージをインフォメーションレベルに変更する］プロパティで［はい (エラー番号指定) (-CHange_message=information=< エラー番号 >)］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，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 colspan="2">32767 文字までの文字列</td></tr>
</table>

インフォメーション・メッセージをウォーニングレベルに変更する	インフォメーション・メッセージをウォーニングレベルに変更するかどうかを選択します。 リンカのオプション -change_message 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 (すべて) (-CHange_message=warning)	すべてのインフォメーション・メッセージをウォーニングレベルに変更します。
		はい (エラー番号指定) (-CHange_message=warning=< エラー番号 >)	インフォメーションレベルの指定エラー番号のみウォーニングレベルに変更します。
		いいえ	インフォメーション・メッセージをウォーニングレベルに変更しません。
インフォメーションレベルのエラー番号	インフォメーションレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 リンカのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション・メッセージをウォーニングレベルに変更する］プロパティで［はい (エラー番号指定) (-CHange_message=warning=< エラー番号 >)］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	32767 文字までの文字列	
インフォメーション、ウォーニング・メッセージをエラーレベルに変更する	インフォメーション，ウォーニング・メッセージをエラーレベルに変更するかどうかを選択します。 リンカのオプション -change_message 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 (すべて) (-CHange_message=error))	すべてのインフォメーション，ウォーニング・メッセージをエラーレベルに変更します。
		はい (エラー番号指定) (-CHange_message=error=< エラー番号 >)	インフォメーション，ウォーニングレベルの指定エラー番号のみエラーレベルに変更します。
		いいえ	インフォメーション，ウォーニング・メッセージをエラーレベルに変更しません。

インフォメーション、ウォーニングレベルのエラー番号	インフォメーション，ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 リンカのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション，ウォーニング・メッセージをエラーレベルに変更する］プロパティで［はい (エラー番号指定) (-CHange_message=error=< エラー番号 >)］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	32767 文字までの文字列	
ローカルシンボル名情報を消去する	ローカルシンボル名の情報を消去するかどうかを選択します。 リンカのオプション -hide 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 (-Hide)	ローカルシンボル名の情報を消去します。
		いいえ	ローカルシンボル名の情報を消去しません。
合計セクション・サイズを表示する	合計セクション・サイズを表示するかどうかを選択します。 リンカのオプション -total_size 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 (-Total_size)	合計セクション・サイズを表示します。
		いいえ	合計セクション・サイズを表示しません。
コピーライト情報を表示する	コピーライト情報を表示するかどうかを選択します。 リンカのオプション -logo，-nologo に相当します。		
	デフォルト	いいえ (-NOLOgo)	
	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 (-LOgo)	コピーライト情報を表示します。
		いいえ (-NOLOgo)	コピーライト情報の表示を抑止します。

<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。
%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・ファイル名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，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。
<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>
<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。
%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・ファイル名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，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。
<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>
</table>

その他の追加オプション	その他に追加するリンカの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，リンカのオプション群の最後に付加されます。	
	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，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
コマンド・ライン	指定されているオプションを表示します。	
	デフォルト	コマンド・ライン [*定義数*]
	変更方法	変更不可

［ライブラリアン・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) ［入力］
(2) ［出力］
(3) ［リスト］
(4) ［その他］

注意 本タブは，アプリケーション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：［ライブラリアン・オプション］タブ

［各カテゴリの説明］

(1) ［入力］

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。

オブジェクト・モジュール・ファイル	オブジェクト・モジュール・ファイルを指定します。 1 行に 1 ファイルずつ指定します。 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。 %ProjectName%: 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。 リンカのオプション -Input に相当します。 指定した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・モジュール・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	32767 文字までの文字列 65536 個まで指定可能です。

使用するライブラリ・ファイル	ライブラリ・ファイルを指定します。 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。 %ProjectName%: 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。 リンカのオプション -library 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	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	32767 文字までの文字列 65536 個まで指定可能です。
システム・ライブラリ・ファイル	リンク時にシステムが設定するシステム・ライブラリ・ファイルの指定順を変更します。 リンカのオプション -library に相当します。	
	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	[...] 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
	指定可能値	変更不可（システム・ライブラリ・ファイルの設定順の変更のみ可能）
バイナリ・データ・ファイル	バイナリ・データ・ファイルを指定します。 「*ファイル名* (*セクション名* : *アライメント数* / *セクション属性* , *シンボル名*)」の形式で 1 行に 1 つずつ指定します。 「: *アライメント数*」，「/ *セクション属性*」，「, *シンボル名*」の部分は省略可能です。 「*アライメント数*」は，1，2，4，8，16，または 32 になります。 省略した場合は，定義値を 1 とします。 「*セクション属性*」は，CODE または DATA になります。 省略した場合は，書き込み，読み取り，実行，すべての属性が有効になります。 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。 %ProjectName%: 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。 リンカのオプション -binary に相当します。 バイナリ・データ・ファイル名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，［出力］カテゴリの［出力ファイル形式］プロパティで［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］を選択した場合のみ表示します。	
	デフォルト	バイナリ・データ・ファイル指定 [*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	32767 文字までの文字列 65536 個まで指定可能です。

プレリンカを起動する	プレリンカ（C++ テンプレート・インスタンスの自動生成）を起動するかどうかを選択します。 リンカのオプション -noprelink に相当します。		
	デフォルト	自動制御	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	自動制御	リンカの入力ファイルに ii ファイルがなければ，プレリンカの起動を抑止します。
		はい	プレリンカを起動します。
		いいえ (-NOPRElink)	プレリンカの起動を抑止します。

(2) ［出力］

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。

出力ファイル形式	出力ファイル形式を選択します。 リンカのオプション -form に相当します。		
	デフォルト	ユーザ・ライブラリ・ファイル (-FOrm=Library=U)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	ユーザ・ライブラリ・ファイル (-FOrm=Library=U)	ユーザ・ライブラリ・ファイルを出力します。 なお，本項目は，［インフォメーションレベル・メッセージ出力を有効にする］プロパティで［はい (参照されない定義シンボルの通知) (-Message -MSg_unused)］を選択した場合は，選択できません。
		システム・ライブラリ・ファイル (-FOrm=Library=S)	システム・ライブラリ・ファイルを出力します。 なお，本項目は，［インフォメーションレベル・メッセージ出力を有効にする］プロパティで［はい (参照されない定義シンボルの通知) (-Message -MSg_unused)］を選択した場合は，選択できません。
		リロケータブル・モジュール・ファイル (-FOrm=Relocate)	リロケータブル・モジュール・ファイルを出力します。
デバッグ情報を出力する	デバッグ情報を出力するかどうかを選択します。 リンカのオプション -debug，-nodebug に相当します。 なお，本プロパティは，［出力ファイル形式］プロパティで［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］を選択した場合のみ表示します。		
	デフォルト	はい (出力ファイル内) (-DEBug)	
	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 (出力ファイル内) (-DEBug)	デバッグ情報を出力します（出力ファイル内）。
		いいえ (-NODEBug)	デバッグ情報を出力しません。

<table>
<tr><td>出力フォルダ</td><td colspan="3">出力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。
%BuildModeName% ： ビルド・モード名に置換します。
%ProjectName% ： 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。
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。
リンカのオプション -output 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">出力ファイル名を指定します。
拡張子を省略した場合は，[出力ファイル形式] プロパティに依存し自動的に付加します。
“ ユーザ・ライブラリ・ファイル (-FOrm=Library=U)”：.lib
“ システム・ライブラリ・ファイル (-FOrm=Library=S)”：.lib
“ リロケータブル・モジュール・ファイル (-FOrm=Relocate)”：.rel
次のプレースホルダに対応しています。
%ProjectName% ： プロジェクト名に置換します。
リンカのオプション -output に相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">［出力ファイル形式］プロパティで，［ユーザ・ライブラリ・ファイル (-FOrm=Library=U)］を選択した場合
%ProjectName%.lib
［出力ファイル形式］プロパティで，［システム・ライブラリ・ファイル (-FOrm=Library=S)］を選択した場合
%ProjectName%.lib
［出力ファイル形式］プロパティで，［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］を選択した場合
%ProjectName%.rel</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>インフォメーションレベル・メッセージ出力を有効にする</td><td colspan="3">インフォメーションレベル・メッセージ出力を有効にするかどうかを選択します。
リンカのオプション -message，-msg_unused，-nomessage に相当します。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">いいえ (-NOMessage)</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="3">指定可能値</td><td>はい (-Message)</td><td>インフォメーション・レベル・メッセージを出力します。</td></tr>
<tr><td>はい (参照されない定義シンボルの通知) (-Message -MSg_unused)</td><td>参照されない定義シンボルを通知します。
なお，本項目は，［出力ファイル形式］プロパティで，［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］を選択した場合のみ選択できます。</td></tr>
<tr><td>いいえ (-NOMessage)</td><td>インフォメーション・レベル・メッセージを抑止します。</td></tr>
</table>

抑止するインフォメーションレベル・メッセージ番号	抑止するインフォメーションレベル・メッセージ番号を指定します。 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：4,200）。 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：4,200-203,1300）。 リンカのオプション -nomessage に相当します。 なお，本プロパティは，［インフォメーションレベル・メッセージ出力を有効にする］プロパティで［いいえ (-NOMessage)］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	32767 文字までの文字列

(3) ［リスト］

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。

リンケージ・リスト・ファイルを出力する	リンケージ・リスト・ファイルを出力するかどうかを選択します。 リンカのオプション -list，-show に相当します。		
	デフォルト	はい (リスト内容 = 選択) (-LISt)	
	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 (リスト内容 = 指定なし) (-LISt -SHow)	リンケージ・リスト・ファイルに出力形式に従った情報を出力します。
		はい (リスト内容 = すべて) (-LISt -SHow=ALL)	リンケージ・リスト・ファイルに出力形式に従ったすべての情報を出力します。
		はい (リスト内容 = 選択) (-LISt)	リンケージ・リスト・ファイルに指定した情報を出力します。
		いいえ	リンケージ・リスト・ファイルを出力しません。
シンボル情報を出力する	モジュール内シンボル名一覧を出力するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］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 (-SHow=SYmbol)	モジュール内シンボル名一覧を出力します。
		いいえ	モジュール内シンボル名一覧を出力しません。

モジュール内セクション一覧を出力する	モジュール内セクション一覧を出力するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］，および［出力］カテゴリの［出力ファイル形式］プロパティで［ユーザ・ライブラリ・ファイル (-FOrm=Library=U)］，または［システム・ライブラリ・ファイル (-FOrm=Library=S)］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 (-SHow=SEction)	モジュール内セクション一覧を出力します。
		いいえ	モジュール内セクション一覧を出力しません。
クロスリファレンス情報を出力する	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。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］，および［出力］カテゴリの［出力ファイル形式］プロパティで［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］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 (-SHow=Xreference)	クロスリファレンス情報を出力します。
		いいえ	クロスリファレンス情報を出力しません。
セクションの合計サイズを表示する	セクションの合計サイズを表示するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］，および［出力］カテゴリの［出力ファイル形式］プロパティで［ユーザ・ライブラリ・ファイル (-FOrm=Library=U)］，または［システム・ライブラリ・ファイル (-FOrm=Library=S)］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 (-SHow=Total_size)	セクションの合計サイズを表示します。
		いいえ	セクションの合計サイズを表示しません。
ベクタ情報を出力する	ベクタ情報を出力するかどうかを選択します。 リンカのオプション -show に相当します。 なお，本プロパティは，［リンケージ・リスト・ファイルを出力する］プロパティで［はい (リスト内容 = 選択) (-LISt)］，および［出力］カテゴリの［出力ファイル形式］プロパティで［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］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 (-SHow=VECTOR)	ベクタ情報を出力します。
		いいえ	ベクタ情報を出力しません。

(4) ［その他］

ライブラリ・ジェネレータ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 rowspan="5">メモリ使用量を削減する</td><td colspan="3">入力ファイル・ロード時のメモリ使用量を削減するかどうかを選択します。
リンカのオプション -memory に相当します。
なお，本プロパティは，以下を選択した場合は表示しません。
［出力］カテゴリの［出力ファイル形式］プロパティで［ユーザ・ライブラリ・ファイル (-FOrm=Library=U)］または［システム・ライブラリ・ファイル (-FOrm=Library=S)］，および［ローカルシンボル名情報を消去する］プロパティで［はい (-Hide)］
［出力］カテゴリの［出力ファイル形式］プロパティで［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］</td></tr>
<tr><td>デフォルト</td><td colspan="2">いいえ (-MEMory=High)</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 (-MEMory=Low)</td><td>入力ファイル・ロード時のメモリ使用量を削減します。</td></tr>
<tr><td>いいえ (-MEMory=High)</td><td>入力ファイル・ロード時のメモリ使用量を削減しません。</td></tr>
<tr><td rowspan="5">ウォーニング・メッセージをインフォメーションレベルに変更する</td><td colspan="3">ウォーニング・メッセージをインフォメーションレベルに変更するかどうかを選択します。
リンカのオプション -change_message 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="3">指定可能値</td><td>はい (すべて) (-CHange_message=information)</td><td>すべてのウォーニング・メッセージをインフォメーションレベルに変更します。</td></tr>
<tr><td>はい (エラー番号指定) (-CHange_message=information=< エラー番号 >)</td><td>ウォーニングレベルの指定エラー番号のみインフォメーションレベルに変更します。</td></tr>
<tr><td>いいえ</td><td>ウォーニング・メッセージをインフォメーションレベルに変更しません。</td></tr>
<tr><td rowspan="4">ウォーニングレベルのエラー番号</td><td colspan="3">ウォーニングレベルのエラー番号を指定します。
複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。
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：4,200-203,1300）。
リンカのオプション -change_message に相当します。
なお，本プロパティは，［ウォーニング・メッセージをインフォメーションレベルに変更する］プロパティで［はい (エラー番号指定) (-CHange_message=information=< エラー番号 >)］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，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 colspan="2">32767 文字までの文字列</td></tr>
</table>

インフォメーション・メッセージをウォーニングレベルに変更する	インフォメーション・メッセージをウォーニングレベルに変更するかどうかを選択します。 リンカのオプション -change_message 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 (すべて) (-CHange_message=warning)	すべてのインフォメーション・メッセージをウォーニングレベルに変更します。
		はい (エラー番号指定) (-CHange_message=warning=< エラー番号 >)	インフォメーションレベルの指定エラー番号のみウォーニングレベルに変更します。
		いいえ	インフォメーション・メッセージをウォーニングレベルに変更しません。
インフォメーションレベルのエラー番号	インフォメーションレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 リンカのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション・メッセージをウォーニングレベルに変更する］プロパティで［はい (エラー番号指定) (-CHange_message=warning=< エラー番号 >)］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	32767 文字までの文字列	
インフォメーション、ウォーニング・メッセージをエラーレベルに変更する	インフォメーション，ウォーニング・メッセージをエラーレベルに変更するかどうかを選択します。 リンカのオプション -change_message 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 (すべて) (-CHange_message=error))	すべてのインフォメーション，ウォーニング・メッセージをエラーレベルに変更します。
		はい (エラー番号指定) (-CHange_message=error=< エラー番号 >)	インフォメーション，ウォーニングレベルの指定エラー番号のみエラーレベルに変更します。
		いいえ	インフォメーション，ウォーニング・メッセージをエラーレベルに変更しません。

インフォメーション、ウォーニングレベルのエラー番号	インフォメーション，ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 リンカのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション，ウォーニング・メッセージをエラーレベルに変更する］プロパティで［はい (エラー番号指定) (-CHange_message=error=< エラー番号 >)］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	32767 文字までの文字列	
ローカルシンボル名情報を消去する	ローカルシンボル名情報を消去するかどうかを選択します。 リンカのオプション -hide 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 (-Hide)	ローカルシンボル名情報を消去します。
		いいえ	ローカルシンボル名情報を消去しません。
合計セクション・サイズを表示する	合計セクション・サイズを表示するかどうかを選択します。 リンカのオプション -total_size に相当します。 なお，本プロパティは，［出力］カテゴリの［出力ファイル形式］プロパティで［リロケータブル・モジュール・ファイル (-FOrm=Relocate)］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 (-Total_size)	合計セクション・サイズを表示します。
		いいえ	合計セクション・サイズを表示しません。
コピーライト情報を表示する	コピーライト情報を表示するかどうかを選択します。 リンカのオプション -logo，-nologo に相当します。		
	デフォルト	いいえ (-NOLOgo)	
	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 (-LOgo)	コピーライト情報を表示します。
		いいえ (-NOLOgo)	コピーライト情報の表示を抑止します。

<table>
<tr><td>ライブラリアン前に実行するコマンド</td><td>ライブラリアン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。
%LibrarianFile%: ライブラリアン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。
%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・ファイル名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，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。
<table><tr><td>デフォルト</td><td>ライブラリアン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>
<tr><td>ライブラリアン後に実行するコマンド</td><td>ライブラリアン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。
%LibrarianFile%: ライブラリアン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。
%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・ファイル名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，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。
<table><tr><td>デフォルト</td><td>ライブラリアン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>
</table>

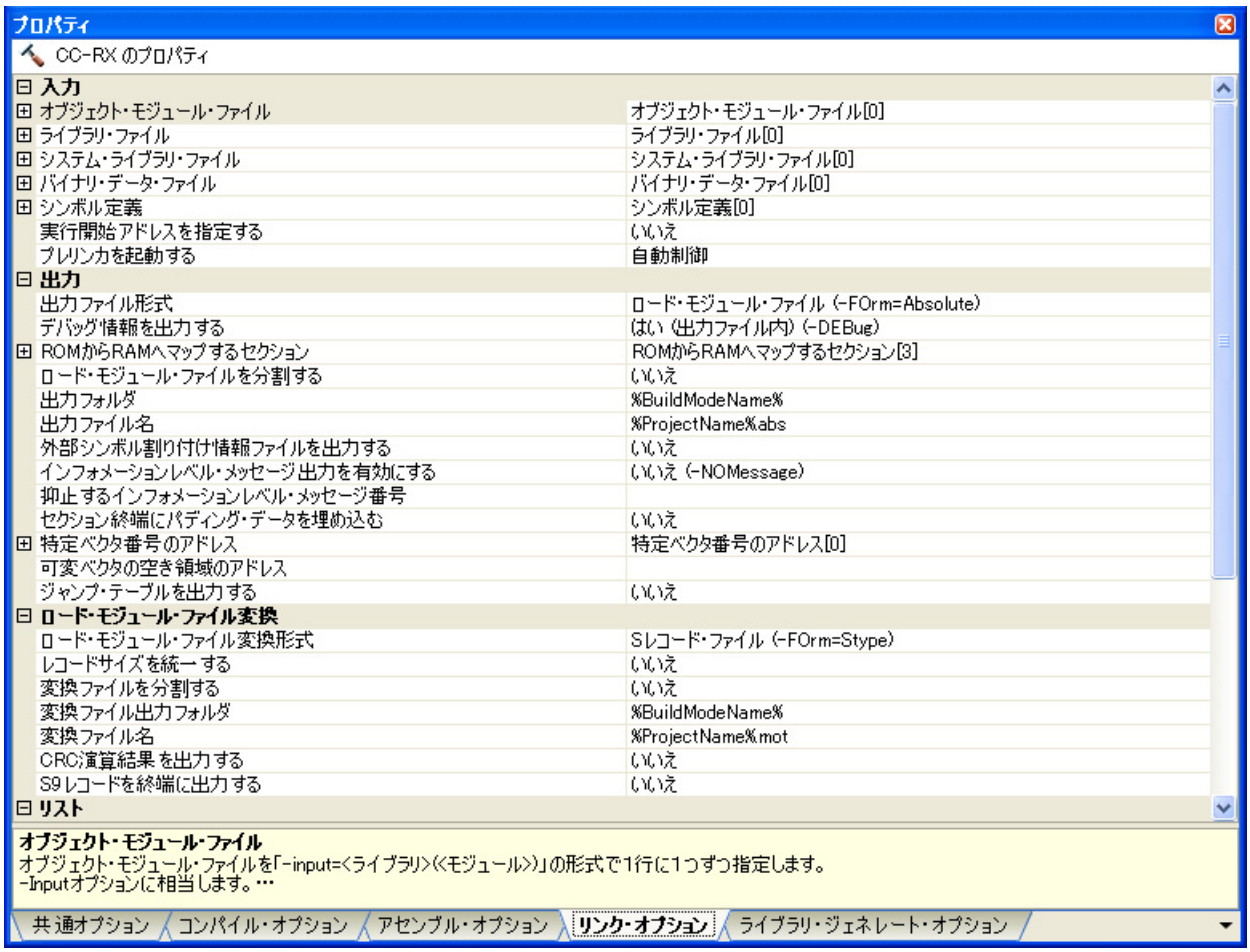
その他の追加オプション	その他に追加するライブラリ・ジェネレータのオプションを入力します。 なお，ここで設定したオプションは，ライブラリ・ジェネレータのオプション群の最後に付加されます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	259 文字までの文字列
コマンド・ライン	指定されているオプションを表示します。	
	デフォルト	コマンド・ライン [*定義数*]
	変更方法	変更不可

［ヘキサ出力オプション］タブ

本タブでは，リンク・フェーズ (ヘキサ出力) 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。

ヘキサ・ファイルを出力する	ヘキサ・ファイルを出力するかどうかを選択します。 リンカのオプション -form 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
		いいえ
出力フォルダ	ヘキサ・ファイルの出力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。 %BuildModeName% ：ビルド・モード名に置換します。 %ProjectName% ：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。 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。 リンカのオプション -output に相当します。 なお、本プロパティは、［ヘキサ・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	%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
出力ファイル名	出力ファイル名を指定します。 拡張子を省略した場合は、［ヘキサ・ファイル・フォーマット］プロパティの選択に依存して、自動的に付加します。 ［インテル拡張ヘキサ・ファイル (-Form=Hexadecimal)］を選択している場合：.hex ［モトローラ・S タイプ・ファイル (-Form=Stype)］を選択している場合：.mot ［バイナリ・ファイル (-Form=Binary)］を選択している場合：.bin 次のプレースホルダに対応しています。 %ProjectName% ：プロジェクト名に置換します。 空欄の場合は，%ProjectName%.abs を指定したものとみなします。リンカのオプション -output に相当します。 なお、本プロパティは、［ヘキサ・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	%ProjectName%.mot
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	255 文字までの文字列

分割出力ファイル	分割出力ファイルを指定します。 分割する出力ファイル名を「ファイル名 = 先頭アドレス - 終了アドレス」または「ファイル名 = セクション名」の形式で 1 行に 1 つずつ指定します。 「セクション名」を複数指定する場合は，「ファイル名 = セクション名 : セクション名」のように，コロンで区切って指定します（例：file1.abs=sec1:sec2）。 アドレスは 16 進数で指定します（例：file2.abs=400-4ff）。 拡張子を省略した場合は，［ヘキサ・ファイル・フォーマット］プロパティに依存して、自動的に付加します。 “ インテル拡張ヘキサ・ファイル (-Form=Hexadecimal)”：.hex “ モトローラ・S タイプ・ファイル (-Form=Stype)”：.mot “ バイナリ・ファイル (-Form=Binary)”：.bin 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。 %ProjectName% ：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。 リンカのオプション -output に相当します。 なお，本プロパティは，［ヘキサ・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	分割出力ファイル［*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	255 文字までの文字列 65536 個まで指定可能

(2) ［ヘキサ・フォーマット］

ヘキサ・ファイルの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	ヘキサ・ファイルのフォーマットを選択します。 リンカのオプション -form 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［ヘキサ・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	モトローラ・S タイプ・ファイル (-FOrm=Stype	
	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・ファイル (-FOrm=Hexadecimal)	インテル拡張ヘキサ・ファイルを出力します。
		モトローラ・S タイプ・ファイル (-FOrm=Stype)	モトローラ・S タイプ・ファイルを出力します。
		バイナリ・ファイル (-FOrm=Binary)	バイナリ・ファイルを出力します。

レコードサイズを統一する	アドレス範囲に関係なく，一定のデータレコードで出力するかどうかを選択します。 リンカのオプション -record 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［インテル拡張ヘキサ・ファイル (-FOrm=Hexadecimal)］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 (インテル HEX レコード) (-REcord=H16)	インテル HEX レコードを出力します。
		はい (インテル拡張 HEX レコード) (-REcord=H20)	インテル拡張 HEX レコードを出力します。
		はい (インテル 32bitHEX レコード) (-REcord=H32)	インテル 32bitHEX レコードを出力します。
		いいえ	アドレスに合わせて混在したデータレコードを出力します。
レコードサイズを統一する	アドレス範囲に関係なく，一定のデータレコードで出力するかどうかを選択します。 リンカのオプション -record 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［モトローラ・S タイプ・ファイル (-FOrm=Stype)］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 (S1 レコード) (-REcord=S1)	S1 レコードを出力します。
		はい (S2 レコード) (-REcord=S2)	S2 レコードを出力します。
		はい (S3 レコード) (-REcord=S3)	S3 レコードを出力します。
		いいえ	アドレスに合わせて混在したデータレコードを出力します。
出力範囲のメモリの空き領域をデータで充填する	出力範囲のメモリの空き領域をデータで充填するかどうかを選択します。 リンカのオプション -space 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［分割出力ファイル］プロパティで出力ファイル名を指定している場合に表示します。		
	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 (乱数) (-SPace=Random)	空きエリアに乱数を出力します。
		はい (データ指定) (-SPace=< 数値 >)	空きエリアに指定した 16 進数の数値を出力します。
		いいえ	空きエリアにデータを出力しません。

空きエリア出力データ	空きエリアへの出力データを指定します。 リンカのオプション -space に相当します。 なお，本プロパティは，［出力範囲のメモリの空き領域をデータで充填する］プロパティで［はい (データ指定) (-SPace=< 数値 >)］を選択した場合のみ表示します。		
	デフォルト	FF（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）	
データ・レコードのバイト数を指定する	データ・レコードのバイト数を指定するかどうかを選択します。 リンカのオプション -byte_count 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［インテル拡張ヘキサ・ファイル (-FOrm=Hexadecimal)］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 (-BYte_count)	データ・レコードのバイト数を指定します。
		いいえ	データ・レコードのバイト数を指定しません。
データ・レコードのバイト数最大値	データ・レコードのバイト数の最大値を指定します。 リンカのオプション -byte_count に相当します。 なお，本プロパティは，［データ・レコードのバイト数を指定する］プロパティで［はい (-BYte_count)］を選択した場合のみ表示します。		
	デフォルト	FF（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	1 ～ FF（16 進数）	

CRC 演算結果を出力する	CRC 演算結果の出力を選択します。 リンカのオプション -crc 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［インテル拡張ヘキサ・ファイル (-Form=Hexadecimal)］，または［モトローラ・S レコード・ファイル (-Form=Stype)］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 (多項式 =CRC-CCITT，エンディアン = 自動) (-CRc)	多項式に CRC-CCITT を選択します。
		はい (多項式 =CRC-CCITT，エンディアン =Big-endian データ) (-CRc)	多項式に CRC-CCITT，エンディアンに BIG を選択します。
		はい (多項式 =CRC-CCITT，エンディアン =Little-endian データ) (-CRc)	多項式に CRC-CCITT，エンディアンに LITTLE を選択します。
		はい (多項式 =CRC-16，エンディアン = 自動) (-CRc)	多項式に CRC-16 を選択します。
		はい (多項式 =CRC-16，エンディアン =Big-endian データ) (-CRc)	多項式に CRC-16，エンディアンに BIG を選択します。
		はい (多項式 =CRC-16，エンディアン =Little-endian データ) (-CRc)	多項式に CRC-16，エンディアンに LITTLE を選択します。
		いいえ	CRC コードを出力しません。
出力アドレス	CRC コードの出力位置のアドレスを指定します。 アドレスは 16 進数で指定します。 リンカのオプション -crc に相当します。 なお，本プロパティは，［CRC 演算結果を出力する］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	0	
	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）	
計算範囲	CRC 計算範囲を指定します。 「*先頭アドレス - 終了アドレス*」の形式で 1 行に 1 つずつ指定します。 アドレスは 16 進数で指定します（例：400-ffff）。 リンカのオプション -crc に相当します。 なお，本プロパティは，［CRC 演算結果を出力する］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	[...] 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	32767 文字までの文字列 65536 個まで指定可能です。	

S9 レコードを終端に出力する	S9 レコードを終端に出力するかどうかを選択します。 リンカのオプション -s9 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［モトローラ・S タイプ・ファイル (-FOrm=Stype)］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 (-S9)	S9 レコードを終端に出力します。
		いいえ	S9 レコードを終端に出力しません。

(3) ［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。

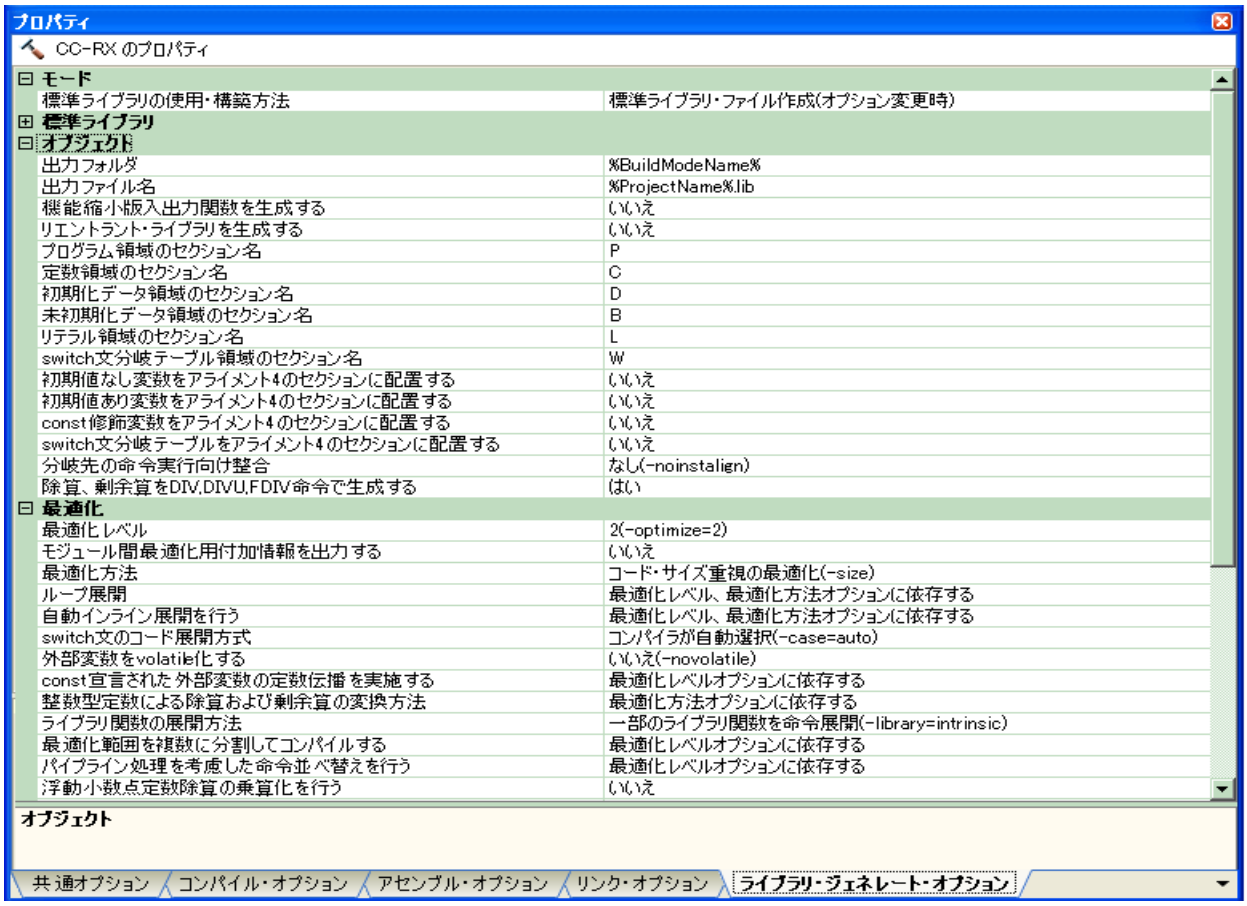
その他の追加オプション	その他に追加するリンカのオプションを入力します。 なお，ここで設定したオプションは，リンカのオプション群の最後に付加されます。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	259 文字までの文字列
コマンド・ライン	指定されているオプションを表示します。	
	デフォルト	コマンド・ライン [*定義数*]
	変更方法	変更不可

［ライブラリ・ジェネレート・オプション］タブ

本タブでは，ライブラリ・ジェネレート・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.10 プロパティ パネル：［ライブラリ・ジェネレート・オプション］タブ



［各カテゴリの説明］

(1) ［モード］

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。

標準ライブラリの使用・構築方法	標準ライブラリの使用・構築方法を選択します。		
	デフォルト	標準ライブラリ・ファイル作成 (オプション変更時)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	標準ライブラリ・ファイル作成 (常に作成)	標準ライブラリ・ファイルを作成します。
		標準ライブラリ・ファイル作成 (オプション変更時)	ビルド，リビルドに関わらず，オプションを変更した場合のみ，標準ライブラリ・ファイルを作成します。
		標準ライブラリ・ファイル指定なし	標準ライブラリ・ファイルを指定しません。

(2) ［標準ライブラリ］

標準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。
なお，本カテゴリは，［モード］カテゴリの［標準ライブラリの使用・構築方法］プロパティで［標準ライブラリ・ファイル指定なし］を選択した場合は表示しません。

ライブラリ構成	使用可能な C 言語標準ライブラリ関数の構成を選択します。 ライブラリ・ジェネレータのオプション -lang に相当します。		
	デフォルト	C(C89) (-lang=c)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	C(C89) (-lang=c)	C 言語の標準関数を C89 規格準拠のものだけで構成します。
		C99 (-lang=c99)	C 言語の標準関数を C89 規格および C99 規格準拠の内容で構成します。
構築対象のライブラリ	構築対象のライブラリを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。		
	デフォルト	カスタム (-head=<SubOption>)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	カスタム (-head=<SubOption>)	構築対象のライブラリを指定します。
		すべて有効 (-head=all)	すべてのライブラリ関数とランタイムライブラリを指定します。
		すべて無効 (-head=runtime)	構築対象のライブラリを指定しません。
ランタイム・ライブラリを有効にする	ランタイム・ライブラリを有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］を選択した場合のみ表示します。		
	デフォルト	はい (-head=runtime)	
	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 (-head=runtime)	ランタイム・ライブラリを有効にします。
		いいえ	ランタイム・ライブラリを無効にします。

ctype.h(C89/C99) を有効にする	ctype.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=ctype)	ctype.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	ctype.h(C89/C99) を無効にします。
math.h(C89/C99) を有効にする	math.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=math)	math.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	math.h(C89/C99) を無効にします。
mathf.h(C89/C99) を有効にする	mathf.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=mathf)	mathf.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	mathf.h(C89/C99) を無効にします。
stdarg.h(C89/C99) を有効にする	stdarg.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=stdarg)	stdarg.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	stdarg.h(C89/C99) を無効にします。
stdio.h(C89/C99) を有効にする	stdio.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］を選択した場合のみ表示します。		
	デフォルト	はい (-head=stdio)	
	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 (-head=stdio)	stdio.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	stdio.h(C89/C99) を無効にします。

stdlib.h(C89/C99) を有効にする	stdlib.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］を選択した場合のみ表示します。		
	デフォルト	はい (-head=stdlib)	
	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 (-head=stdlib)	stdlib.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	stdlib.h(C89/C99) を無効にします。
string.h(C89/C99) を有効にする	string.h(C89/C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］を選択した場合のみ表示します。		
	デフォルト	はい (-head=string)	
	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 (-head=string)	string.h(C89/C99) とランタイムライブラリを有効にします。
		いいえ	string.h(C89/C99) を無効にします。
ios(EC++) を有効にする	ios(EC++) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=ios)	ios(EC++) とランタイムライブラリを有効にします。
		いいえ	ios(EC++) を無効にします。
new(EC++) を有効にする	new(EC++) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］を選択した場合のみ表示します。		
	デフォルト	はい (-head=new)	
	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 (-head=new)	new(EC++) とランタイムライブラリを有効にします。
		いいえ	new(EC++) を無効にします。
complex(EC++) を有効にする	complex(EC++) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=complex)	complex(EC++) とランタイムライブラリを有効にします。
		いいえ	complex(EC++) を無効にします。

string(EC++) を有効にする	string(EC++) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=cppstring)	string(EC++) とランタイムライブラリを有効にします。
		いいえ	string(EC++) を無効にします。
complex.h(C99) を有効にする	complex.h(C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［ライブラリ構成］プロパティで［C99 (-lang=c99)］，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=C99_complex)	complex.h(C99) とランタイムライブラリを有効にします。
		いいえ	complex.h(C99) を無効にします。
fenv.h(C99) を有効にする	fenv.h(C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［ライブラリ構成］プロパティで［C99 (-lang=c99)］，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=fenv)	fenv.h(C99) とランタイムライブラリを有効にします。
		いいえ	fenv.h(C99) を無効にします。
inttypes.h(C99) を有効にする	inttypes.h(C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［ライブラリ構成］プロパティで［C99 (-lang=c99)］，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=inttypes)	inttypes.h(C99) とランタイムライブラリを有効にします。
		いいえ	inttypes.h(C99) を無効にします。

wchar.h(C99) を有効にする	wchar.h(C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［ライブラリ構成］プロパティで［C99 (-lang=c99)］，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=wchar)	wchar.h(C99) とランタイムライブラリを有効にします。
		いいえ	wchar.h(C99) を無効にします。
wctype.h(C99) を有効にする	wctype.h(C99) を有効にするかどうかを選択します。 ライブラリ・ジェネレータのオプション -head に相当します。 なお，本プロパティは，［ライブラリ構成］プロパティで［C99 (-lang=c99)］，［構築対象のライブラリ］プロパティで［カスタム (-head=<SubOption>)］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 (-head=wctype)	wctype.h(C99) とランタイムライブラリを有効にします。
		いいえ	wctype.h(C99) を無効にします。

(3) ［オブジェクト］

オブジェク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。
なお，本カテゴリは，［モード］カテゴリの［標準ライブラリの使用・構築方法］プロパティで［標準ライブラリ・ファイル指定なし］を選択した場合は表示しません。

出力フォルダ	出力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。 %BuildModeName%：ビルド・モード名に置換します。 %ProjectName%：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。 パスはプロジェクト・フォルダを基点とします。 ライブラリ・ジェネレータのオプション -output 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
出力ファイル名	出力ファイル名を指定します。 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。 %ProjectName%：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。 ライブラリ・ジェネレータのオプション -output に相当します。	
	デフォルト	%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	32767 文字までの文字列

機能縮小版入出力関数を生成する	機能縮小版入出力関数を生成するかどうかを選択します。 ライブラリ・ジェネレータのオプション -nofloat，-simple_stdio 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 (機能縮小版 1) (-nofloat)	浮動小数点変換 (%f，%e，%E，%g，%G) をサポートしない簡易入出力関数を生成します。
		はい (機能縮小版 2) (-simple_stdio)	浮動小数点変換，long long 型の変換，2 バイトコードの変換をサポートしない簡易入出力関数を生成します。
		いいえ	機能縮小版入出力関数を生成しません。
プログラム領域のセクション名	プログラム領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	P	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
定数領域のセクション名	定数領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	C	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
初期化データ領域のセクション名	初期化データ領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	D	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
未初期化データ領域のセクション名	未初期化データ領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	B	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
リテラル領域のセクション名	リテラル領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	L	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
switch 文分岐テーブル領域のセクション名	switch 文分岐テーブル領域のセクション名を指定します。 コンパイラのオプション -section に相当します。		
	デフォルト	W	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	

初期値なし変数をアライメント 4 のセクションに配置する	初期値なし変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=B)	初期値なし変数をアライメント 4 のセクションに配置します。
		いいえ	初期値なし変数をアライメント 4 のセクションに配置しません。
初期値あり変数をアライメント 4 のセクションに配置する	初期値あり変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=D)	初期値あり変数をアライメント 4 のセクションに配置します。
		いいえ	初期値あり変数をアライメント 4 のセクションに配置しません。
const 修飾変数をアライメント 4 のセクションに配置する	const 修飾変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=C)	const 修飾変数をアライメント 4 のセクションに配置します。
		いいえ	const 修飾変数をアライメント 4 のセクションに配置しません。
switch 文分岐テーブルをアライメント 4 のセクションに配置する	switch 文分岐テーブルをアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=W)	switch 文分岐テーブルをアライメント 4 のセクションに配置します。
		いいえ	switch 文分岐テーブルをアライメント 4 のセクションに配置しません。

分岐先の命令実行向け整合	分岐先の命令実行向け整合を選択します。 コンパイラのオプション -noinstalign，-instalign4，-instalign8 に相当します。		
	デフォルト	なし (-noinstalign)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	なし (-noinstalign)	分岐先を命令実行向け整合をしません。
		4 バイト整合 (-instalign4)	分岐先を 4 バイトで命令実行向け整合します。
		4 バイト整合 (各ループの先頭含む) (-instalign4=loop)	分岐先を 4 バイトで命令実行向け整合します (各ループの先頭含む)。
		4 バイト整合 (各最内周ループの先頭含む) (-instalign4=inmostloop)	分岐先を 4 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
		8 バイト整合 (-instalign8)	分岐先を 8 バイトで命令実行向け整合します。
		8 バイト整合 (各ループの先頭含む) (-instalign8=loop)	分岐先を 8 バイトで命令実行向け整合します (各ループの先頭含む)。
		8 バイト整合 (各最内周ループの先頭含む) (-instalign8=inmostloop)	分岐先を 8 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
除算、剰余算を DIV,DIVU,FDIV 命令で生成する	除算，剰余算を DIV,DIVU,FDIV 命令で生成するかどうかを選択します。 コンパイラのオプション -nouse_div_inst 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	除算，剰余算に DIV,DIVU,FDIV 命令を使ったコードを生成します。
		いいえ (-nouse_div_inst)	除算，剰余算に DIV,DIVU,FDIV 命令を使わないコードを生成します。

(4) ［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。
なお，本カテゴリは，［モード］カテゴリの［標準ライブラリの使用・構築方法］プロパティで［標準ライブラリ・ファイル指定なし］を選択した場合は表示しません。

最適化レベル	最適化レベルを選択します。 ライブラリ・ジェネレータのオプション -optimize に相当します。		
	デフォルト	2 (-optimize=2)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	0 (-optimize=0)	最適化を実施しません。
		1 (-optimize=1)	自動変数のレジスタ割り付け，関数出口ブロックの統合，統合可能な複数命令の統合など，一部最適化を実施します。
		2 (-optimize=2)	全般的に最適化を実施します。
		Max (-optimize=max)	実施可能な最適化を最大限に行います。

モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 ライブラリ・ジェネレータのオプション -goptimize 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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。
最適化方法	最適化方法を選択します。 ライブラリ・ジェネレータのオプション -speed，-size に相当します。		
	デフォルト	コード・サイズ重視の最適化 (-size)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	実行性能重視の最適化 (-speed)	実行性能重視の最適化を実施します。
		コード・サイズ重視の最適化 (-size)	コードサイズ重視の最適化を実施します。
ループ展開	ループ文 (for, while, do-while) を展開するかどうかを選択します。 ライブラリ・ジェネレータのオプション -loop に相当します。		
	デフォルト	最適化レベル，最適化方法オプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		展開する (-loop=< 数値 >)	ループ文 (for, while, do-while) を展開します。
最大展開数	最大で何倍の展開を行うかを指定します。 ライブラリ・ジェネレータのオプション -loop に相当します。 なお，本プロパティは，［ループ展開］プロパティで［展開する (-loop=< 数値 >)］を選択した場合のみ表示します。		
	デフォルト	2（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	1 ～ 32（10 進数）	
自動インライン展開を行う	自動インライン展開を行うかどうかを選択します。 ライブラリ・ジェネレータのオプション -inline，-noinline に相当します。		
	デフォルト	最適化レベル，最適化方法オプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		はい (-inline=< 数値 >)	自動インライン展開を行います。
		いいえ (-noinline)	自動インライン展開を行いません。

関数サイズの最大増加率	関数サイズの最大増加率を指定します。 ライブラリ・ジェネレータのオプション -inline に相当します。 なお，本プロパティは，［自動インライン展開を行う］プロパティで［はい (-inline=< 数値 >)］を選択した場合のみ表示します。		
	デフォルト	100（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	1 ～ 65535（10 進数）	
switch 文のコード展開方式	switch 文のコード展開方式を選択します。 ライブラリ・ジェネレータのオプション -case に相当します。		
	デフォルト	コンパイラが自動選択 (-case=auto)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	if_then 方式 (-case=ifthen)	switch 文を if_then 方式で展開します。
		テーブル・ジャンプ方式 (-case=table)	switch 文をテーブル方式で展開します。
		コンパイラが自動選択 (-case=auto)	if_then 方式，テーブル方式いずれかをコンパイラが自動的に選択します。
外部変数を volatile 化する	すべての外部変数を volatile 宣言したものとして扱うかどうかを選択します。 ライブラリ・ジェネレータのオプション -volatile，-novolatile に相当します。		
	デフォルト	いいえ (-novolatile)	
	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 (-volatile)	すべての外部変数を volatile 宣言したものとして扱います。
		いいえ (-novolatile)	volatile 修飾のない外部変数に対して最適化を行います。
const 宣言された外部変数の定数伝播を実施する	const 宣言された外部変数の定数伝播を実施するかどうかを選択します。 ライブラリ・ジェネレータのオプション -const_copy，-noconst_copy に相当します。		
	デフォルト	最適化レベルオプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-const_copy)	const 修飾型外部変数についても定数伝播を行います。
		いいえ (-noconst_copy)	const 修飾型外部変数の定数伝播を抑止します。

整数型定数による除算および剰余算の変換方法	整数型定数による除算および剰余算の変換方法を選択します。 ライブラリ・ジェネレータのオプション -const_div，-noconst_div に相当します。		
	デフォルト	最適化方法オプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化方法オプションに依存する	最適化方法オプションに従います。
		乗算を用いた命令列に変換 (-const_div)	ソースファイル中の整数型定数による除算および剰余算を，乗算を用いた命令列に変換します。
		除算を用いた命令列に変換 (-noconst_div)	ソースファイル中の整数型定数による除算および剰余算を，除算を用いた命令列に変換します。
ライブラリ関数の展開方法	ライブラリ関数の展開方法を選択します。 ライブラリ・ジェネレータのオプション -library に相当します。		
	デフォルト	一部のライブラリ関数を命令展開 (-library=intrinsic)	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	すべて関数呼び出し (-library=function)	ライブラリ関数をすべて関数呼び出しします。
		一部のライブラリ関数を命令展開 (-library=intrinsic)	abs()，absf()，およびストリング操作命令が使用できるライブラリ関数を命令展開します。
最適化範囲を複数に分割してコンパイルする	サイズの大きい関数について，最適化範囲を複数に分割してコンパイルするかどうかを指定します。 ライブラリ・ジェネレータのオプション -scope，-noscope に相当します。		
	デフォルト	最適化レベルオプションに依存する	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-scope)	コンパイルの前にサイズの大きい関数について，最適化範囲を複数に分割します。
		いいえ (-noscope)	コンパイルの前に最適化範囲を分割しません。
パイプライン処理を考慮した命令並べ替えを行う	パイプライン処理を考慮した命令並べ替えを行うかどうかを選択します。 ライブラリ・ジェネレータのオプション -schedule，-noschedule 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-schedule)	パイプライン処理を考慮した命令並べ替えを行います。
		いいえ (-noschedule)	命令並べ替えを行いません。

浮動小数点定数除算の乗算化を行う	浮動小数点定数除算の乗算化を行うかどうかを選択します。 ライブラリ・ジェネレータのオプション -approxdiv 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 (-approxdiv)	浮動小数点定数除算の乗算化を行います。
		いいえ	浮動小数点定数除算の乗算化を行いません。
浮動小数点型 <-> 符号無し整数型の範囲チェックを省略する	浮動小数点型 <-> 符号無し整数型の範囲チェックを省略するかどうかを選択します。 ライブラリ・ジェネレータのオプション -simple_float_conv 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 (-simple_float_conv)	浮動小数点型の型変換処理の一部を省略します。
		いいえ	浮動小数点型の型変換処理の一部を省略しません。
ポインタ指示先の型を考慮した最適化を実施する	ポインタ指示先の型を考慮した最適化を実施するかどうかを選択します。 ライブラリ・ジェネレータのオプション -alias 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 (-alias=ansi)	ANSI 規格に基づき，ポインタ指示先の型を考慮した最適化を行います。
		いいえ	ANSI 規格に基づくポインタ指示先の型を考慮した最適化を行いません。
浮動小数点式の演算順序変更の最適化を行う	浮動小数点演算式の演算順序変更の最適化を行うかどうかを選択します。 ライブラリ・ジェネレータのオプション -float_order に相当します。 また，本プロパティは，［最適化レベル］プロパティが［2 (-optimize=2)］または［Max (-optimize=max)］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 (-float_order)	浮動小数点演算式の演算順序変更の最適化を行います。
		いいえ	浮動小数点演算式の演算順序変更の最適化を行いません。

(5) ［その他］

ライブラリ・ジェネレータ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 rowspan="4">コピーライトを出力する</td><td colspan="3">コピーライトを出力するかどうかを選択します。
ライブラリ・ジェネレータのオプション -nologo に相当します。
なお，本プロパティは，［モード］カテゴリの［標準ライブラリの使用・構築方法］プロパティで［標準ライブラリ・ファイル指定なし］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">いいえ (-nologo)</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 (-logo)</td><td>コピーライトを出力します。</td></tr>
<tr><td>いいえ (-nologo)</td><td>コピーライトの出力を抑止します。</td></tr>
<tr><td rowspan="4">ライブラリ・ジェネレート前に実行するコマンド</td><td colspan="3">ライブラリ・ジェネレート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・ジェネレート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。
%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・ファイル名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，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。</td></tr>
<tr><td>デフォルト</td><td colspan="2">ライブラリ・ジェネレート前に実行するコマンド [定義数]</td></tr>
<tr><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>指定可能値</td><td colspan="2">1023 文字までの文字列
64 個まで指定可能です。</td></tr>
</table>

ライブラリ・ジェネレート後に実行するコマンド	ライブラリ・ジェネレート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・ジェネレート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。 %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・ファイル名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，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。	
	デフォルト	ライブラリ・ジェネレート後に実行するコマンド [*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	1023 文字までの文字列 64 個まで指定可能です。
その他の追加オプション	その他に追加するライブラリ・ジェネレータの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，ライブラリ・ジェネレータのオプション群の最後に付加されます。 なお，本プロパティは，［モード］カテゴリの［標準ライブラリの使用・構築方法］プロパティで［標準ライブラリ・ファイル指定なし］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，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
コマンド・ライン	指定されているオプションを表示します。 なお，本プロパティは，［モード］カテゴリの［標準ライブラリの使用・構築方法］プロパティで［標準ライブラリ・ファイル指定なし］を選択した場合は表示しません。	
	デフォルト	コマンド・ライン [*定義数*]
	変更方法	変更不可

［ビルド設定］タブ

本タブでは，各 C 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，オブジェクト・モジュール・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。

(1) ［ビルド］

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）

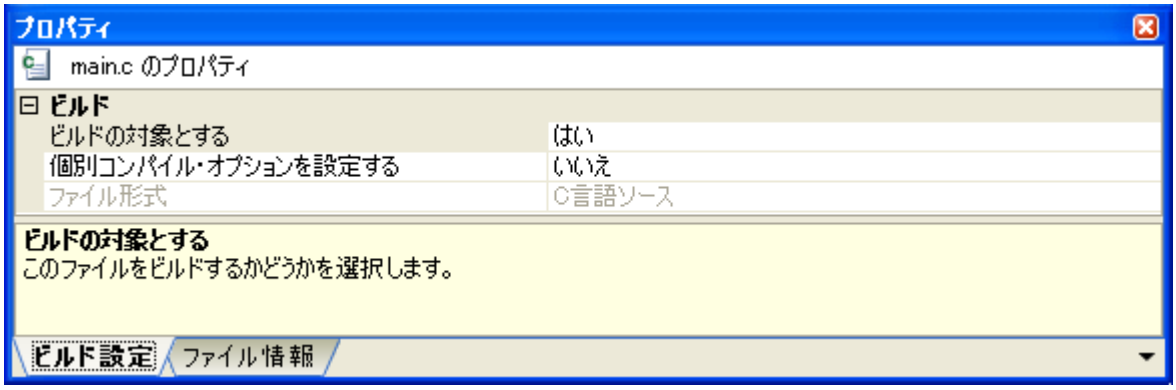


図 A.12 プロパティ パネル：［ビルド設定］タブ（C++ ソース・ファイルを選択した場合）

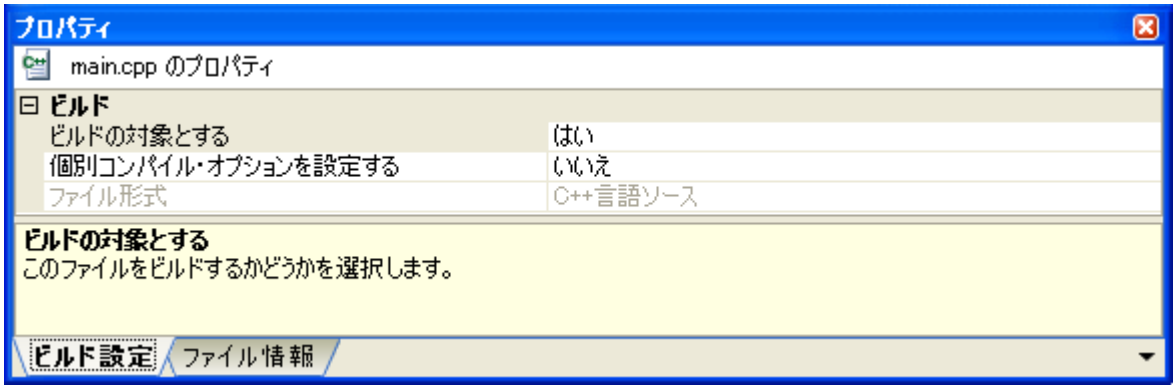


図 A.13 プロパティ パネル：［ビルド設定］タブ（アセンブラ・ソース・ファイルを選択した場合）

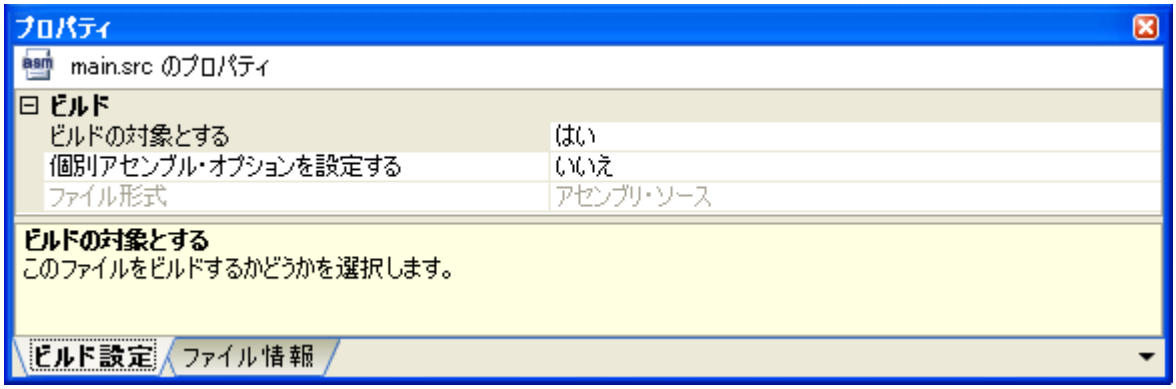


図 A.14 プロパティ パネル：[ビルド設定] タブ（オブジェクト・モジュール・ファイルを選択した場合）

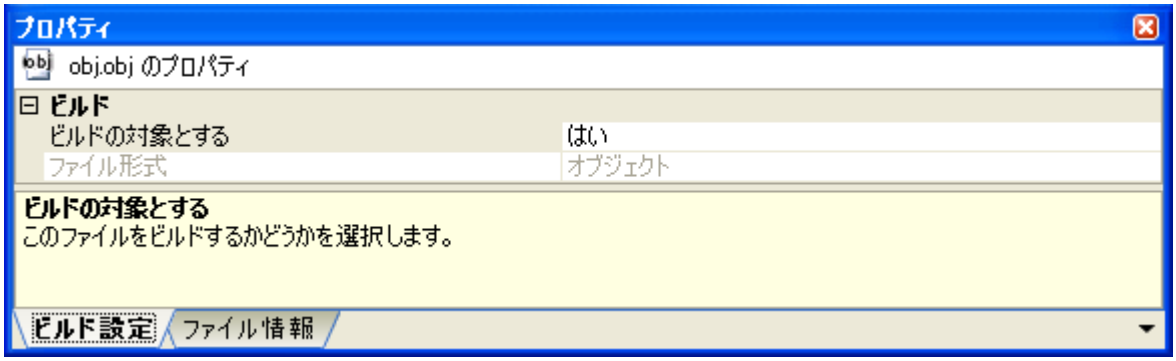
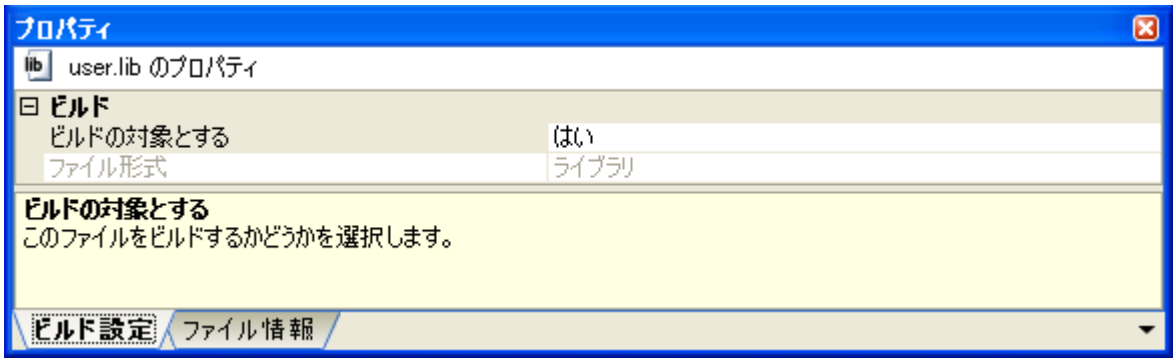


図 A.15 プロパティ パネル：[ビルド設定] タブ（ライブラリ・ファイルを選択した場合）



[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・ファイルまたは 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 パネルで C ソース・ファイルまたは C++ ソース・ファイルを選択し，[ビルド] カテゴリの [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・ファイルまたは 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・ファイルまたは 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 パネルでアセンブラ・ソース・ファイルを選択し，［ビルド］カテゴリの［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 言語ソース（C ソース・ファイルを選択している場合） C++ 言語ソース（C++ ソース・ファイルを選択している場合） アセンブリ・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（オブジェクト・モジュール・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	

［個別コンパイル・オプション (C)］タブ

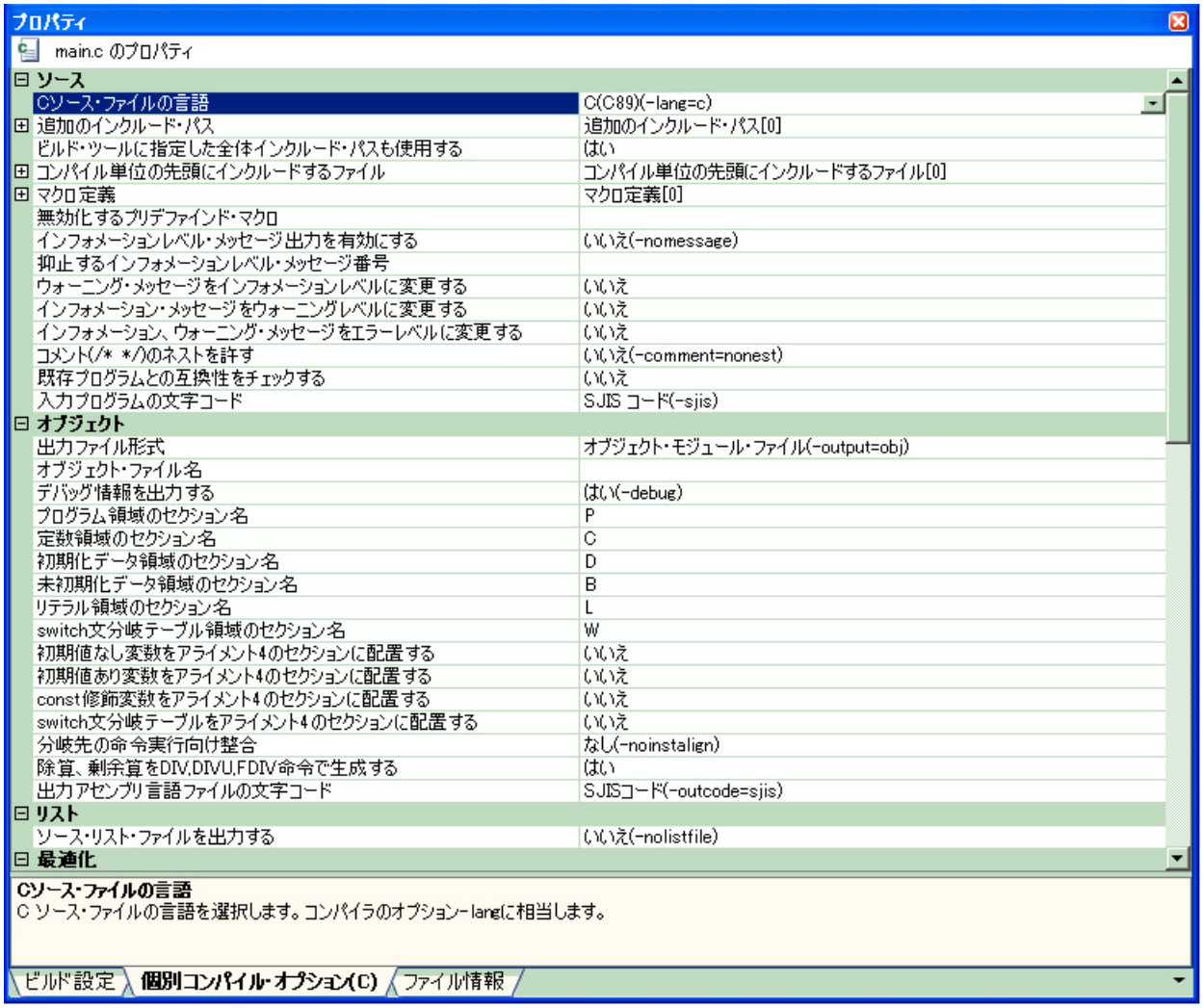
本タブでは，1 つの C ソース・ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

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.16 プロパティ パネル：［個別コンパイル・オプション (C)］タブ



［各カテゴリの説明］

(1) ［ソース］

ソー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。

C ソース・ファイルの言語	C ソース・ファイルの言語を選択します。 コンパイラのオプション -lang 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	C(C89) (-lang=c)	C(C89) 言語ソースファイルとしてコンパイルします。
		C99 (-lang=c99)	C(C99) 言語ソースファイルとしてコンパイルします。
追加のインクルード・パス	インクルード・ファイルの存在するパス名を指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 コンパイラのオプション -include 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。		
	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	247 文字までの文字列 65536 個まで指定可能です。	
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	使用するビルド・ツールの［コンパイル・オプション］タブの［ソース］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］プロパティに指定したパス ［コンパイル・オプション］タブの［ソース］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パス コンパイラのオプション -include 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	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。
		いいえ	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。

コンパイル単位の先頭にインクルードするファイル	コンパイル単位の先頭にインクルードするファイルを指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 コンパイラのオプション -preinclude 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	[...] 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	259 文字までの文字列 65536 個まで指定可能です。	
マクロ定義	定義したいマクロ名を指定します。 「*マクロ名* = *文字列*」の形式で 1 行に 1 つずつ指定します。 「= *文字列*」の部分は省略可能で，省略した場合，そのマクロ名が定義されたものと仮定します。 コンパイラのオプション -define 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	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，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	32767 文字までの文字列 65536 個まで指定可能です。	
無効化するプリデファインド・マクロ	無効化するプリデファインド・マクロを指定します。 複数指定する場合は，マクロ名をカンマで区切って指定します（例：__DBL4,__SCHAR）。 コンパイラのオプション -undefine 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	[...] ボタンをクリックし，無効化するマクロの指定 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
インフォメーションレベル・メッセージ出力を有効にする	インフォメーションレベル・メッセージ出力を有効にするかどうかを選択します。 コンパイラのオプション -message，および -nomessage 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 (-message)	インフォメーション・レベル・メッセージを有効にします。
		いいえ (-nomessage)	インフォメーション・レベル・メッセージを無効にします。

抑止するインフォメーションレベル・メッセージ番号	抑止するインフォメーション・レベル・メッセージ番号を指定します。 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：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -nomessage に相当します。 なお，本プロパティは，［インフォメーション・レベル・メッセージを抑止する］プロパティで［いいえ (-nomessage)］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，文字列入力 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
ウォーニング・メッセージをインフォメーションレベルに変更する	ウォーニング・メッセージをインフォメーションレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=information)	すべてのウォーニング・メッセージをインフォメーションレベルに変更します。
		はい (エラー番号指定) (-change_message=information=< エラー番号 >)	ウォーニングレベルの指定エラー番号のみインフォメーションレベルに変更します。
		いいえ	ウォーニング・メッセージをインフォメーションレベルに変更しません。
ウォーニングレベルのエラー番号	ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［ウォーニング・メッセージをインフォメーションレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=information=< エラー番号 >)］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，文字列入力 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	

インフォメーション・メッセージをウォーニングレベルに変更する	インフォメーション・メッセージをウォーニングレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=warning)	すべてのインフォメーション・メッセージをウォーニングレベルに変更します。
		はい (エラー番号指定) (-change_message=warning=< エラー番号 >)	インフォメーションレベルの指定エラー番号のみウォーニングレベルに変更します。
		いいえ	インフォメーション・メッセージをウォーニングレベルに変更しません。
インフォメーションレベルのエラー番号	インフォメーションレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション・メッセージをウォーニングレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=warning=< エラー番号 >)］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，文字列入力 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
インフォメーション、ウォーニング・メッセージをエラーレベルに変更する	インフォメーション，ウォーニング・メッセージをエラーレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=error))	すべてのインフォメーション，ウォーニング・メッセージをエラーレベルに変更します。
		はい (エラー番号指定) (-change_message=error=< エラー番号 >)	インフォメーション，ウォーニングレベルの指定エラー番号のみエラーレベルに変更します。
		いいえ	インフォメーション，ウォーニング・メッセージをエラーレベルに変更しません。

インフォメーション、ウォーニングレベルのエラー番号	インフォメーション，ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション、ウォーニング・メッセージをエラーレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=error=< エラー番号 >)］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，文字列入力 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
コメント (/* */) のネストを許す	コメント (/* */) のネストを許すかどうかを選択します。 コンパイラのオプション -comment 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 (-comment=nest)	コメント (/* */) のネストを許します。
		いいえ (-comment=nonest)	コメント (/* */) のネストを許しません。
既存プログラムとの互換性をチェックする	既存プログラムとの互換性をチェックするかどうかを選択します。 コンパイラのオプション -check 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 (NC コンパイラ) (-check=nc)	R8C,M16C ファミリ用 C コンパイラとの互換性をチェックします。
		はい (H8 コンパイラ) (-check=ch38)	H8,H8S,H8S ファミリ用 C/C++ コンパイラとの互換性をチェックします。
		はい (SH コンパイラ) (-check=sh)	SuperH ファミリ用 C/C++ コンパイラとの互換性をチェックします。
		いいえ	既存プログラムとの互換性をチェックしません。

入力プログラムの文字コード	入力プログラムの文字コードを選択します。 コンパイラのオプション -euc，-sjis，-latin1，-utf8，-big5，-gb2312 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	EUC コード (-euc)	文字列，文字定数，およびコメント内の文字を EUC で扱います。
		SJIS コード (-sjis)	文字列，文字定数，およびコメント内の文字を SJIS で扱います。
		ISO-Latin1 コード (-latin1)	文字列，文字定数，およびコメント内の文字を ISO-Latin1 で扱います。
		UTF-8 コード (-utf8)	文字列，文字定数，およびコメント内の文字を UTF-8 で扱います。 なお，本項目は，［C ソース・ファイルの言語］プロパティで［C(C89) (-lang=c)］を選択した場合は選択できません。
		繁体中国語 (-big5)	文字列，文字定数，およびコメント内の文字を繁体中国語で扱います。
		簡体中国語 (-gb2312)	文字列，文字定数，およびコメント内の文字を簡体中国語で扱います。

(2) ［オブジェクト］

オブジェク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。

出力ファイル形式	出力ファイル形式を選択します。 コンパイラのオプション -output 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	オブジェクト・モジュール・ファイル (-output=obj)	オブジェクト・モジュール・ファイルを出力します。
オブジェクト・ファイル名	コンパイル後に生成するオブジェクト・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。 コンパイラのオプション -output 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	
デバッグ情報を出力する	デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。コンパイラのオプション -debug，-nodebug 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 (-debug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力します。
		いいえ (-nodebug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。

プログラム領域のセクション名	プログラム領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
定数領域のセクション名	定数領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
初期化データ領域のセクション名	初期化データ領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
未初期化データ領域のセクション名	未初期化データ領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
リテラル領域のセクション名	リテラル領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
switch 文分岐テーブル領域のセクション名	switch 文分岐テーブル領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
初期値なし変数をアライメント 4 のセクションに配置する	初期値なし変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=B)	初期値なし変数をアライメント 4 のセクションに配置します。
		いいえ	初期値なし変数をアライメント 4 のセクションに配置しません。

初期値あり変数をアライメント 4 のセクションに配置する	初期値あり変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=D)	初期値あり変数をアライメント 4 のセクションに配置します。
		いいえ	初期値あり変数をアライメント 4 のセクションに配置しません。
const 修飾変数をアライメント 4 のセクションに配置する	const 修飾変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=C)	const 修飾変数をアライメント 4 のセクションに配置します。
		いいえ	const 修飾変数をアライメント 4 のセクションに配置しません。
switch 文分岐テーブルをアライメント 4 のセクションに配置する	switch 文分岐テーブルをアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=W)	switch 文分岐テーブルをアライメント 4 のセクションに配置します。
		いいえ	switch 文分岐テーブルをアライメント 4 のセクションに配置しません。

分岐先の命令実行向け整合	分岐先の命令実行向け整合を選択します。 コンパイラのオプション -noinstalign , -instalign4, -instalign8 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	なし (-noinstalign)	分岐先を命令実行向け整合をしません。
		4 バイト整合 (-instalign4)	分岐先を 4 バイトで命令実行向け整合します。
		4 バイト整合 (各ループの先頭含む) (-instalign4=loop)	分岐先を 4 バイトで命令実行向け整合します (各ループの先頭含む)。
		4 バイト整合 (各最内周ループの先頭含む) (-instalign4=inmostloop)	分岐先を 4 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
		8 バイト整合 (-instalign8)	分岐先を 8 バイトで命令実行向け整合します。
		8 バイト整合 (各ループの先頭含む) (-instalign8=loop)	分岐先を 8 バイトで命令実行向け整合します (各ループの先頭含む)。
		8 バイト整合 (各最内周ループの先頭含む) (-instalign8=inmostloop)	分岐先を 8 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
除算、剰余算を DIV,DIVU,FDIV 命令で生成する	除算，剰余算を DIV,DIVU,FDIV 命令で生成するかどうかを選択します。 コンパイラのオプション -nouse_div_inst 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	除算，剰余算に DIV,DIVU,FDIV 命令を使ったコードを生成します。
		いいえ (-nouse_div_inst)	除算，剰余算に DIV,DIVU,FDIV 命令を使わないコードを生成します。

出力アセンブリ言語ファイルの文字コード	出力アセンブリ言語ファイルの文字コードを選択します。 コンパイラのオプション -outcode 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	EUC コード (-outcode=euc)	文字列，文字定数内の文字を EUC で出力します。
		SJIS コード (-outcode=sjis)	文字列，文字定数内の文字を SJIS で出力します。
		UTF-8 コード (-outcode=utf8)	文字列，文字定数内の文字を UTF-8 で出力します。 なお，本項目は，［ソース］カテゴリの［C ソース・ファイルの言語］プロパティで［C(C89) (-lang=c)］を選択した場合は選択できません。
		繁体中国語 (-outcode=big5)	文字列，文字定数内の文字を繁体中国語で出力します。
		簡体中国語 (-outcode=gn2312)	文字列，文字定数内の文字を簡体中国語で出力します。

(3) ［リスト］

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。

ソース・リスト・ファイルを出力する	ソース・リスト・ファイルを出力するかどうかを選択します。 コンパイラのオプション -listfile ， -nolistfile 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 (-listfile)	ソース・リスト・ファイルを出力します。
		いいえ (-nolistfile)	ソース・リスト・ファイルを出力しません。
C/C++ ソースを出力する	ソース・リスト・ファイルの内容の設定を行います。 C/C++ ソースを出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=source)	C/C++ ソースを出力します。
		いいえ	C/C++ ソースを出力しません。

条件アセンブルで偽の行を出力する	ソース・リスト・ファイルの内容の設定を行います。 条件アセンブルで偽の行を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=conditionals)	条件アセンブルで偽の行を出力します。
		いいえ	条件アセンブルで偽の行を出力しません。
.DEFINE 置換前の情報を出力する	ソース・リスト・ファイルの内容の設定を行います。 .DEFINE 置換前の情報を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=definitions)	.DEFINE 置換前の情報を出力します。
		いいえ	.DEFINE 置換前の情報を出力しません。
アセンブラ・マクロ記述展開行を出力する	ソース・リスト・ファイルの内容の設定を行います。 アセンブラ・マクロ記述展開行を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=expansions)	アセンブラ・マクロ記述展開行を出力します。
		いいえ	アセンブラ・マクロ記述展開行を出力しません。

(4) ［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	最適化レベルを選択します。 コンパイラのオプション -optimize 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	0 (-optimize=0)	最適化を実施しません。
		1 (-optimize=1)	自動変数のレジスタ割り付け，関数出口ブロックの統合，統合可能な複数命令の統合など，一部最適化を実施します。
		2 (-optimize=2)	全般的に最適化を実施します。
		Max (-optimize=max)	実施可能な最適化を最大限に行います。

モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 コンパイラのオプション -goptimize 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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。
最適化方法	最適化方法を選択します。 コンパイラのオプション -speed ， -size 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	実行性能重視の最適化 (-speed)	実行性能重視の最適化を実施します。
		コード・サイズ重視の最適化 (-size)	コードサイズ重視の最適化を実施します。
ループ展開	ループ文 (for, while, do-while) を展開するかどうかを選択します。 コンパイラのオプション -loop 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	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		展開する (-loop=< 数値 >)	ループ文 (for, while, do-while) を展開します。
最大展開数	最大で何倍の展開を行うかを指定します。 コンパイラのオプション -loop に相当します。 なお，本プロパティは，［ループ展開］プロパティで［展開する (-loop=< 数値 >)］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	1 ～ 32（10 進数）	
自動インライン展開を行う	自動インライン展開を行うかどうかを選択します。 コンパイラのオプション -inline ， -noinline 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	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		はい (-inline=< 数値 >)	自動インライン展開を行います。
		いいえ (-noinline)	自動インライン展開を行いません。

関数サイズの最大増加率	関数サイズの最大増加率を指定します。 たとえば，100 を指定した場合，関数サイズが 100% 増加するまで (2 倍まで) インライン展開を行います。 コンパイラのオプション -inline に相当します。 なお，本プロパティは，［自動インライン展開を行う］プロパティで［はい (-inline=< 数値 >)］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	1 ～ 65535（10 進数）	
switch 文のコード展開方式	switch 文のコード展開方式を選択します。 コンパイラのオプション -case 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	if_then 方式 (-case=ifthen)	switch 文を if_then 方式で展開します。
		テーブル・ジャンプ方式 (-case=table)	switch 文をテーブル方式で展開します。
		コンパイラが自動選択 (-case=auto)	if_then 方式，テーブル方式いずれかをコンパイラが自動的に選択します。
外部変数を volatile 化する	すべての外部変数を volatile 宣言したものとして扱うかどうかを選択します。 コンパイラのオプション -volatile，-novolatile 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 (-volatile)	すべての外部変数を volatile 宣言したものとして扱います。
		いいえ (-novolatile)	volatile 修飾のない外部変数に対して最適化を行います。
const 宣言された外部変数の定数伝播を実施する	const 宣言された外部変数の定数伝播を実施するかどうかを選択します。 C++ 言語ソースファイルの const 修飾型変数については，本オプションで制御することはできません (常に定数伝播されます)。 コンパイラのオプション -const_copy，-noconst_copy 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-const_copy)	const 修飾型外部変数についても定数伝播を行います。
		いいえ (-noconst_copy)	const 修飾型外部変数の定数伝播を抑止します。

整数型定数による除算および剰余算の変換方法	整数型定数による除算および剰余算の変換方法を選択します。 コンパイラのオプション -const_div，-noconst_div 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	最適化方法オプションに依存する	最適化方法オプションに従います。
		乗算を用いた命令列に変換 (-const_div)	ソースファイル中の整数型定数による除算および剰余算を，乗算を用いた命令列に変換します。
		除算を用いた命令列に変換 (-noconst_div)	ソースファイル中の整数型定数による除算および剰余算を，除算を用いた命令列に変換します。
ライブラリ関数の展開方法	ライブラリ関数の展開方法を選択します。 コンパイラのオプション -library に相当します。 なお，本プロパティは，［オブジェクト］カテゴリの［出力ファイル形式］プロパティでオブジェクト・モジュール・ファイル (-output=obj) 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	すべて関数呼び出し (-library=function)	ライブラリ関数をすべて関数呼び出しします。
		一部のライブラリ関数を命令展開 (-library=intrinsic)	abs()，absf() およびストリング操作命令が使用できるライブラリ関数を命令展開します。
最適化範囲を複数に分割してコンパイルする	サイズの大きい関数について，最適化範囲を複数に分割してコンパイルするかどうかを指定します。 コンパイラのオプション -scope，-noscope 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-scope)	コンパイルの前にサイズの大きい関数について，最適化範囲を複数に分割します。
		いいえ (-noscope)	コンパイルの前に最適化範囲を分割しません。
パイプライン処理を考慮した命令並べ替えを行う	パイプライン処理を考慮した命令並べ替えを行うかどうかを選択します。 コンパイラのオプション -schedule，-noschedule 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-schedule)	パイプライン処理を考慮した命令並べ替えを行います。
		いいえ (-noschedule)	命令並べ替えを行いません。

外部変数アクセス最適化を行う	外部変数アクセス最適化を行うかどうかを選択します。 コンパイラのオプション -nomap，-smap，-map 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 (モジュール内で最適化) (-smap)	コンパイル対象ファイル内で定義された外部変数もしくは静的変数についてベースアドレスを設定し，アクセスをベースアドレス相対で行うコードを生成します。
		はい (モジュール間で最適化) (-map)	最適化リンケージエディタが生成する外部シンボル割り付け情報を元にベースアドレスを設定し，外部変数もしくは静的変数のアクセスをベースアドレス相対で行うコードを生成します。
		いいえ (-nomap)	外部変数アクセス最適化を行いません。
大域最適化を行う	大域最適化 (関数の統合など) を行うレベルを指定します。 ［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］を選択した場合は［はい (レベル 1)(最適化を行う (-ip_optimize))］、および［いいえ］のみ表示します。 コンパイラのオプション -whole_program、-merge_files、-ip_optimize 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 (レベル 1)(最適化を行う)(-ip_optimize)	ソースファイル毎に、大域最適化を行います。
		いいえ	大域最適化を行いません。
浮動小数点定数除算の乗算化を行う	浮動小数点定数除算を，定数の逆数の乗算に変換するかどうかを選択します。 コンパイラのオプション -approxdiv 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 (-approxdiv)	浮動小数点定数除算を，定数の逆数の乗算に変換します。
		いいえ	浮動小数点定数除算を，定数の逆数の乗算に変換しません。
浮動小数点型 <-> 符号無し整数型の範囲チェックを省略する	浮動小数点型 <-> 符号無し整数型の範囲チェックを省略するかどうかを選択します。 “ はい ” を選択した時，該当する型変換の処理に対するコード性能は向上しますが，変換結果が C,C++ 言語規格と異なる場合がありますので，ご注意ください。 コンパイラのオプション -simple_float_conv 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 (-simple_float_conv)	浮動小数点型の型変換処理の一部を省略します。
		いいえ	浮動小数点型の型変換処理の一部を省略しません。

ポインタ指示先の型を考慮した最適化を実施する	ポインタ指示先の型を考慮した最適化を実施するかどうかを選択します。 “ はい ” を指定した場合，一般には，alias=noansi を指定した場合よりもオブジェクト性能が向上しますが，alias=ansi と alias=noansi とで実行結果が異なる場合があります。 コンパイラのオプション -alias 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 (-alias=ansi)	ANSI 規格に基づき，ポインタ指示先の型を考慮した最適化を行います。
		いいえ (-alias=noansi)	ANSI 規格に基づくポインタ指示先の型を考慮した最適化を行いません。
浮動小数点式の演算順序変更の最適化を行う	浮動小数点演算式の演算順序変更の最適化を行うかどうかを選択します。 “ はい ” を指定した場合，一般には，-float_order を指定しない場合よりもオブジェクト性能が向上しますが，演算の精度が -float_order を指定しなかった場合と異なることがあります。 コンパイラのオプション -float_order に相当します。 また，本プロパティは，［最適化レベル］プロパティが［2 (-optimize=2)］または［Max (-optimize=max)］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 (-float_order)	浮動小数点演算式の演算順序変更の最適化を行います。
		いいえ	浮動小数点演算式の演算順序変更の最適化を行いません。

(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。

アセンブリ・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リ・ソース・ファイルを出力するかどうかを選択します。 コンパイラのオプション -output=src 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 (-output=src)	C ソースのコンパイル結果のアセンブリ・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リ・ソース・ファイルを出力しません。

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。 コンパイラのオプション -output, -noline 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 (-output=prep)	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。
		はい (#line 出力抑止)(-output=prep -noline)	ソースファイルに対して、プリプロセス処理 (#line 出力抑止) 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。

(6) ［MISRA C ルール検査］

MISRA C ルール検査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="11">適用するルール</td><td colspan="3">適用する MISRA C ルールを選択します。
コンパイラのオプション -misra2004 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 rowspan="8">指定可能値</td><td>すべてのルールを適用 (-misra2004=all)</td><td>サポートしているすべてのルールをチェック対象とします。</td></tr>
<tr><td>指定したルール番号を適用 (-misra2004=apply)</td><td>サポートしているルールのうち，指定されたルール番号をチェック対象とします。</td></tr>
<tr><td>指定したルール番号を除外 (-misra2004=ignore)</td><td>サポートしているルールのうち，指定されたルール番号以外のルールをチェック対象とします。</td></tr>
<tr><td>必須ルールを適用 (-misra2004=required)</td><td>サポートしているルールのうち，ルールの分類が “required” になっているルールをチェック対象とします。</td></tr>
<tr><td>必須ルールと指定したルールを適用 (-misra2004=required_add)</td><td>サポートしているルールのうち，ルールの分類が “required” になっているルールと指定されたルール番号をチェック対象とします。</td></tr>
<tr><td>必須ルールから指定したルール番号を除外 (-misra2004=required_remove)</td><td>サポートしているルールのうち，ルールの分類が “required” になっているルールから指定されたルール番号を除いたルール番号をチェック対象とします。</td></tr>
<tr><td>指定ファイルに記載されたルール番号を適用 (-misra2004=< ファイル名 >)</td><td>サポートしているルールのうち，指定されたファイル名に記載されたルール番号をチェック対象とします。</td></tr>
<tr><td>適用ルールなし</td><td>MISRA C ルールを適用しません。</td></tr>
<tr><td rowspan="4">ルール番号記載ファイル</td><td colspan="3">ルール番号記載ファイル (misra2004 ルール・ファイル) を指定します。
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。
%ProjectName%: 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。
コンパイラのオプション -misra2004 に相当します。
なお，本プロパティは，［適用するルール］プロパティで［指定ファイルに記載されたルール番号を適用 (-misra2004=< ファイル名 >)］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，または［...］ボタンをクリックし，Misra2004 ルール・ファイルの指定 ダイアログによる編集</td></tr>
<tr><td>指定可能値</td><td colspan="2">259 文字までの文字列</td></tr>
</table>

ルール番号	ルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［適用するルール］プロパティで［指定したルール番号を適用 (-misra2004=apply］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
除外するルール番号	除外するルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［適用するルール］プロパティで［指定したルール番号を除外 (-misra2004=ignore)］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
必須ルールの他にチェックするルール番号	必須ルールの他にチェックするルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［必須ルールと指定したルールを適用 (-misra2004=required_add)］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列
必須ルールから除外するルール番号	必須ルールの他にチェックするルール番号を指定します。 ルール番号は，必ず 10 進数値で 1 つ以上指定してください。 コンパイラのオプション -misra2004 に相当します。 なお，本プロパティは，［適用するルール］プロパティで［必須ルールから指定したルール番号を除外 (-misra2004=required_remove)］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，ルール番号の指定 ダイアログによる編集
	指定可能値	259 文字までの文字列

ルール・チェック対象外のファイル	MISRA-C2004 のルール・チェック対象外のファイルを指定します。 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。 %ProjectName%: 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。 コンパイラのオプション -ignore_files_misra に相当します。 なお，本プロパティは，［適用するルール］プロパティで［適用ルールなし］を選択した場合は表示されません。		
	デフォルト	コンパイル・オプションの設定値	
	変更方法	[...] 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	259 文字までの文字列 65536 個まで指定可能	
拡張キーワードや拡張仕様をメッセージ出力する	拡張キーワードや拡張仕様をメッセージ出力するかどうかを選択します。 コンパイラのオプション -check_language_extension に相当します。 なお，本プロパティは，［適用するルール］プロパティで［適用ルールなし］を選択した場合は表示されません。		
	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 (-check_language_extension)	C/C++ 言語規格から独自に拡張した言語仕様が有効な場合に一部抑止される，MISRA2004 ルールチェックを有効にします。
		いいえ	C/C++ 言語規格から独自に拡張した言語仕様が有効な場合に一部抑止される，MISRA2004 ルールチェックを無効にします。

(7) ［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。

コピーライトを出力する	コピーライトを出力するかどうかを選択します。 コンパイラのオプション -nologo 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 (-logo)	コピーライトを出力します。
		いいえ (-nologo)	コピーライトの出力を抑止します。
クロス・リファレンス情報を出力する	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。 本オプションを変更するには，「プログラム解析」のプロパティの設定を変更する必要があります。		
	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	はい (-Xcref)	クロス・リファレンス情報を出力します。
		いいえ	クロス・リファレンス情報を出力しません。

<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。
%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・ファイル名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，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・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>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。
%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・ファイル名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，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></td><td>デフォルト</td><td>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>[...] 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>1023 文字までの文字列
64 個まで指定可能です。</td></tr>
<tr><td>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。
なお，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されます。</td></tr>
<tr><td></td><td>デフォルト</td><td>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>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></td><td>指定可能値</td><td>259 文字までの文字列</td></tr>
<tr><td>コマンド・ライン</td><td colspan="2">指定されているオプションを表示します。</td></tr>
<tr><td></td><td>デフォルト</td><td>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>変更不可</td></tr>
</table>

［個別コンパイル・オプション (C++)］タブ

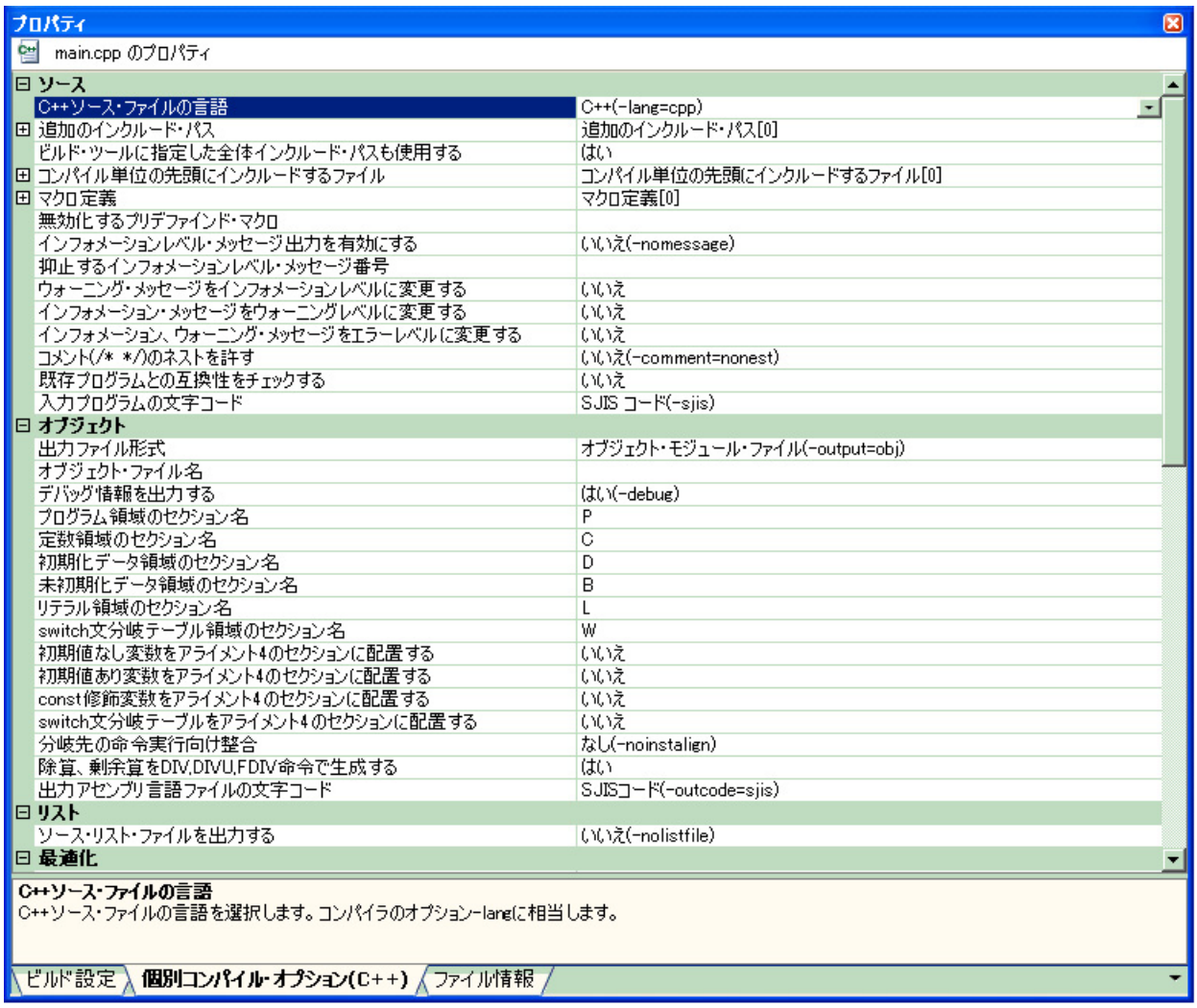
本タブでは，1 つの C++ ソース・ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

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 (C++)］タブ



［各カテゴリの説明］

(1) ［ソース］
ソー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。

C++ ソース・ファイルの言語	C++ ソース・ファイルの言語を選択します。 コンパイラのオプション -lang 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	C++ (-lang=cpp)	C++ 言語ソースファイルとしてコンパイルします。
		EC++ (-lang=ecpp)	EC++ 言語ソースファイルとしてコンパイルします。
追加のインクルード・パス	インクルード・ファイルの存在するパス名を指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 コンパイラのオプション -include 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。		
	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	247 文字までの文字列 65536 個まで指定可能です。	
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	使用するビルド・ツールの［コンパイル・オプション］タブの［ソース］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］プロパティに指定したパス ［コンパイル・オプション］タブの［ソース］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パス コンパイラのオプション -include 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	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。
		いいえ	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。

コンパイル単位の先頭にインクルードするファイル	コンパイル単位の先頭にインクルードするファイルを指定します。 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。 なお，パスはプロジェクト・フォルダを基点とします。 コンパイラのオプション -preinclude 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	[...] 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	259 文字までの文字列 65536 個まで指定可能です。	
マクロ定義	定義したいマクロ名を指定します。 「*マクロ名* = *文字列*」の形式で 1 行に 1 つずつ指定します。 「= *文字列*」の部分は省略可能で，省略した場合，そのマクロ名が定義されたものと仮定します。 コンパイラのオプション -define 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	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，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	32767 文字までの文字列 65536 個まで指定可能です。	
無効化するプリデファインド・マクロ	無効化するプリデファインド・マクロを指定します。 複数指定する場合は，マクロ名をカンマで区切って指定します（例：__DBL4,__SCHAR）。 コンパイラのオプション -undefine 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	[...] ボタンをクリックし，無効化するマクロの指定 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
インフォメーションレベル・メッセージ出力を有効にする	インフォメーションレベル・メッセージ出力を有効にするかどうかを選択します。 コンパイラのオプション -message，および -nomessage 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 (-message)	インフォメーション・レベル・メッセージを有効にします。
		いいえ (-nomessage)	インフォメーション・レベル・メッセージを無効にします。

<table>
<tr><td rowspan="4">抑止するインフォメーションレベル・メッセージ番号</td><td colspan="3">抑止するインフォメーション・レベル・メッセージ番号を指定します。
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：4,200）。
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：4,200-203,1300）。
コンパイラのオプション -nomessage に相当します。
なお，本プロパティは，［インフォメーション・レベル・メッセージを抑止する］プロパティで［いいえ (-nomessage)］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，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 colspan="2">32767 文字までの文字列</td></tr>
<tr><td rowspan="5">ウォーニング・メッセージをインフォメーションレベルに変更する</td><td colspan="3">ウォーニング・メッセージをインフォメーションレベルに変更するかどうかを選択します。
コンパイラのオプション -change_message 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 rowspan="3">指定可能値</td><td>はい (すべて) (-change_message=information)</td><td>すべてのウォーニング・メッセージをインフォメーションレベルに変更します。</td></tr>
<tr><td>はい (エラー番号指定) (-change_message=information=< エラー番号 >)</td><td>ウォーニングレベルの指定エラー番号のみインフォメーションレベルに変更します。</td></tr>
<tr><td>いいえ</td><td>ウォーニング・メッセージをインフォメーションレベルに変更しません。</td></tr>
<tr><td rowspan="4">ウォーニングレベルのエラー番号</td><td colspan="3">ウォーニングレベルのエラー番号を指定します。
複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。
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：4,200-203,1300）。
コンパイラのオプション -change_message に相当します。
なお，本プロパティは，［ウォーニング・メッセージをインフォメーションレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=information=< エラー番号 >)］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，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 colspan="2">32767 文字までの文字列</td></tr>
</table>

インフォメーション・メッセージをウォーニングレベルに変更する	インフォメーション・メッセージをウォーニングレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=warning)	すべてのインフォメーション・メッセージをウォーニングレベルに変更します。
		はい (エラー番号指定) (-change_message=warning=< エラー番号 >)	インフォメーションレベルの指定エラー番号のみウォーニングレベルに変更します。
		いいえ	インフォメーション・メッセージをウォーニングレベルに変更しません。
インフォメーションレベルのエラー番号	インフォメーションレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション・メッセージをウォーニングレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=warning=< エラー番号 >)］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，文字列入力 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
インフォメーション、ウォーニング・メッセージをエラーレベルに変更する	インフォメーション，ウォーニング・メッセージをエラーレベルに変更するかどうかを選択します。 コンパイラのオプション -change_message 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 (すべて) (-change_message=error))	すべてのインフォメーション，ウォーニング・メッセージをエラーレベルに変更します。
		はい (エラー番号指定) (-change_message=error=< エラー番号 >)	インフォメーション，ウォーニングレベルの指定エラー番号のみエラーレベルに変更します。
		いいえ	インフォメーション，ウォーニング・メッセージをエラーレベルに変更しません。

インフォメーション、ウォーニングレベルのエラー番号	インフォメーション，ウォーニングレベルのエラー番号を指定します。 複数指定する場合は，エラー番号をカンマで区切って指定します（例：4,200）。 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：4,200-203,1300）。 コンパイラのオプション -change_message に相当します。 なお，本プロパティは，［インフォメーション、ウォーニング・メッセージをエラーレベルに変更する］プロパティで［はい (エラー番号指定) (-change_message=error=< エラー番号 >)］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，文字列入力 ダイアログによる編集	
	指定可能値	32767 文字までの文字列	
コメント (/* */) のネストを許す	コメント (/* */) のネストを許すかどうかを選択します。 コンパイラのオプション -comment 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 (-comment=nest)	コメント (/* */) のネストを許します。
		いいえ (-comment=nonest)	コメント (/* */) のネストを許しません。
既存プログラムとの互換性をチェックする	既存プログラムとの互換性をチェックするかどうかを選択します。 コンパイラのオプション -check 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 (NC コンパイラ) (-check=nc)	R8C,M16C ファミリ用 C コンパイラとの互換性をチェックします。
		はい (H8 コンパイラ) (-check=ch38)	H8,H8S,H8S ファミリ用 C/C++ コンパイラとの互換性をチェックします。
		はい (SH コンパイラ) (-check=sh)	SuperH ファミリ用 C/C++ コンパイラとの互換性をチェックします。
		いいえ	既存プログラムとの互換性をチェックしません。

入力プログラムの文字コード	入力プログラムの文字コードを選択します。 コンパイラのオプション -euc，-sjis，-latin1，-utf8，-big5，-gb2312 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	EUC コード (-euc)	文字列，文字定数およびコメント内の文字を EUC で扱います。
		SJIS コード (-sjis)	文字列，文字定数およびコメント内の文字を SJIS で扱います。
		ISO-Latin1 コード (-latin1)	文字列，文字定数およびコメント内の文字を ISO-Latin1 で扱います。
		UTF-8 コード (-utf8)	文字列，文字定数およびコメント内の文字を UTF-8 で扱います。 なお，本項目は，［C ソース・ファイルの言語］プロパティで［C(C89) (-lang=c)］を選択した場合は選択できません。
		繁体中国語 (-big5)	文字列，文字定数，およびコメント内の文字を繁体中国語で扱います。
		簡体中国語 (-gb2312)	文字列，文字定数，およびコメント内の文字を簡体中国語で扱います。

(2) ［オブジェクト］

オブジェク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。

出力ファイル形式	出力ファイル形式を選択します。 コンパイラのオプション -output 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	オブジェクト・モジュール・ファイル (-output=obj)	オブジェクト・モジュール・ファイルを出力します。
		プリプロセッサ展開後のソース・ファイル (-output=prep)	プリプロセッサ展開後のソースファイルを出力します。
		プリプロセッサ展開後のソース・ファイル (#line 出力抑止) (-output=prep -noline)	プリプロセッサ展開時に #line 出力を抑止します。
オブジェクト・ファイル名	コンパイル後に生成するオブジェクト・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。 コンパイラのオプション -output 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	

デバッグ情報を出力する	デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。コンパイラのオプション -debug, -nodebug 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 (-debug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力します。
		いいえ (-nodebug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。
プログラム領域のセクション名	プログラム領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
定数領域のセクション名	定数領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
初期化データ領域のセクション名	初期化データ領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
未初期化データ領域のセクション名	未初期化データ領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
リテラル領域のセクション名	リテラル領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	
switch 文分岐テーブル領域のセクション名	switch 文分岐テーブル領域のセクション名を指定します。 コンパイラのオプション -section 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	テキスト・ボックスによる直接入力	
	指定可能値	32767 文字までの文字列	

初期値なし変数をアライメント 4 のセクションに配置する	初期値なし変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=B)	初期値なし変数をアライメント 4 のセクションに配置します。
		いいえ	初期値なし変数をアライメント 4 のセクションに配置しません。
初期値あり変数をアライメント 4 のセクションに配置する	初期値あり変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=D)	初期値あり変数をアライメント 4 のセクションに配置します。
		いいえ	初期値あり変数をアライメント 4 のセクションに配置しません。
const 修飾変数をアライメント 4 のセクションに配置する	const 修飾変数をアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=C)	const 修飾変数をアライメント 4 のセクションに配置します。
		いいえ	const 修飾変数をアライメント 4 のセクションに配置しません。
switch 文分岐テーブルをアライメント 4 のセクションに配置する	switch 文分岐テーブルをアライメント 4 のセクションに配置するかどうかを選択します。 コンパイラのオプション -nostuff 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 (-nostuff=W)	switch 文分岐テーブルをアライメント 4 のセクションに配置します。
		いいえ	switch 文分岐テーブルをアライメント 4 のセクションに配置しません。

分岐先の命令実行向け整合	分岐先の命令実行向け整合を選択します。 コンパイラのオプション -noinstalign，-instalign4，-instalign8 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	なし (-noinstalign)	分岐先を命令実行向け整合をしません。
		4 バイト整合 (-instalign4)	分岐先を 4 バイトで命令実行向け整合します。
		4 バイト整合 (各ループの先頭含む) (-instalign4=loop)	分岐先を 4 バイトで命令実行向け整合します (各ループの先頭含む)。
		4 バイト整合 (各最内周ループの先頭含む) (-instalign4=inmostloop)	分岐先を 4 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
		8 バイト整合 (-instalign8)	分岐先を 8 バイトで命令実行向け整合します。
		8 バイト整合 (各ループの先頭含む) (-instalign8=loop)	分岐先を 8 バイトで命令実行向け整合します (各ループの先頭含む)。
		8 バイト整合 (各最内周ループの先頭含む) (-instalign8=inmostloop)	分岐先を 8 バイトで命令実行向け整合します (各最内周ループの先頭含む)。
除算、剰余算を DIV,DIVU,FDIV 命令で生成する	除算，剰余算を DIV,DIVU,FDIV 命令で生成するかどうかを選択します。 コンパイラのオプション -nouse_div_inst 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	除算，剰余算に DIV,DIVU,FDIV 命令を使ったコードを生成します。
		いいえ (-nouse_div_inst)	除算，剰余算に DIV,DIVU,FDIV 命令を使わないコードを生成します。

出力アセンブリ言語ファイルの文字コード	出力アセンブリ言語ファイルの文字コードを選択します。 コンパイラのオプション -outcode 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	EUC コード (-outcode=euc)	文字列，文字定数内の文字を EUC で出力します。
		SJIS コード (-outcode=sjis)	文字列，文字定数内の文字を SJIS で出力します。
		UTF-8 コード (-outcode=utf8)	文字列，文字定数内の文字を UTF-8 で出力します。 なお，本項目は，［ソース］カテゴリの［C ソース・ファイルの言語］プロパティで［C(C89) (-lang=c)］を選択した場合は選択できません。
		繁体中国語 (-outcode=big5)	文字列，文字定数内の文字を繁体中国語で出力します。
		簡体中国語 (-outcode=gn2312)	文字列，文字定数内の文字を簡体中国語で出力します。

(3) ［リスト］

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。

ソース・リスト・ファイルを出力する	ソース・リスト・ファイルを出力するかどうかを選択します。 コンパイラのオプション -listfile ， -nolistfile 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 (-listfile)	ソース・リスト・ファイルを出力します。
		いいえ (-nolistfile)	ソース・リスト・ファイルを出力しません。
C/C++ ソースを出力する	ソース・リスト・ファイルの内容の設定を行います。 C/C++ ソースを出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=source)	C/C++ ソースを出力します。
		いいえ	C/C++ ソースを出力しません。

条件アセンブルで偽の行を出力する	ソース・リスト・ファイルの内容の設定を行います。 条件アセンブルで偽の行を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=conditionals)	条件アセンブルで偽の行を出力します。
		いいえ	条件アセンブルで偽の行を出力しません。
.DEFINE 置換前の情報を出力する	ソース・リスト・ファイルの内容の設定を行います。 .DEFINE 置換前の情報を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=definitions)	.DEFINE 置換前の情報を出力します。
		いいえ	.DEFINE 置換前の情報を出力しません。
アセンブラ・マクロ記述展開行を出力する	ソース・リスト・ファイルの内容の設定を行います。 アセンブラ・マクロ記述展開行を出力するかどうかを選択します。 コンパイラのオプション -show に相当します。 なお，本プロパティは，［ソース・リスト・ファイルを出力する］プロパティで［はい (-listfile］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 (-show=expansions)	アセンブラ・マクロ記述展開行を出力します。
		いいえ	アセンブラ・マクロ記述展開行を出力しません。

(4) ［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	最適化レベルを選択します。 コンパイラのオプション -optimize 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	0 (-optimize=0)	最適化を実施しません。
		1 (-optimize=1)	自動変数のレジスタ割り付け，関数出口ブロックの統合，統合可能な複数命令の統合など，一部最適化を実施します。
		2 (-optimize=2)	全般的に最適化を実施します。
		Max (-optimize=max)	実施可能な最適化を最大限に行います。
モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 コンパイラのオプション -goptimize 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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。
最適化方法	最適化方法を選択します。 コンパイラのオプション -speed ， -size 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	実行性能重視の最適化 (-speed)	実行性能重視の最適化を実施します。
		コード・サイズ重視の最適化 (-size)	コードサイズ重視の最適化を実施します。
ループ展開	ループ文 (for, while, do-while) を展開するかどうかを選択します。 コンパイラのオプション -loop 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	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		展開する (-loop=< 数値 >)	ループ文 (for, while, do-while) を展開します。
最大展開数	最大で何倍の展開を行うかを指定します。 コンパイラのオプション -loop に相当します。 なお，本プロパティは，［ループ展開］プロパティで［展開する (-loop=< 数値 >)］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	1 ～ 32（10 進数）	

自動インライン展開を行う	自動インライン展開を行うかどうかを選択します。 コンパイラのオプション -inline，-noinline 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	最適化レベル，最適化方法オプションに依存する	最適化レベル，最適化方法オプションの指定に従います。
		はい (-inline=< 数値 >)	自動インライン展開を行います。
		いいえ (-noinline)	自動インライン展開を行いません。
関数サイズの最大増加率	関数サイズの最大増加率を指定します。 たとえば，100 を指定した場合，関数サイズが 100% 増加するまで (2 倍まで) インライン展開を行います。 コンパイラのオプション -inline に相当します。 なお，本プロパティは，［自動インライン展開を行う］プロパティで［はい (-inline=< 数値 >)］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	1 ～ 65535（10 進数）	
switch 文のコード展開方式	switch 文のコード展開方式を選択します。 コンパイラのオプション -case 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	if_then 方式 (-case=ifthen)	switch 文を if_then 方式で展開します。
		テーブル・ジャンプ方式 (-case=table)	switch 文をテーブル方式で展開します。
		コンパイラが自動選択 (-case=auto)	if_then 方式，テーブル方式いずれかをコンパイラが自動的に選択します。
外部変数を volatile 化する	すべての外部変数を volatile 宣言したものとして扱うかどうかを選択します。 コンパイラのオプション -volatile，-novolatile 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 (-volatile)	すべての外部変数を volatile 宣言したものとして扱います。
		いいえ (-novolatile)	volatile 修飾のない外部変数に対して最適化を行います。

const 宣言された外部変数の定数伝播を実施する	const 宣言された外部変数の定数伝播を実施するかどうかを選択します。 C++ 言語ソースファイルの const 修飾型変数については，本オプションで制御することはできません (常に定数伝播されます)。 コンパイラのオプション -const_copy，-noconst_copy 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-const_copy)	const 修飾型外部変数についても定数伝播を行います。
		いいえ (-noconst_copy)	const 修飾型外部変数の定数伝播を抑止します。
整数型定数による除算および剰余算の変換方法	整数型定数による除算および剰余算の変換方法を選択します。 コンパイラのオプション -const_div，-noconst_div 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	最適化方法オプションに依存する	最適化方法オプションに従います。
		乗算を用いた命令列に変換 (-const_div)	ソースファイル中の整数型定数による除算および剰余算を，乗算を用いた命令列に変換します。
		除算を用いた命令列に変換 (-noconst_div)	ソースファイル中の整数型定数による除算および剰余算を，除算を用いた命令列に変換します。
ライブラリ関数の展開方法	ライブラリ関数の展開方法を選択します。 コンパイラのオプション -library 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	すべて関数呼び出し (-library=function)	ライブラリ関数をすべて関数呼び出しします。
		一部のライブラリ関数を命令展開 (-library=intrinsic)	abs()，absf() およびストリング操作命令が使用できるライブラリ関数を命令展開します。
最適化範囲を複数に分割してコンパイルする	サイズの大きい関数について，最適化範囲を複数に分割してコンパイルするかどうかを指定します。 コンパイラのオプション -scope，-noscope 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-scope)	コンパイルの前にサイズの大きい関数について，最適化範囲を複数に分割します。
		いいえ (-noscope)	コンパイルの前に最適化範囲を分割しません。

パイプライン処理を考慮した命令並べ替えを行う	パイプライン処理を考慮した命令並べ替えを行うかどうかを選択します。 コンパイラのオプション -schedule，-noschedule 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	最適化レベルオプションに依存する	最適化レベルオプションに従います。
		はい (-schedule)	パイプライン処理を考慮した命令並べ替えを行います。
		いいえ (-noschedule)	命令並べ替えを行いません。
外部変数アクセス最適化を行う	外部変数アクセス最適化を行うかどうかを選択します。 コンパイラのオプション -nomap，-smap，-map 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 (モジュール内で最適化) (-smap)	コンパイル対象ファイル内で定義された外部変数もしくは静的変数についてベースアドレスを設定し，アクセスをベースアドレス相対で行うコードを生成します。
		はい (モジュール間で最適化) (-map)	最適化リンケージエディタが生成する外部シンボル割り付け情報を元にベースアドレスを設定し，外部変数もしくは静的変数のアクセスをベースアドレス相対で行うコードを生成します。
		いいえ (-nomap)	外部変数アクセス最適化を行いません。
大域最適化を行う	大域最適化 (関数の統合など) を行うレベルを指定します。 ［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］を選択した場合は［はい (レベル 1)(最適化を行う (-ip_optimize))］、および［いいえ］のみ表示します。 コンパイラのオプション -whole_program、-merge_files、-ip_optimize 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 (レベル 1)(最適化を行う)(-ip_optimize)	ソースファイル毎に、大域最適化を行います。
		いいえ	大域最適化を行いません。
浮動小数点定数除算の乗算化を行う	浮動小数点定数除算を，定数の逆数の乗算に変換するかどうかを選択します。 コンパイラのオプション -approxdiv 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 (-approxdiv)	浮動小数点定数除算を，定数の逆数の乗算に変換します。
		いいえ	浮動小数点定数除算を，定数の逆数の乗算に変換しません。

浮動小数点型 <-> 符号無し整数型の範囲チェックを省略する	浮動小数点型 <-> 符号無し整数型の範囲チェックを省略するかどうかを選択します。 “ はい ” を選択した時，該当する型変換の処理に対するコード性能は向上しますが，変換結果が C,C++ 言語規格と異なる場合がありますので，ご注意ください。 コンパイラのオプション -simple_float_conv 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 (-simple_float_conv)	浮動小数点型の型変換処理の一部を省略します。
		いいえ	浮動小数点型の型変換処理の一部を省略しません。
ポインタ指示先の型を考慮した最適化を実施する	ポインタ指示先の型を考慮した最適化を実施するかどうかを選択します。 “ はい ” を指定した場合，一般には，alias=noansi を指定した場合よりもオブジェクト性能が向上しますが，alias=ansi と alias=noansi とで実行結果が異なる場合があります。 コンパイラのオプション -alias 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 (-alias=ansi)	ANSI 規格に基づき，ポインタ指示先の型を考慮した最適化を行います。
		いいえ (-alias=noansi)	ANSI 規格に基づくポインタ指示先の型を考慮した最適化を行いません。
浮動小数点式の演算順序変更の最適化を行う	浮動小数点演算式の演算順序変更の最適化を行うかどうかを選択します。 “ はい ” を指定した場合，一般には，-float_order を指定しない場合よりもオブジェクト性能が向上しますが，演算の精度が -float_order を指定しなかった場合と異なることがあります。 コンパイラのオプション -float_order に相当します。 また，本プロパティは，［最適化レベル］プロパティが［2 (-optimize=2)］または［Max (-optimize=max)］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 (-float_order)	浮動小数点演算式の演算順序変更の最適化を行います。
		いいえ	浮動小数点演算式の演算順序変更の最適化を行いません。

(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。

アセンブリ・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リ・ソース・ファイルを出力するかどうかを選択します。 コンパイラのオプション -output=src 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 (-output=src)	C ソースのコンパイル結果のアセンブリ・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リ・ソース・ファイルを出力しません。

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。 コンパイラのオプション -output, -noline 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 (-output=prep)	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。
		はい (#line 出力抑止)(-output=prep -noline)	ソースファイルに対して、プリプロセス処理 (#line 出力抑止) 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。

(6) ［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。

コピーライトを出力する	コピーライトを出力するかどうかを選択します。 コンパイラのオプション -nologo 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 (-logo)	コピーライトを出力します。
		いいえ (-nologo)	コピーライトの出力を抑止します。
クロス・リファレンス情報を出力する	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。 本オプションを変更するには，「プログラム解析」のプロパティの設定を変更する必要があります。		
	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	はい (-Xcref)	クロス・リファレンス情報を出力します。
		いいえ	クロス・リファレンス情報を出力しません。

<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。
%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・ファイル名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，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・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>

コンパイル後に実行するコマンド	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。 %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・ファイル名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，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。	
	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，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	1023 文字までの文字列 64 個まで指定可能です。
その他の追加オプション	その他に追加するコンパイル・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，コンパイル・オプション群の最後に付加されます。	
	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，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
コマンド・ライン	指定されているオプションを表示します。	
	デフォルト	コマンド・ライン [*定義数*]
	変更方法	変更不可

［個別アセンブル・オプション］タブ

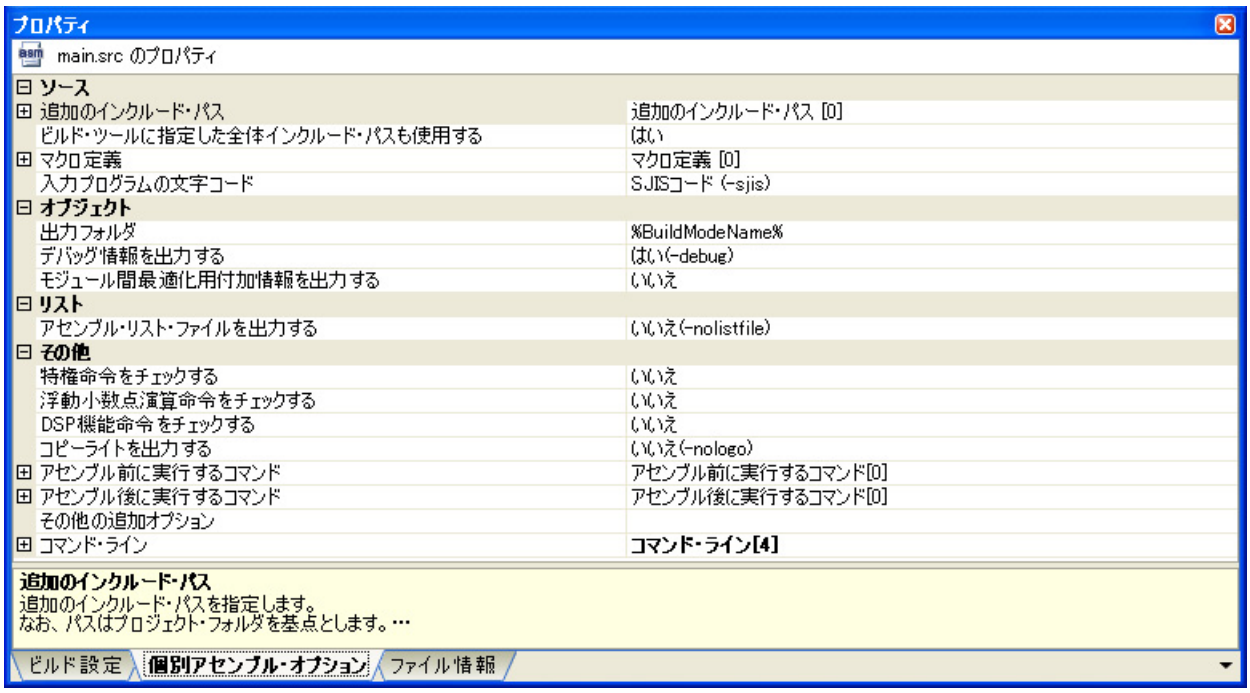
本タブでは，1 つのアセンブラ・ソース・ファイルに対して，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

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。

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］を選択した場合に表示されます。

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) ［ソース］

ソー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 colspan="3">インクルード・ファイルの存在するパス名を指定します。
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。
なお，パスはプロジェクト・フォルダを基点とします。
アセンブラのオプション -include 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。</td></tr>
<tr><td></td><td>デフォルト</td><td colspan="2">追加のインクルード・パス [定義数]</td></tr>
<tr><td></td><td>変更方法</td><td colspan="2">［...］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 colspan="2">247 文字までの文字列
65536 個まで指定可能です。</td></tr>
<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［ソース］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。
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］プロパティに指定したパス
［アセンブル・オプション］タブの［ソース］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パス
アセンブラのオプション -include 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 rowspan="2">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>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>マクロ定義</td><td colspan="3">定義したいマクロ名を指定します。
「マクロ名 = 文字列」の形式で 1 行に 1 つずつ指定します。
アセンブラのオプション -define 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">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">32767 文字までの文字列
65536 個まで指定可能です。</td></tr>
</table>

入力プログラムの文字コード	入力プログラムの文字コードを選択します。 アセンブラのオプション -euc，-sjis，-latin1，-big5，-gb2312 に相当します。		
	デフォルト	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	EUC コード (-euc)	文字列，文字定数，およびコメント内の文字を EUC で扱います。
		SJIS コード (-sjis)	文字列，文字定数，およびコメント内の文字を SJIS で扱います。
		ISO-Latin1 コード (-latin1)	文字列，文字定数，およびコメント内の文字を ISO-Latin1 で扱います。
		繁体中国語 (-big5)	文字列，文字定数，およびコメント内の文字を繁体中国語で扱います。
		簡体中国語 (-gb2312)	文字列，文字定数，およびコメント内の文字を簡体中国語で扱います。

(2) ［オブジェクト］

オブジェク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。

オブジェクト・ファイル名	アセンブル後に生成するオブジェクト・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。 アセンブラのオプション -output 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，フォルダの参照 ダイアログによる編集	
	指定可能値	259 文字までの文字列	
デバッグ情報を出力する	デバッグ情報をオブジェクト・モジュール・ファイルに出力するかどうかを選択します。アセンブラのオプション -debug，-nodebug に相当します。		
	デフォルト	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 (-debug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力します。
		いいえ (-nodebug)	デバッグ情報をオブジェクト・モジュール・ファイルに出力しません。

(3) ［リスト］

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。 アセンブラのオプション -listfile，-nolistfile に相当します。		
	デフォルト	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 (-listfile)	アセンブル・リスト・ファイルを出力します。
		いいえ (-nolistfile)	アセンブル・リスト・ファイルを出力しません。
条件アセンブルで偽の行を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 条件アセンブルで偽の行を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］を選択した場合のみ表示します。		
	デフォルト	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 (-show=conditionals)	条件アセンブルで条件が偽となった行を出力します。
		いいえ	条件アセンブルで条件が偽となった行を出力しません。
.DEFINE 置換前の情報を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 .DEFINE 置換前の情報を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］を選択した場合のみ表示します。		
	デフォルト	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 (-show=definitions)	.DEFINE 置換前の情報を出力します。
		いいえ	.DEFINE 置換前の情報を出力しません。
アセンブラ・マクロ記述展開行を出力する	アセンブル・リスト・ファイルの内容の設定を行います。 アセンブラ・マクロ記述展開行を出力するかどうかを選択します。 アセンブラのオプション -show 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 (-listfile)］を選択した場合のみ表示します。		
	デフォルト	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 (-show=expansions)	アセンブラ・マクロ記述展開行を出力します。
		いいえ	アセンブラ・マクロ記述展開行を出力しません。

(4) ［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。

モジュール間最適化用付加情報を出力する	モジュール間最適化用付加情報を出力するかどうかを選択します。 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。 アセンブラのオプション -goptimize 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 (-goptimize)	モジュール間最適化用付加情報を出力します。
		いいえ	モジュール間最適化用付加情報を出力しません。

(5) ［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。

特権命令をチェックする	特権命令をチェックするかどうかを選択します。 アセンブラのオプション -chkpm に相当します。		
	デフォルト	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 (-chkpm)	特権命令をチェックします。
		いいえ	特権命令をチェックしません。
浮動小数点演算命令をチェックする	浮動小数点演算命令をチェックするかどうかを選択します。 アセンブラのオプション -chkfpu に相当します。		
	デフォルト	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 (-chkfpu)	浮動小数点演算命令をチェックします。
		いいえ	浮動小数点演算命令をチェックしません。
DSP 機能命令をチェックする	DSP 機能命令をチェックするかどうかを選択します。 アセンブラのオプション -chkdsp に相当します。		
	デフォルト	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 (-chkdsp)	DSP 機能命令をチェックします。
		いいえ	DSP 機能命令をチェックしません。
コピーライトを出力する	コピーライトを出力するかどうかを選択します。 アセンブラのオプション -logo，-nologo に相当します。		
	デフォルト	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 (-logo)	コピーライトを出力します。
		いいえ (-nologo)	コピーライトの出力を抑止します。

<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。
%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・ファイル名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，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・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>

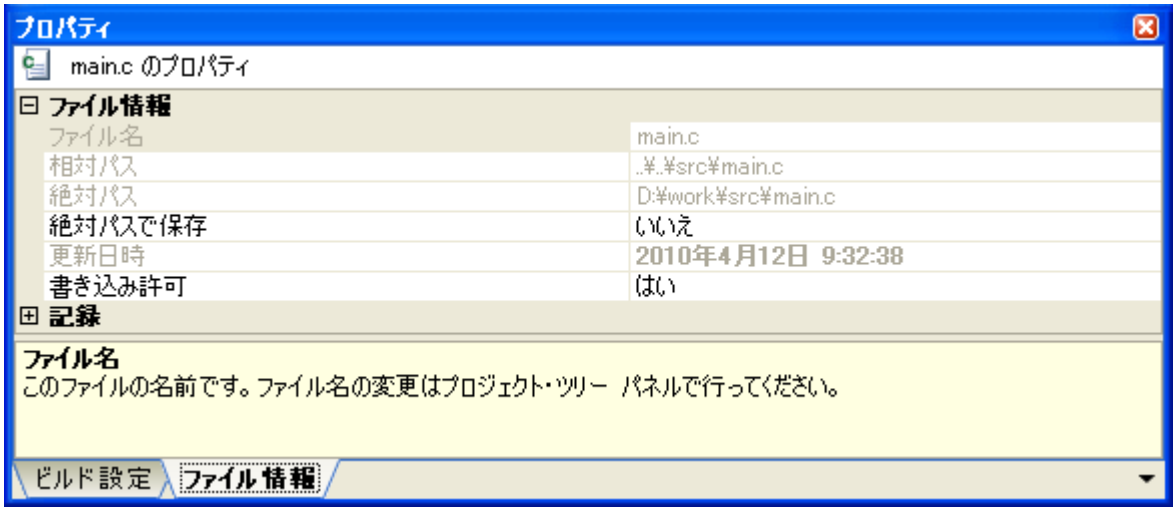
<table>
<tr><td rowspan="4">アセンブル後に実行するコマンド</td><td colspan="2">アセンブル処理後に実行するコマンドを指定します。
バッチファイルを指定する場合は，call 命令を使用してください（例：call a.bat）。
次のプレースホルダに対応しています。
%ActiveProjectDir%:アクティブ・プロジェクト・フォルダの絶対パスに置換します。
%ActiveProjectName%:アクティブ・プロジェクト名に置換します。
%AssembledFile%:アセンブル時の出力ファイルの絶対パスに置換します。
%BuildModeName%: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。
%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・ファイル名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，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・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ラの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，アセンブラのオプション群の最後に付加されます。</td></tr>
<tr><td>デフォルト</td><td>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>
<tr><td rowspan="3">コマンド・ライン</td><td colspan="2">指定されているオプションを表示します。</td></tr>
<tr><td>デフォルト</td><td>アセンブル・オプションの設定値</td></tr>
<tr><td>変更方法</td><td>変更不可</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。

(1) ［ファイル情報］
(2) ［記録］

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 パネルで行ってください。		
	デフォルト	ファイル名	
	変更方法	変更不可	
相対パス	ファイルのプロジェクト・フォルダからの相対パス名を表示します。		
	デフォルト	ファイルのプロジェクト・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。		
	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	ファイルが最後に変更され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。		
	デフォルト	はい（ファイルに書き込みが許可されている場合） いいえ（ファイルに書き込みが許可されていない場合）	
	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］

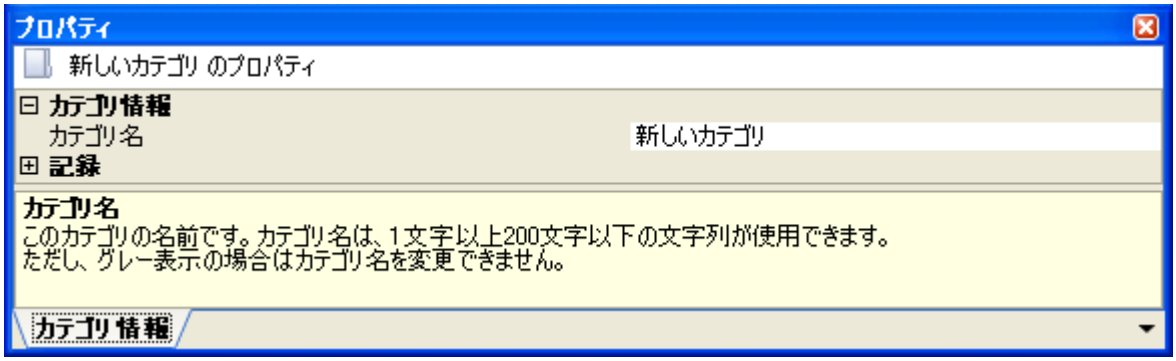
記録に関する詳細情報の表示，および設定の変更を行います。

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

(2) ［記録］

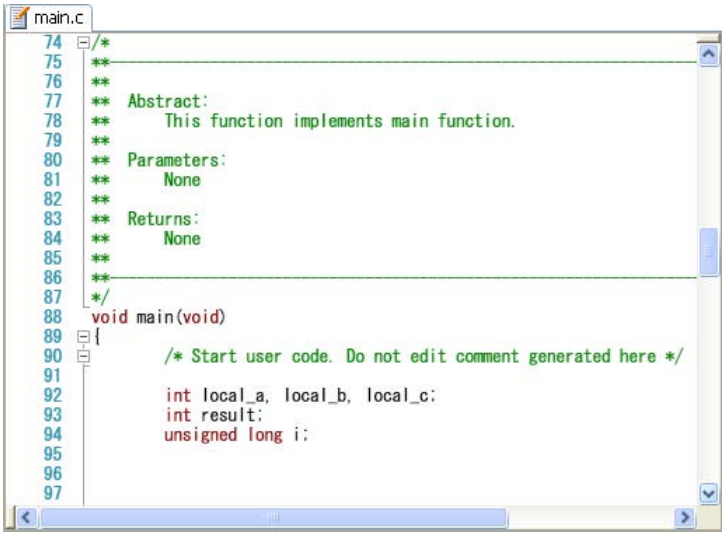
記録に関する詳細情報の表示，および設定の変更を行います。
なお，本カテゴリは，ファイル・ノード，ビルド・ツール生成ファイル・ノードについては表示されません。

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+ RX コーディング編」を参照してください。

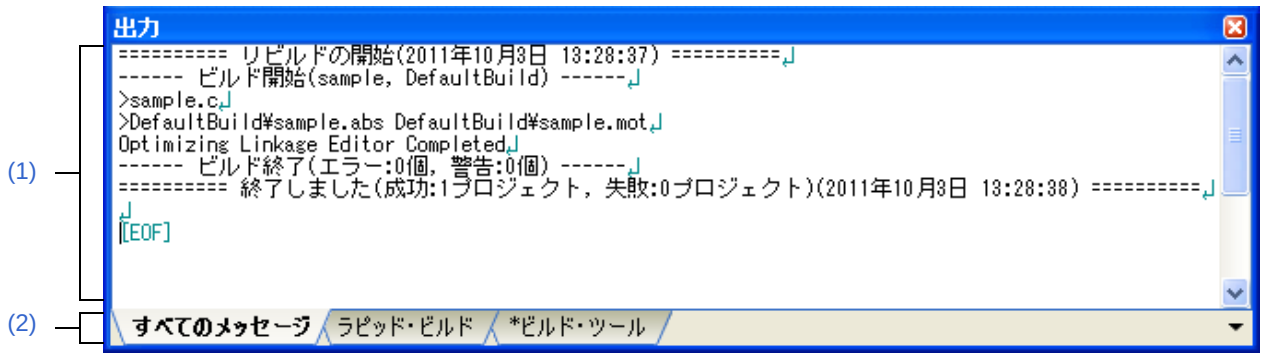
図 A.21 エディタ パネル



出力 パネル

ビルド・ツールから出力され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されたタブ上でそれぞれ個別に表示されます。

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］を選択

［各エリアの説明］

(1) メッセージ・エリア
各ツールから出力されたメッセージ，および検索結果を表示します。
ビルド結果／検索結果（一括検索）の表示では，ビルド／検索を行うごとに，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が出力された場合は，古い行から削除されます。

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されます。
		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されます。
		背景色	標準色	

メッセージ種別	表示例（デフォルト）			説明
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されます。
		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
出力され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，ビルド時に出力されたエラー・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され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。
表示され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・ツールから出力され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・ツールから出力されたメッセージを表示します。

注意 新たなメッセージが非選択状態のタブ上に出力されても，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，新たなメッセージが出力され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されている内容を，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されている内容を，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。

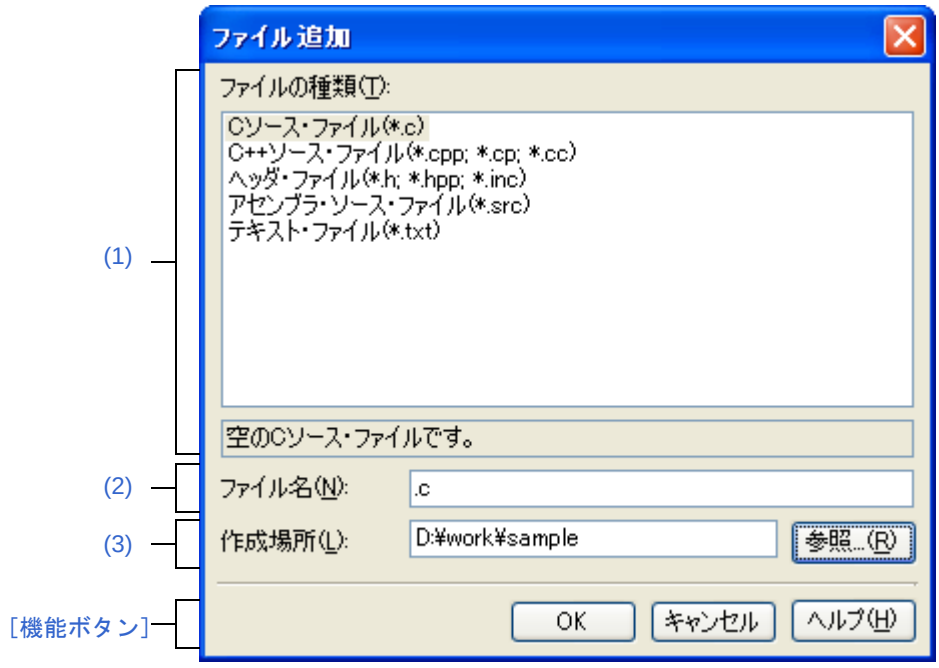
[コンテキスト・メニュー]

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	キャレット行のメッセージに対応するエディタ（ファイル，行，桁）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。
表示され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）
- C++ ソース・ファイル（*.cpp; *.cp; *.cc）
- ヘッダ・ファイル（*.h; *.hpp; *.inc）
- アセンブラ・ソース・ファイル（*.src; *.s）
- 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が付加されます。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が付加されます（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。

(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・ボックスにパスが追加されます。

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。

備考 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，以下のメッセージが［ファイル名］エリアにツールチップ表示されます。

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が指定されました。ファイル名，およびフォルダ名に文字 (￥, /, :, *, ?, ", <, >, |) は使用できません。

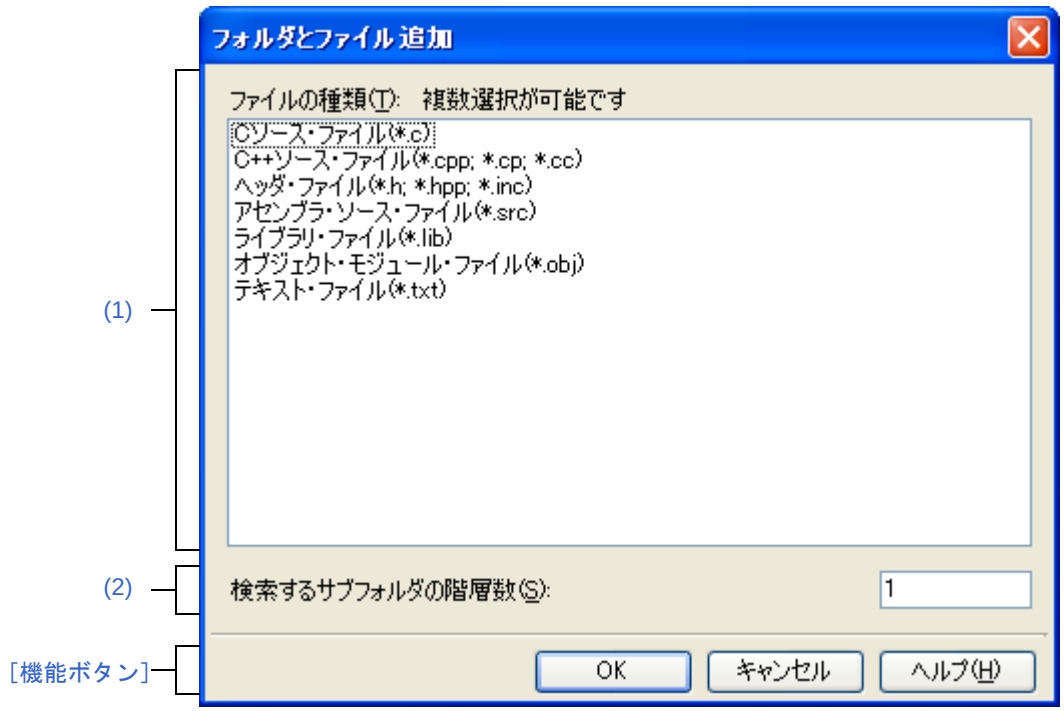
［機能ボタン］

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。
表示され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）
- C++ ソース・ファイル（*.cpp; *.cp; *.cc）
- ヘッダ・ファイル（*.h; *.hpp; *.inc）
- アセンブラ・ソース・ファイル（*.src; *.s）
- ジャンプ・テーブル・ファイル（*.jmp）
- シンボル・アドレス・ファイル（*.fsy）
- ライブラリ・ファイル（*.lib）
- オブジェクト・モジュール・ファイル（*.obj）
- リロケータブル・ファイル（*.rel）
- 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，以下のメッセージがツールチップ表示されます。

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が指定されました。

［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［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］
- [コンパイル・オプション] タブの［ソース］カテゴリの［抑止するインフォメーションレベル・メッセージ番号］，［ウォーニングレベルのエラー番号］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニングレベルのエラー番号］，［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［入力］カテゴリの［実行開始アドレス］，［出力］カテゴリの［抑止するインフォメーションレベル・メッセージ番号］，［可変ベクタの空き領域のアドレス］，［その他］カテゴリの［ウォーニングレベルのエラー番号］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニングレベルのエラー番号］，［その他の追加オプション］，［その他の追加オプション (Hex/Srecord/Binary data)］
- [ライブラリアン・オプション] タブの［出力］カテゴリの［抑止するインフォメーションレベル・メッセージ番号］，［その他］カテゴリの［ウォーニングレベルのエラー番号］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニングレベルのエラー番号］，［その他の追加オプション］
- [ライブラリ・ジェネレート・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 (C)] タブの［ソース］カテゴリの［抑止するインフォメーションレベル・メッセージ番号］，［ウォーニングレベルのエラー番号］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニングレベルのエラー番号］，［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 (C++)] タブの［ソース］カテゴリの［抑止するインフォメーションレベル・メッセージ番号］，［ウォーニングレベルのエラー番号］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニングレベルのエラー番号］，［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］
- オプション ダイアログの［全般 - 外部ツール］カテゴリにおいて，新規登録エリアの［起動時に引数を入力する］をチェックしたのち，［ツール］メニューより外部ツールの起動時に自動的にオープン

[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）

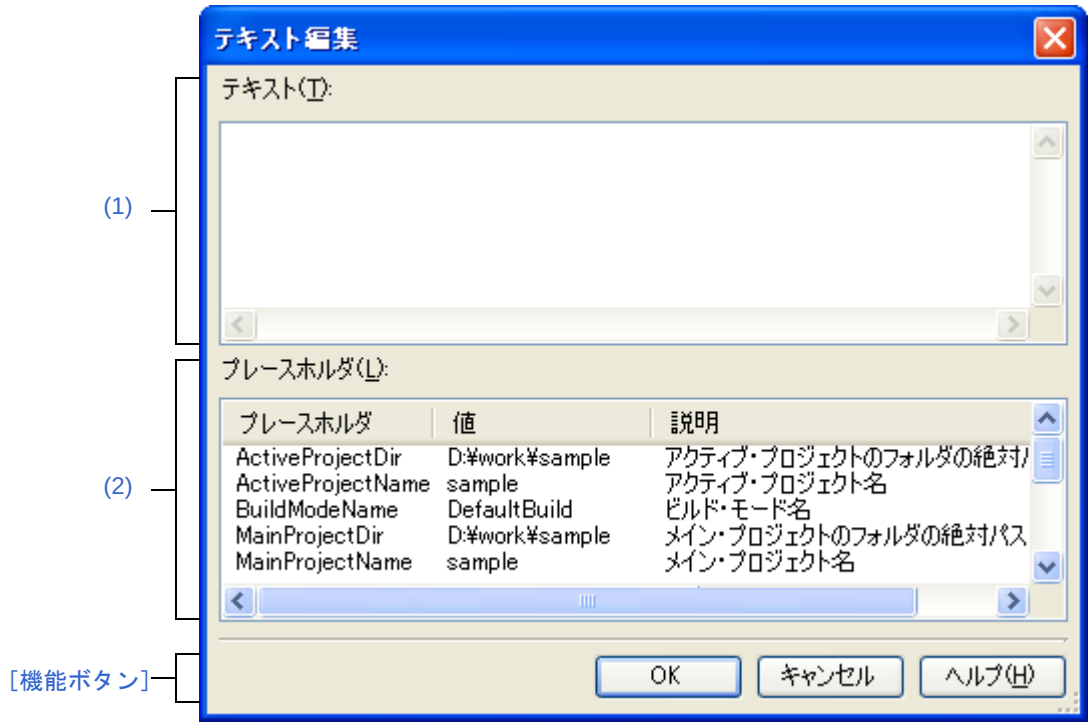
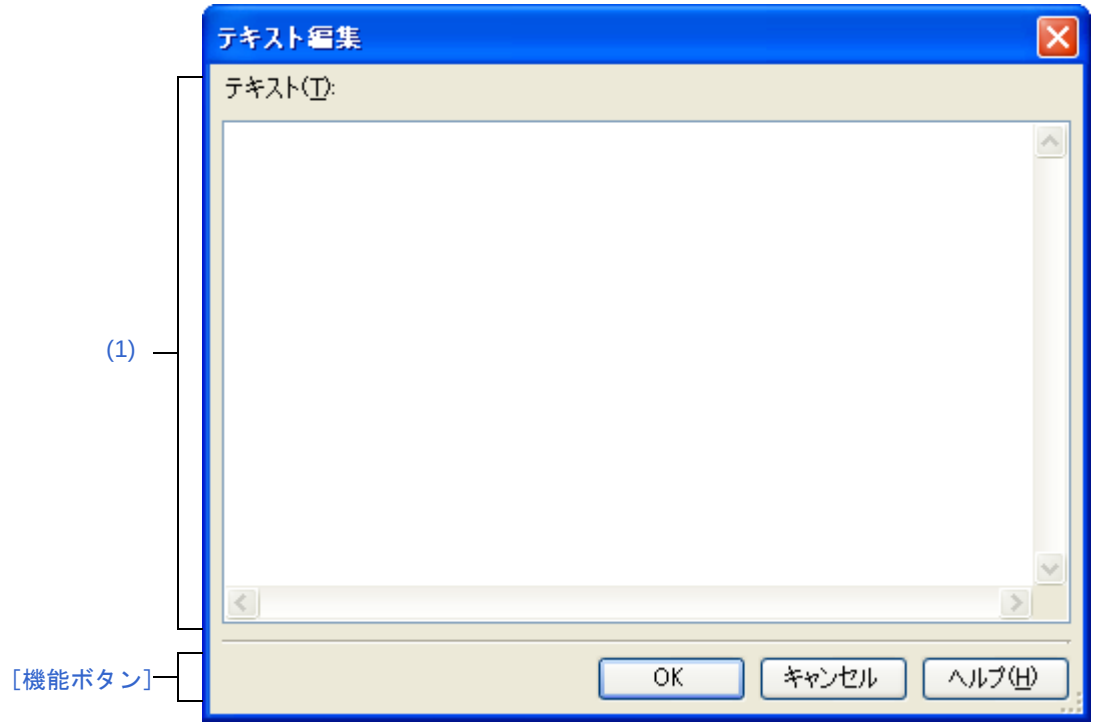


図 A.27 テキスト編集 ダイアログ（呼び出し元がプレースホルダに対応していない場合）



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［マクロ定義］，［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］，［よく使うオプション（アセンブル）］カテゴリの［マクロ定義］，［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［ソース］カテゴリの［コンパイル単位の先頭にインクルードするファイル］，［マクロ定義］，［MISRA C ルール検査］カテゴリの［ルール・チェック対象外のファイル］，［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［ソース］カテゴリの［マクロ定義］，［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［入力］カテゴリの［オブジェクト・モジュール・ファイル］，［使用するライブラリ・ファイル］，［バイナリ・データ・ファイル］，［シンボル定義］，［出力］カテゴリの［ROM から RAM へマップするセクション］，［分割出力ファイル］，［特定ベクタ番号のアドレス］，［外部定義シンボルへ分岐するジャンプ・テーブルを出力するセクション］，［ロード・モジュール・ファイル変換］カテゴリの［分割変換ファイル］，［計算範囲］，［最適化］カテゴリの［最適化による削除を抑止する未参照シンボル］，［最適化で統合を抑止する共通コード］，［最適化を抑止するセクション］，［最適化を抑止するアドレス範囲］，［セクション］カテゴリの［外部定義シンボルをファイル出力するセクション］，［セクション・アライメント］，［ベリファイ］カテゴリの［メモリ種別のアドレス範囲］，［分割対象外セクション］，［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］
- ［ライブラリアン・オプション］タブの［入力］カテゴリの［オブジェクト・モジュール・ファイル］，［使用するライブラリ・ファイル］，［バイナリ・データ・ファイル］，［その他］カテゴリの［ライブラリアン前に実行するコマンド］，［ライブラリアン後に実行するコマンド］
- ［ライブラリ・ジェネレート・オプション］タブの［その他］カテゴリの［ライブラリ・ジェネレート前に実行するコマンド］，［ライブラリ・ジェネレート後に実行するコマンド］
- ［個別コンパイル・オプション (C)］タブの［ソース］カテゴリの［コンパイル単位の先頭にインクルードするファイル］，［マクロ定義］，［MISRA C ルール検査］カテゴリの［ルール・チェック対象外のファイル］，［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］
- ［個別コンパイル・オプション (C++)］タブの［ソース］カテゴリの［コンパイル単位の先頭にインクルードするファイル］，［マクロ定義］，［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［ソース］カテゴリの［マクロ定義］，［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が反映されます。

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，以下のメッセージがツールチップ表示されます。

メッセージ	説明
*呼び出し元で指定されている最大文字数*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。	入力された文字列の文字数が，呼び出し元で指定され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）

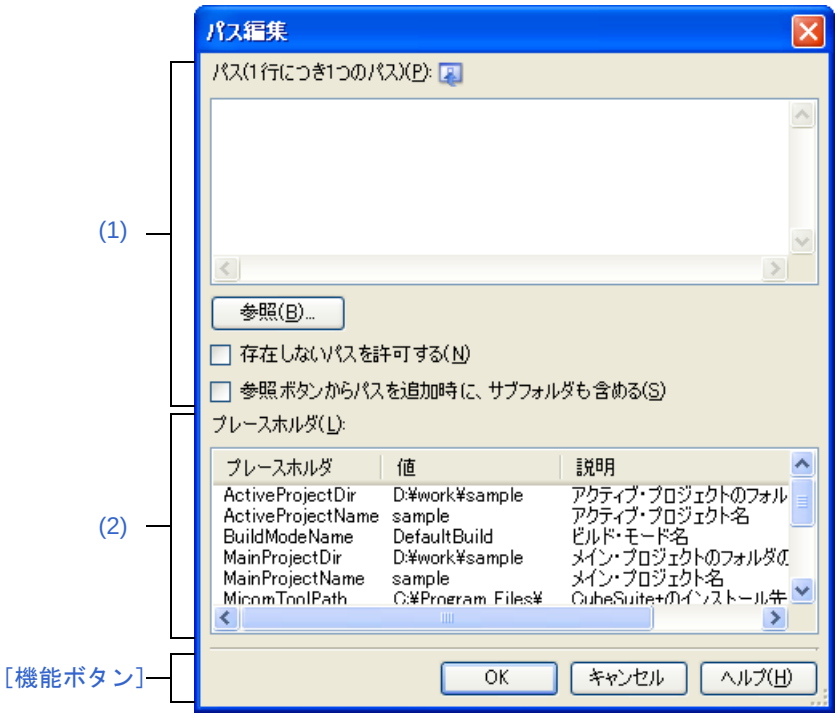
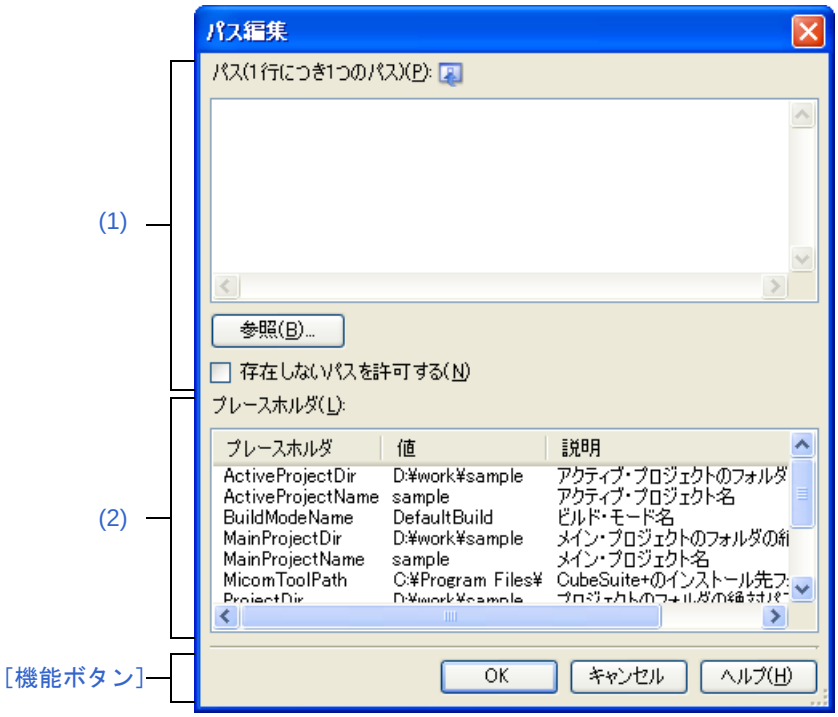


図 A.29 パス編集 ダイアログ（パスを含むファイル名を編集する場合）



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［使用するライブラリ・ファイル］
- ［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］，［MISRA-C:2004 ルール検査］カテゴリの［ルール・チェック対象外のファイル］
- ［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［使用するライブラリ・ファイル］
- ［ライブラリ・ジェネレート・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］
- ［個別コンパイル・オプション (C)］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］，［MISRA-C:2004 ルール検査］カテゴリの［ルール・チェック対象外のファイル］
- ［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，対象外ファイルの追加 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。	空白文字だけの行になっています。
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。

メッセージ	説明
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。

(c) ［存在しないパスを許可する］
このチェック・ボックスをチェックすると，［パス (1 行につき 1 つのパス)］に指定したパスの存在確認や文字列のチェックを行いません。

(d) ［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）。

注意 本項目は，パスを編集する場合のみ表示されます。

(2) ［プレースホルダ］エリア
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。

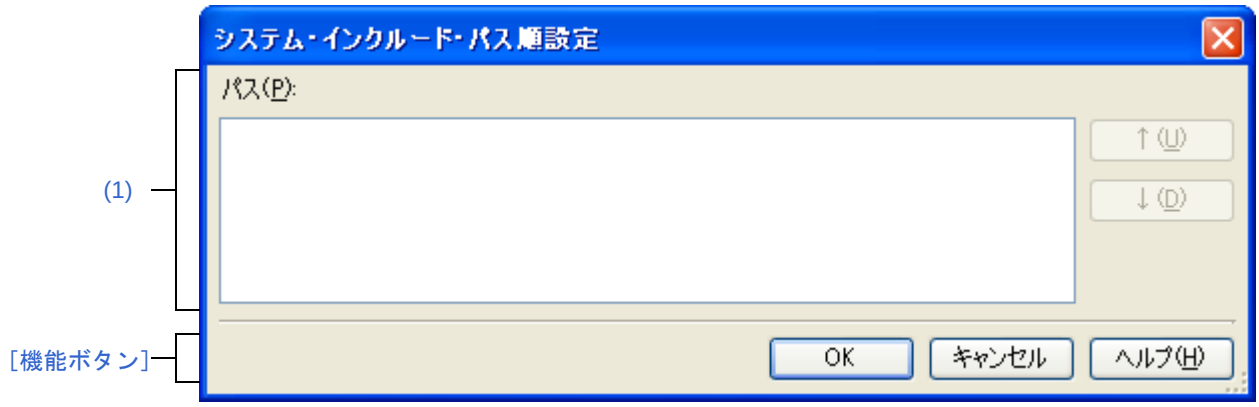
［機能ボタン］

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［ソース］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［ソース］カテゴリの［システム・インクルード・パス］
- [リンク・オプション] タブの［入力］カテゴリの［システム・ライブラリ・ファイル］

[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され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。

備考 2. 新規に追加されたシステム・インクルード・パスは，一覧の最後のパスの次に追加されます。

備考 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］

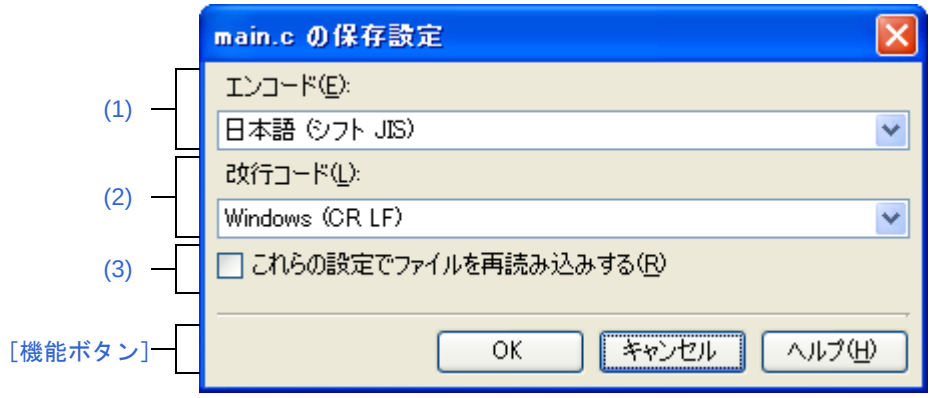
ボタン	機能
OK	コンパイラへのパスの指定順をパス一覧表示エリアの表示順に設定し，本ダイアログをクローズします。
キャンセル	パスの指定順の設定をキャンセルし，本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。

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が表示されます。

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，以下の順番で表示されます。
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されません。

- *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・ファイルの参照，およびリンク順の設定を行います。

図 A.32 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されているソース・ファイルから生成されるオブジェクト・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

デフォルトでは，プロジェクトに追加され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。

備考 2. 新規に追加したソース・ファイルから生成されるオブジェクト・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されます。

備考 3. ファイル名をドラッグ・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，「2.12.2 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.33 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・モードが選択されます。一致していない場合は，“DefaultBuild” が選択されます。
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，“*” が付加されます。
なお，ビルド・モードには，あらかじめ “DefaultBuild” が用意されており，常に先頭に表示されます。

(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，“*” が付加されているビルド・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，以下のメッセージがツールチップ表示されます。

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・モードが指定されました。
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・モード名が指定されました。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。

備考 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，以下のメッセージがツールチップ表示されます。

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され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.34 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。

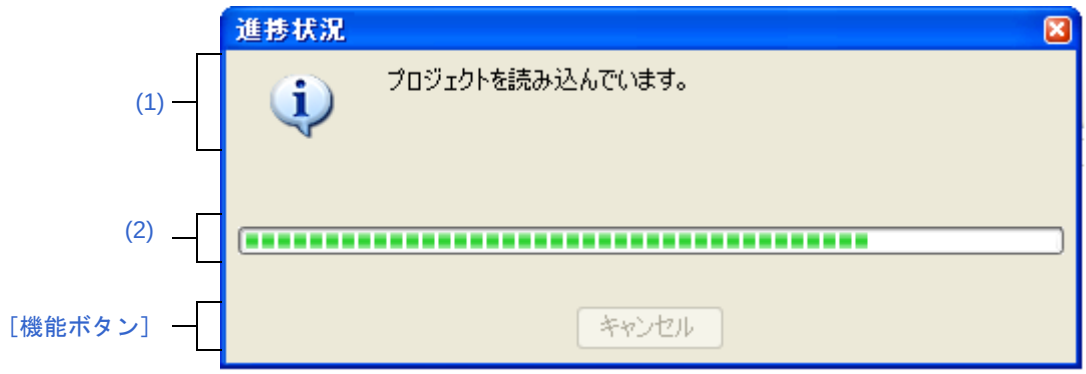
［機能ボタン］

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 パネルに表示されます。 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 パネルに表示されます。 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 パネルに表示されます。 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.35 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。

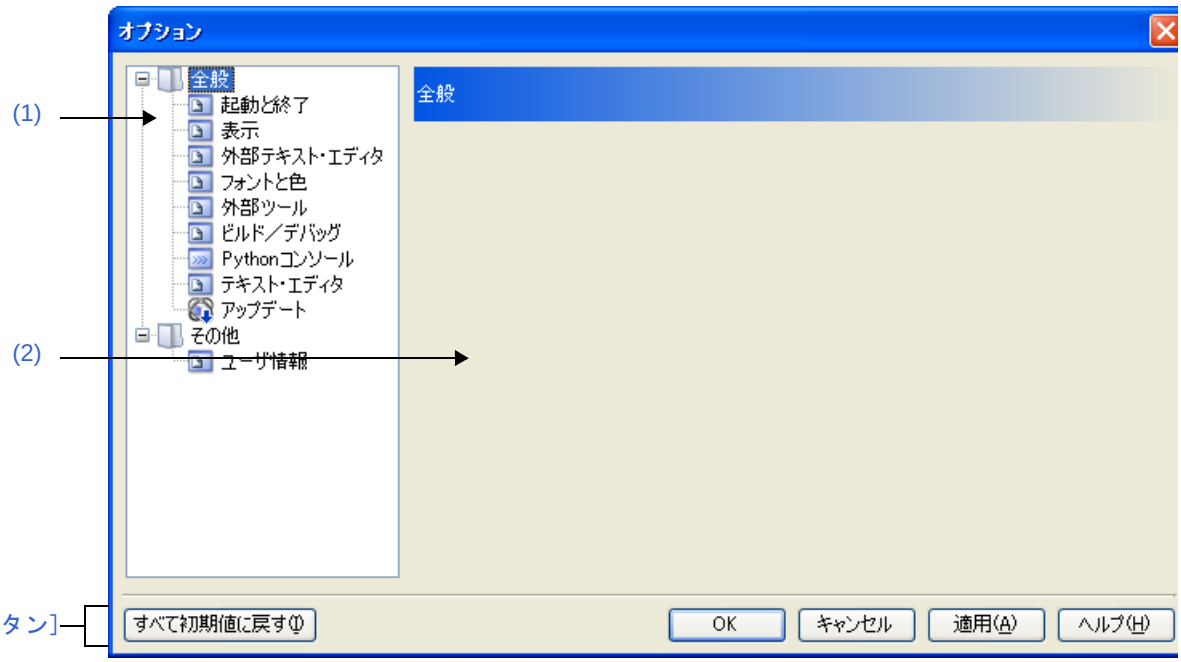
[機能ボタン]

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.36 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。

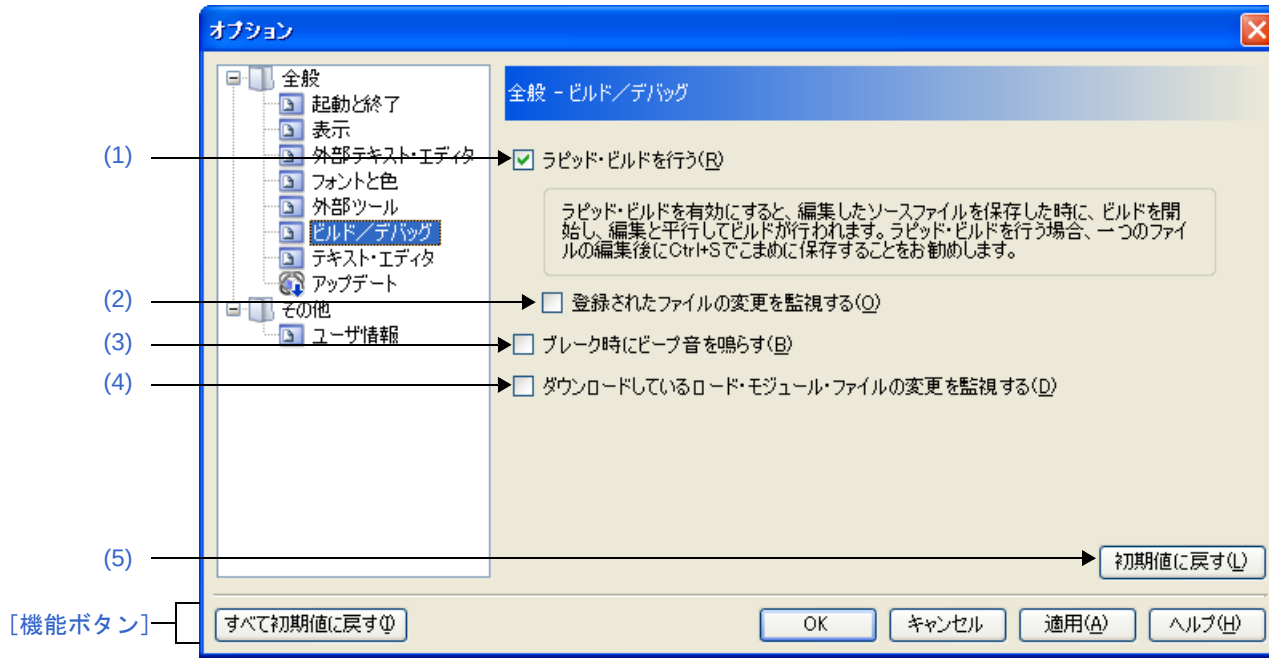
[機能ボタン]

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.37 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されたソース・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／保存されたときに，ラピッド・ビルドを開始します。
☐	プロジェクトに登録されたソース・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／保存されたときに，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。

注意 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) ［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）。

(4) ［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）。

(5) 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。

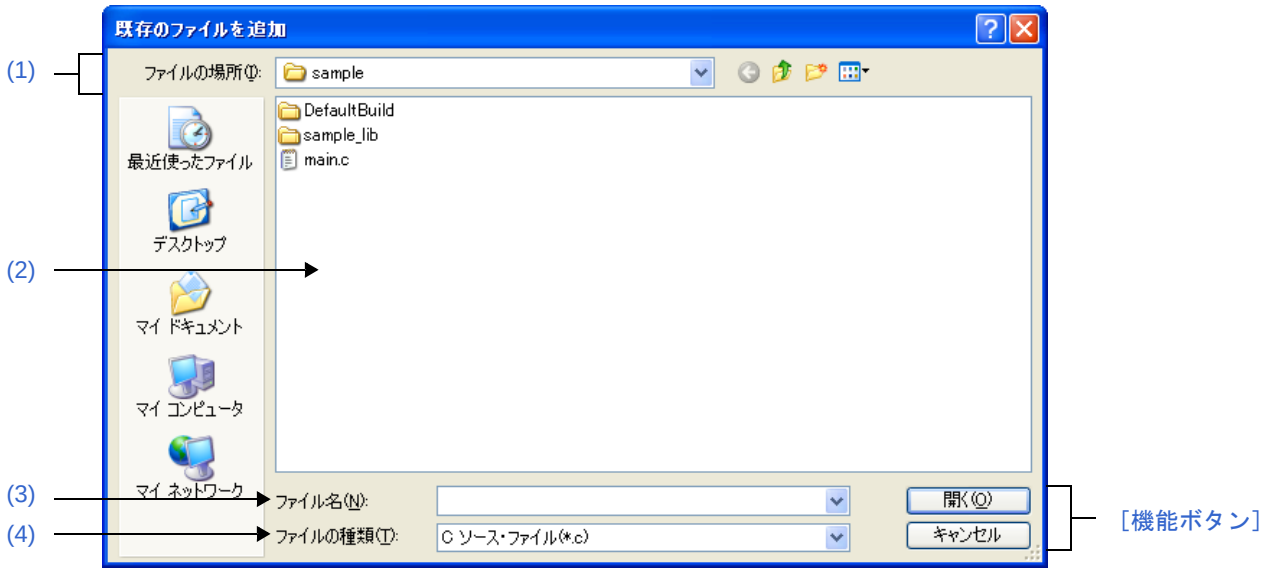
［機能ボタン］

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.38 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・フォルダが選択されます。

(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］で選択され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
C++ ソース・ファイル (*.cpp; *.cp; *.cc)	C++ ソース・ファイル
ヘッダ・ファイル (*.h; *.hpp; *.inc)	ヘッダ・ファイル
アセンブラ・ソース・ファイル (*.src; *.s)	アセンブラ・ソース・ファイル
ジャンプ・テーブル・ファイル (*.jmp)	ジャンプ・テーブル・ファイル
シンボル・アドレス・ファイル (*.fsy)	シンボル・アドレス・ファイル
ライブラリ・ファイル (*.lib)	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)	オブジェクト・モジュール・ファイル
リロケータブル・ファイル (*.lib)	リロケータブル・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）

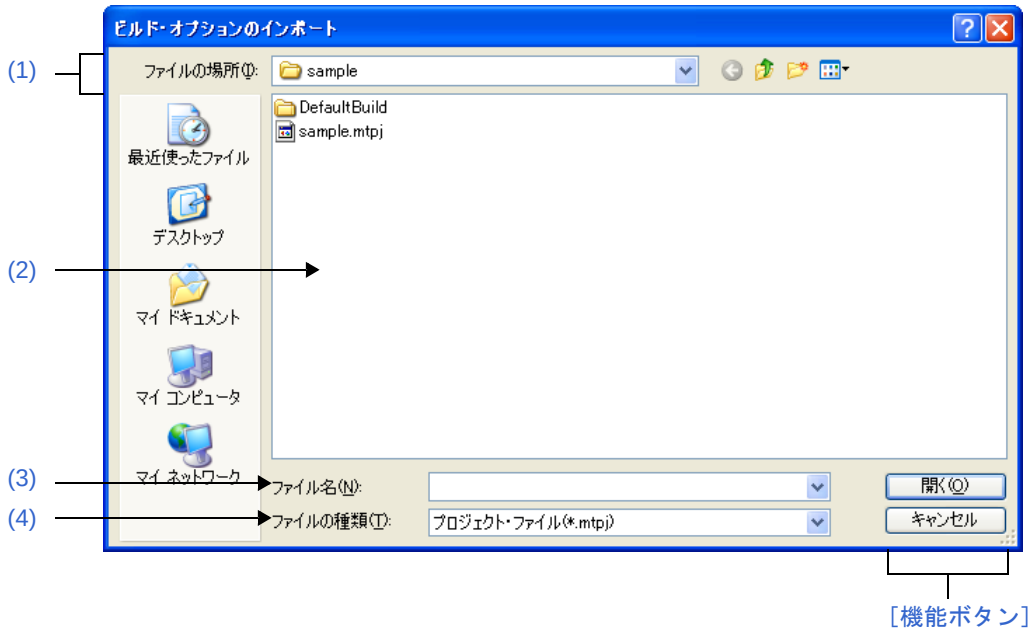
［機能ボタン］

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.39 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

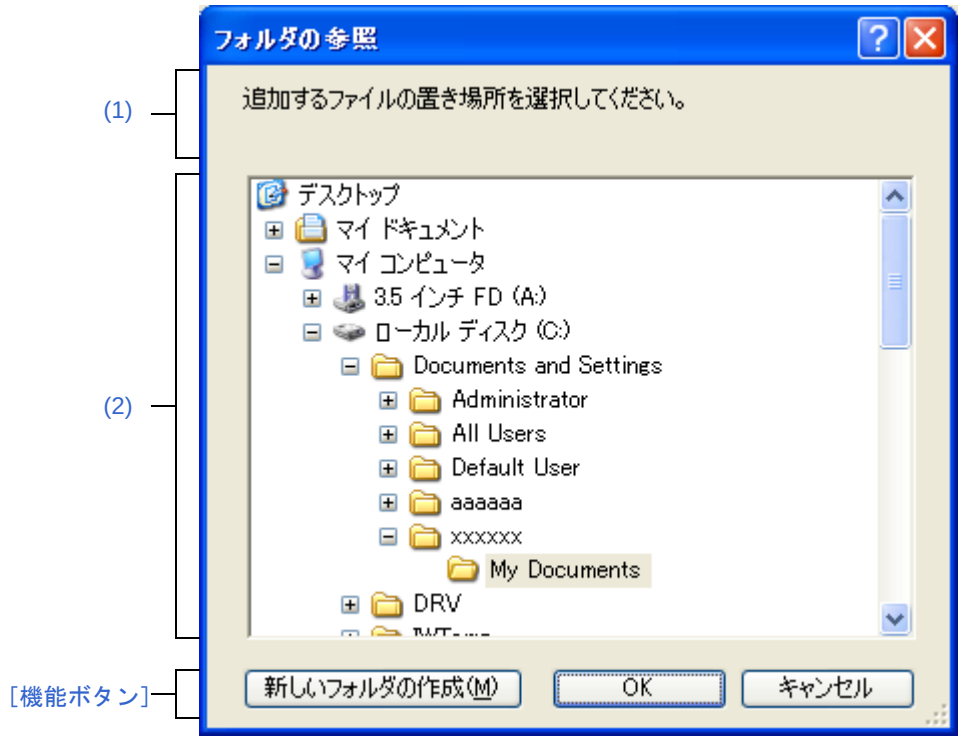
［機能ボタン］

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.40 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
- [コンパイル・オプション] タブの [オブジェクト] カテゴリの [出力フォルダ]
- [アセンブル・オプション] タブの [オブジェクト] カテゴリの [出力フォルダ]
- [リンク・オプション] タブの [オブジェクト] カテゴリの [出力フォルダ]，[ロード・モジュール・ファイル変換] カテゴリの [変換ファイル出力フォルダ]
- [ライブラリアン・オプション] タブの [出力] カテゴリの [出力フォルダ]
- [ライブラリ・ジェネレート・オプション] タブの [オブジェクト] カテゴリの [出力フォルダ]
- [個別コンパイル・オプション (C)] タブの [オブジェクト] カテゴリの [出力フォルダ]
- [個別コンパイル・オプション (C++)] タブの [オブジェクト] カテゴリの [出力フォルダ]
- [個別アセンブル・オプション] タブの [オブジェクト] カテゴリの [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。
デフォルトでは，呼び出し元に設定しているフォルダが選択されます。

備考 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，“C:¥Documents and Settings¥*ユーザ名*¥My Documents” が選択されます。

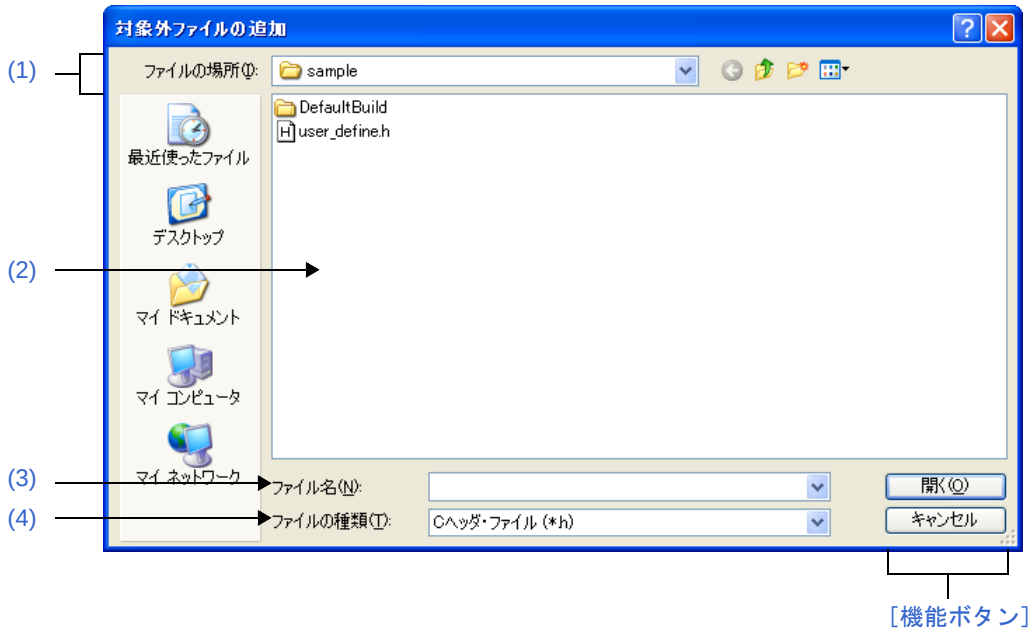
[機能ボタン]

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。

対象外ファイルの追加 ダイアログ

本ダイアログの呼び出し元に設定する MISRA-C:2004 ルールのチェック対象外のファイルの選択を行います。

図 A.41 対象外ファイルの追加 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]

- 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，[...] ボタンをクリックしてパス編集 ダイアログをオープン
- [コンパイル・オプション] タブの［MISRA-C:2004 ルール検査］カテゴリの［ルール・チェック対象外のファイル］
- [個別コンパイル・オプション (C)] タブの［MISRA-C:2004 ルール検査］カテゴリの［ルール・チェック対象外のファイル］
  →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・フォルダが選択されます。

(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］で選択され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。

C ヘッダ・ファイル (*.h)	C ヘッダ・ファイル（デフォルト）
C ソース・ファイル (*.c)	C ソース・ファイル
すべてのファイル (*.*)	すべての形式

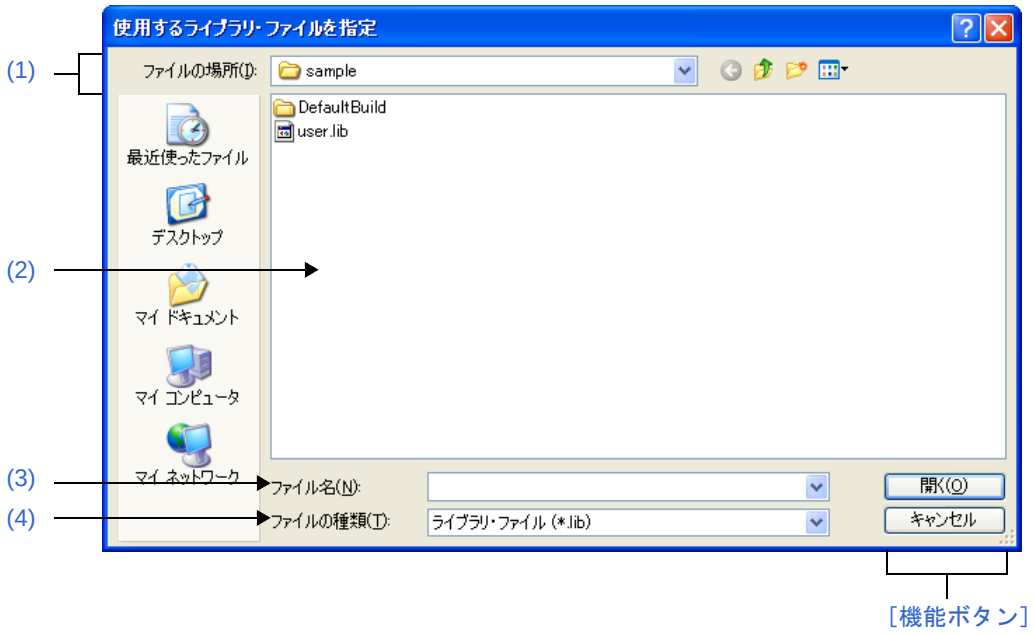
［機能ボタン］

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。

使用するライブラリ・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ライブラリ・ファイルの選択を行います。

図 A.42 使用するライブラリ・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，[...] ボタンをクリックしてパス編集 ダイアログをオープン
- [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］
- [ライブラリアン・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］
  →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・フォルダが選択されます。

(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］で選択され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。

ライブラリ・ファイル (*.lib)	ライブラリ・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。

セクション設定 ダイアログ

セクションの追加，変更，削除を行います。

図 A.43 セクション設定 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（リンク）］カテゴリの［セクションの開始アドレス］
- [リンク・オプション] タブの［セクション］カテゴリの［セクションの開始アドレス］

[各エリアの説明]

(1) アドレス - セクション エリア
現在設定しているセクション配置の一覧を表示します。

(a) ［アドレス］
セクションの開始アドレスを表示します。

(b) ［セクション］
セクション名を表示します。

(c) ［オーバーレイ *n*］
オーバーレイ配置されたセクション名を表示します。*n* は 1 で始まる数字です。

(d) ボタン

追加 ...	- セクション・グループ（アドレス）の追加 アドレス - セクションエリアでアドレスを選択している場合は，セクションのアドレス ダイアログをオープンします。 アドレス - セクションエリアの適切な位置に行（アドレス）を追加します（セクション名は空欄）。 - セクションの追加 アドレス - セクションエリアでセクションを選択している場合は，セクション追加 ダイアログをオープンします。 選択したセクションの属するセクション・グループの該当列に空欄がない場合は，そのセクション・グループの最下部に新しい行（セクション）を追加します。 選択したセクションの列の直下に空欄がある場合は，そこにセクションを設定します。
変更 ...	- セクション・グループ（アドレス）の変更 アドレス - セクションエリアでアドレスを選択している場合は，セクションのアドレス ダイアログをオープンします。 アドレス - セクションエリアでそのセクション・グループ (アドレスと属するセクション名) の位置を移動します。 - セクション名の変更 アドレス - セクションエリアでセクションを選択している場合は，セクション編集 ダイアログをオープンします。 選択したセクション名をセクション編集 ダイアログで入力したセクション名に置き換えます。
複数割り付け ...	- オーバーレイ配置セクションの追加 オーバーレイ配置セクション追加 ダイアログをオープンします。 アドレス - セクションエリアに [オーバーレイ *n*] 列（*n* は 1 で始まる数字）を追加します。選択しているセクション・グループ（アドレス）に属する [オーバーレイ *n*] 列にセクションを設定します。
削除	- セクション・グループ（アドレス）の削除 アドレス - セクションエリアでアドレスを選択している場合は，削除するセクション ダイアログをオープンします。 削除するセクション ダイアログで選択したセクションをアドレス - セクションエリアから削除します。セクション・グループに属するすべてのセクション名を選択した場合は，そのセクション・グループを削除します。 - セクションの削除 アドレス - セクションエリアで選択しているセクションを削除します。 セクション・グループに属するセクションがない場合は、そのセクション・グループを削除します。 - アドレス - セクションエリアの［オーバーレイ *n*］に属するセクション名がない場合は，その列を削除します。
↑	- 選択しているセクションを上へ移動します。
↓	- 選択しているセクションを下へ移動します。
インポート ...	- セクション設定を読み込むために，ファイルを開くダイアログをオープンします。ファイルを開くダイアログで［開く］ボタンをクリックすると，指定したファイルを読み込みアドレス - セクションエリアに表示します。
エクスポート ...	- セクション設定を保存するために，名前を付けて保存 ダイアログをオープンします。 名前を付けて保存 ダイアログで［保存］ボタンをクリックすると，指定したファイルにアドレス - セクションエリアの内容を書き込みます。

［機能ボタン］

ボタン	機能
OK	設定したセクション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	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。

セクション追加，セクション編集，オーバーレイ配置セクションの追加 ダイアログ

セクション追加，編集，複数割り付け時のセクション名の設定を行います。

図 A.44 セクション追加 ダイアログ

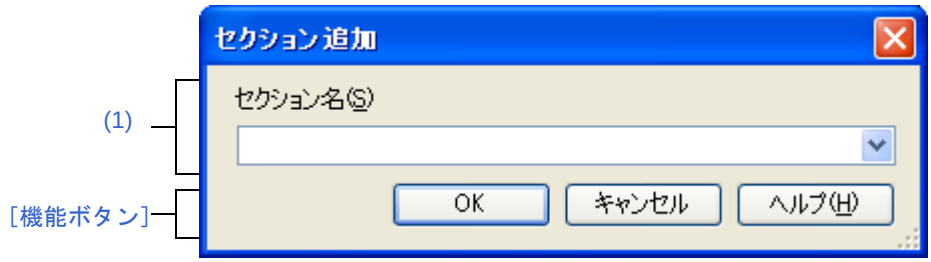


図 A.45 セクション編集 ダイアログ

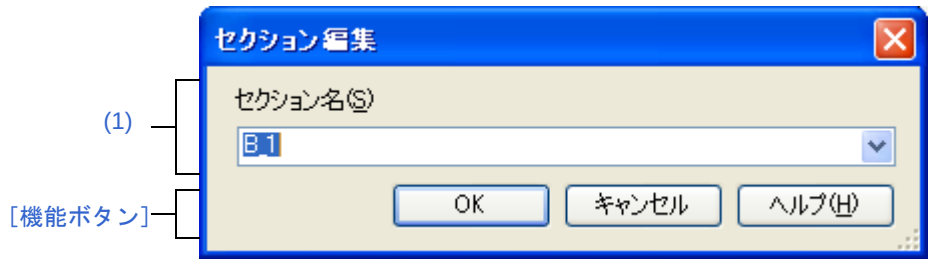
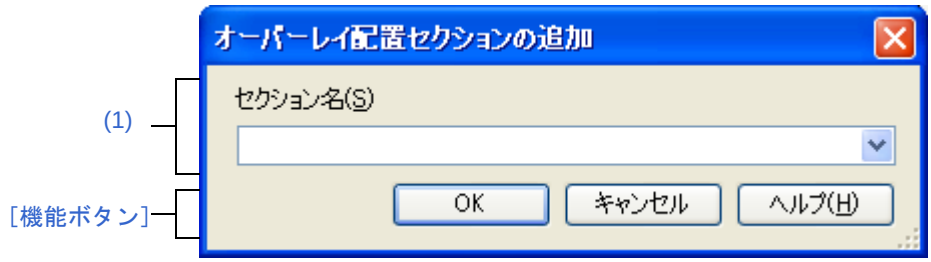


図 A.46 オーバーレイ配置セクションの追加 ダイアログ



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]

- セクション追加 ダイアログ
- セクション設定 ダイアログ において，アドレス - セクションエリア内のセクションを選択したのち，[追加] ボタンをクリック
- セクション編集 ダイアログ
- セクション設定 ダイアログ において，アドレス - セクションエリア内のセクションを選択したのち，[変更] ボタンをクリック
- オーバーレイ配置セクションの追加
- セクション設定 ダイアログ において，アドレス - セクションエリア内のアドレスを選択したのち，[複数割り付け] ボタンをクリック

[各エリアの説明]

(1) [セクション名]

セクション名の入力を行います。
テキスト・ボックスに直接入力するか，またはドロップダウン・リストよりセクション名を選択します。
使用可能な文字は，英大文字（A ～ Z），英小文字（a ～ z），数字（0 ～ 9），アンダスコア（_），ドル記号（$）です。セクション名の最初の文字には 0 から 9 は使用できません。
ドロップダウン・リストには，リンカにより以下の予約セクションが設定されています。
$ABS16B，$ABS16C，$ABS16D，$ABS8B，$ABS8C，$ABS8D，$INDIRECT，B，B$1，B$2，B$4，C，C$1，C$2，C$4，C$BSEC，C$DSEC，C$INIT，C$VTBL，D，D$1，D$2，D$4，L，PS

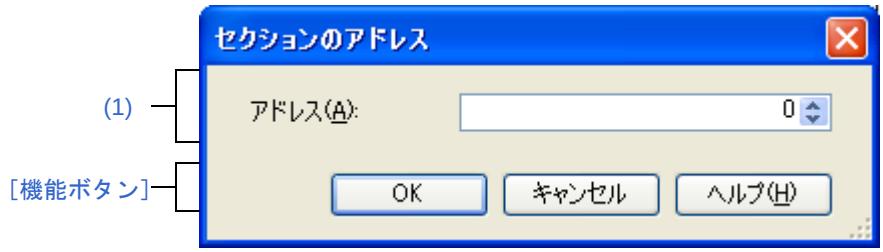
[機能ボタン]

ボタン	機能
OK	- セクション追加 ダイアログの場合 本ダイアログをクローズし，セクション設定 ダイアログのアドレス - セクションエリアのセクション・グループ（アドレス）にセクションを追加します。 選択したセクションの属するセクション・グループの該当列に空欄がない場合は，そのセクション・グループの最下部に新しい行（セクション）を追加します。 選択したセクションの列の直下に空欄がある場合は，そこにセクションを設定します。 - セクション編集 ダイアログの場合 本ダイアログをクローズし，セクション設定 ダイアログのアドレス - セクションエリアで選択しているセクション名をセクション編集 ダイアログで入力したセクション名に置き換えます。 - オーバーレイ配置セクションの追加 ダイアログの場合 本ダイアログをクローズし，セクション設定 ダイアログのアドレス - セクションエリアに [オーバーレイ *n*] 列（*n* は 1 で始まる数字）を追加します。選択しているセクション・グループ（アドレス）に属する [オーバーレイ *n*] 列にセクションを設定します。
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。

セクションのアドレス ダイアログ

セクションの追加，または変更時のアドレスの設定を行います。

図 A.47 セクションのアドレス 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］

- セクション設定 ダイアログ において，アドレス - セクションエリア内のアドレスを選択したのち，［追加］，［変更］ボタンをクリック

［各エリアの説明］

(1) ［アドレス］
セクションの開始アドレスの入力を行います。
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。

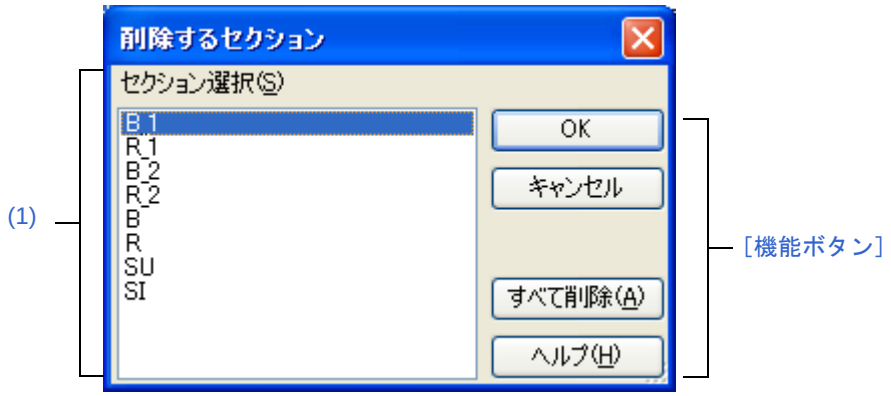
［機能ボタン］

ボタン	機能
OK	本ダイアログをクローズし，入力したアドレスをセクション設定 ダイアログのアドレス - セクションエリアに設定します。 - セクション・グループ（アドレス）の追加 アドレス - セクションエリアの適切な位置に行（アドレス）を追加します（セクション名は空欄）。 - セクション・グループ（アドレス）の変更 アドレス - セクションエリアでそのセクション・グループ (アドレスと属するセクション名) の位置を移動します。
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。

削除するセクション ダイアログ

セクションの削除を行います。

図 A.48 削除するセクション 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］

- セクション設定 ダイアログ において，アドレス - セクションエリア内のアドレスを選択したのち，［削除］ボタンをクリック

［各エリアの説明］

(1) ［セクション選択］エリア
削除するセクション名を選択します。
［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。

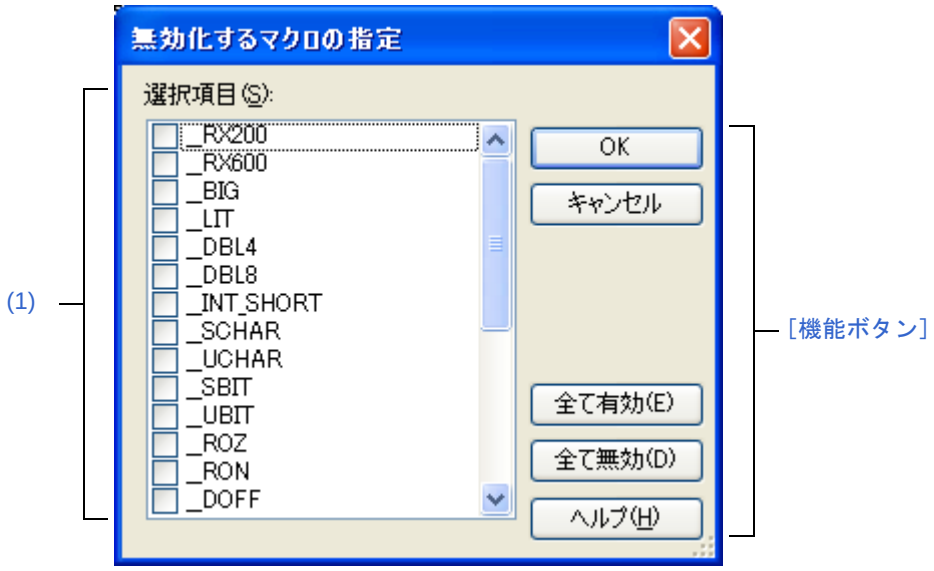
［機能ボタン］

ボタン	機能
OK	［セクション選択］エリアで選択しているセクションをセクション設定 ダイアログのアドレス - セクションエリアから削除し，本ダイアログをクローズします。 セクション・グループ（アドレス）に属するセクションがない場合は、そのセクション・グループを削除します。 アドレス - セクションエリアの［オーバーレイ *n*］列に属するセクション名がない場合は，その列を削除します。
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。
すべて削除	［セクション選択］エリアのすべてのセクションをセクション設定 ダイアログの［アドレス - セクション］エリアから削除し，本ダイアログをクローズします。 セクション・グループ（アドレス）を削除します。
ヘルプ	本ダイアログのヘルプを表示します。

無効化するマクロの指定 ダイアログ

本ダイアログの呼び出し元に設定する無効化するプリデファインド・マクロを選択します。

図 A.49 無効化するマクロの指定 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 パネルの以下のプロパティにおいて，以下のプロパティを選択したのち，[...] ボタンをクリック
- [コンパイル・オプション] タブの［ソース］カテゴリの［無効化するプリデファインド・マクロ］
- [個別コンパイル・オプション (C)] タブの［ソース］カテゴリの［無効化するプリデファインド・マクロ］
- [個別コンパイル・オプション (C++)] タブの［ソース］カテゴリの［無効化するプリデファインド・マクロ］

[各エリアの説明]

(1) ［選択項目］エリア
無効化が可能なプリデファインド・マクロの一覧を表示します。
本ダイアログの呼び出し元に設定する無効化するプリデファインド・マクロをチェック・ボックスにより選択します。

備考 このダイアログの呼び出し元で，すでに無効化するプリデファインド・マクロを設定していた場合は，該当するプリデファインド・マクロのチェック・ボックスはデフォルトでチェック状態となります。

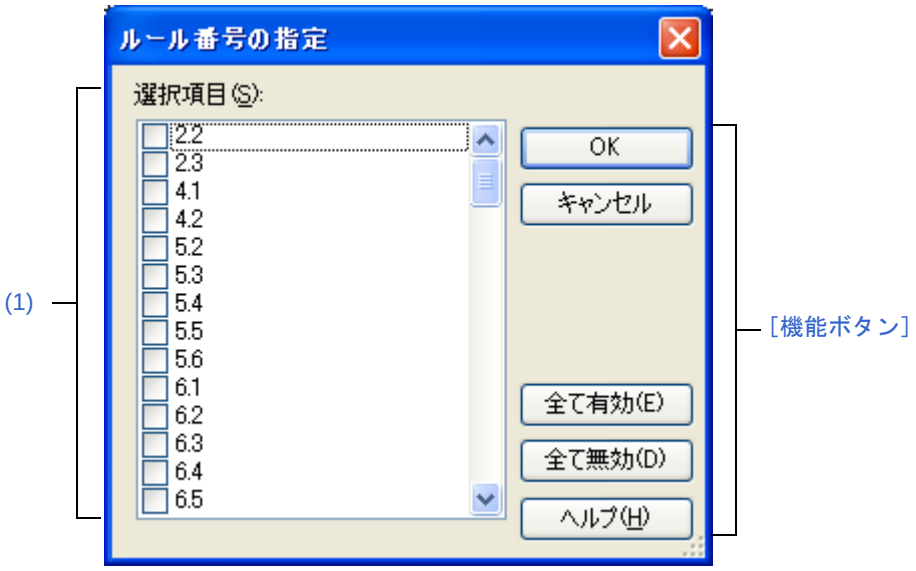
［機能ボタン］

ボタン	機能
OK	本ダイアログをクローズし，選択した無効化するプリデファインド・マクロを呼び出し元に設定します。
キャンセル	無効化するプリデファインド・マクロの選択をキャンセルし，本ダイアログをクローズします。
全て有効	すべてのチェック・ボックスを選択状態にします。
全て無効	すべてのチェック・ボックスを非選択状態にします。
ヘルプ	本ダイアログのヘルプを表示します。

ルール番号の指定 ダイアログ

本ダイアログの呼び出し元に設定する MISRA-C: 2004 ルール番号の選択を行います。

図 A.50 ルール番号の指定 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 パネルの以下のプロパティにおいて，以下のプロパティを選択したのち，[...] ボタンをクリック
- [コンパイル・オプション] タブ，[MISRA C ルール検査] カテゴリの [ルール番号]，[除外するルール番号]，[必須ルールの他にチェックするルール番号]，[必須ルールから除外するルール番号]
- [個別コンパイル・オプション (C)] タブ，[MISRA C ルール検査] カテゴリの [ルール番号]，[除外するルール番号]，[必須ルールの他にチェックするルール番号]，[必須ルールから除外するルール番号]

[各エリアの説明]

(1) [選択項目]

本ダイアログの呼び出し元に指定可能な MISRA-C: 2004 ルール番号の一覧を表示します（昇順）。
設定するルール番号をチェック・ボックスにより選択します。

備考 このダイアログの呼び出し元ですでにルール番号を設定していた場合は，該当するルール番号のチェック・ボックスはデフォルトでチェック状態となります。

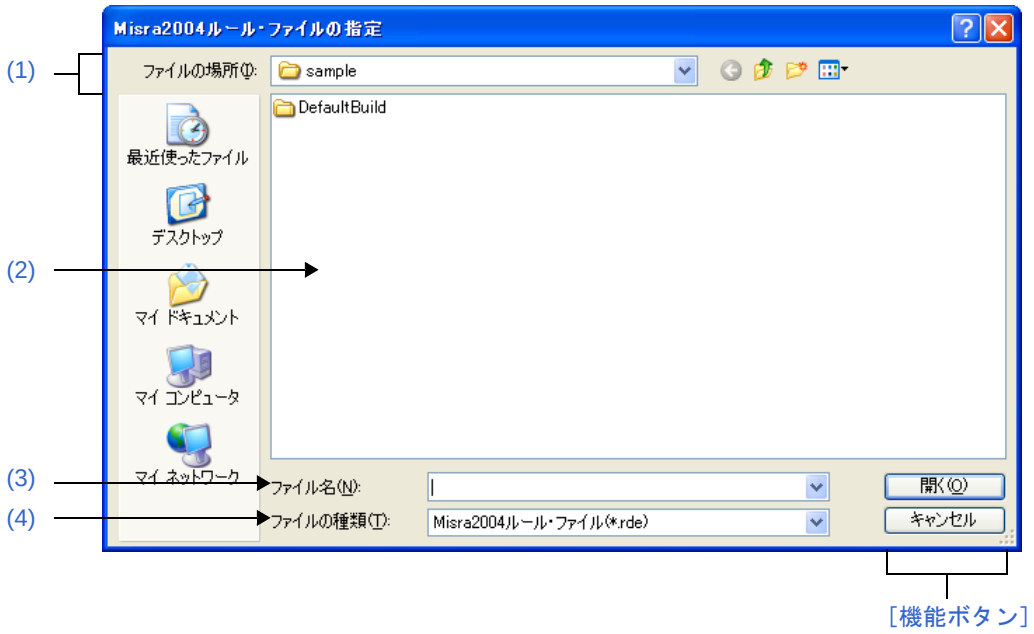
［機能ボタン］

ボタン	機能
OK	本ダイアログをクローズし，選択したルール番号を呼び出し元に設定します。
キャンセル	ルール番号の選択をキャンセルし，本ダイアログをクローズします。
全て有効	［選択項目］において，すべてのルール番号を選択します。
全て無効	［選択項目］において，すべてのルール番号の選択を解除します。
ヘルプ	本ダイアログのヘルプを表示します。

Misra2004 ルール・ファイルの指定 ダイアログ

本ダイアログの呼び出し元に設定する MISRA-C: 2004 ルール・ファイルの選択を行います。

図 A.51 Misra2004 ルール・ファイルの指定 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
- [コンパイル・オプション] タブの［MISRA C ルール検査］カテゴリの［ルール番号記載ファイル］
- [個別コンパイル・オプション (C)] タブの［MISRA C ルール検査］カテゴリの［ルール番号記載ファイル］

[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・フォルダが選択されます。

(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］で選択され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。

Misra2004 ルール・ファイル (*.rde)	Misra2004 ルール・ファイル（デフォルト）
すべてのファイル (*.*)	すべての形式

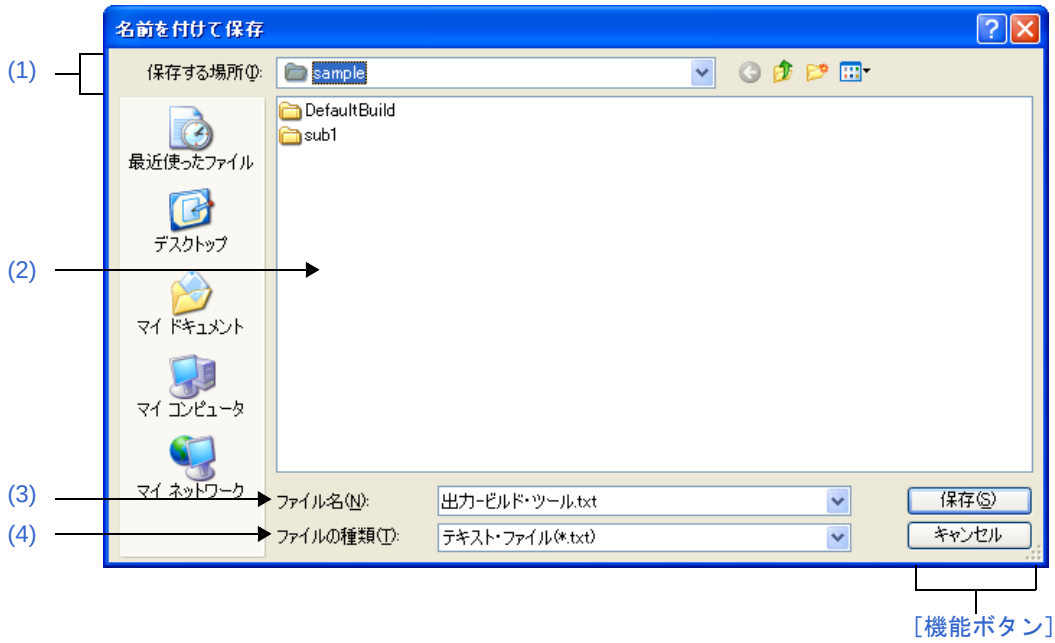
[機能ボタン]

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。

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名前を付けてファイルに保存します。

図 A.52 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，以下のフォルダが選択されます。

(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］エリアで選択され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）が表示されます。

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
C++ ソース・ファイル (*.cpp; *.cp; *.cc)	C++ ソース・ファイル
ヘッダ・ファイル (*.h; *.hpp; *.inc)	ヘッダ・ファイル
アセンブル・ファイル (*.src; *.s)	アセンブラ・ソース・ファイル
ジャンプ・テーブル・ファイル (*.jmp)	ジャンプ・テーブル・ファイル
シンボル・アドレス・ファイル (*.fsy)	シンボル・アドレス・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
マップ・ファイル (*.map; *.lbp)	マップ・ファイル
ヘキサ・ファイル (*.hex)	ヘキサ・ファイル
S レコード・ファイル (*.mot)	S レコード・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）が表示されます。

テキスト・ファイル (*.txt)	テキスト形式

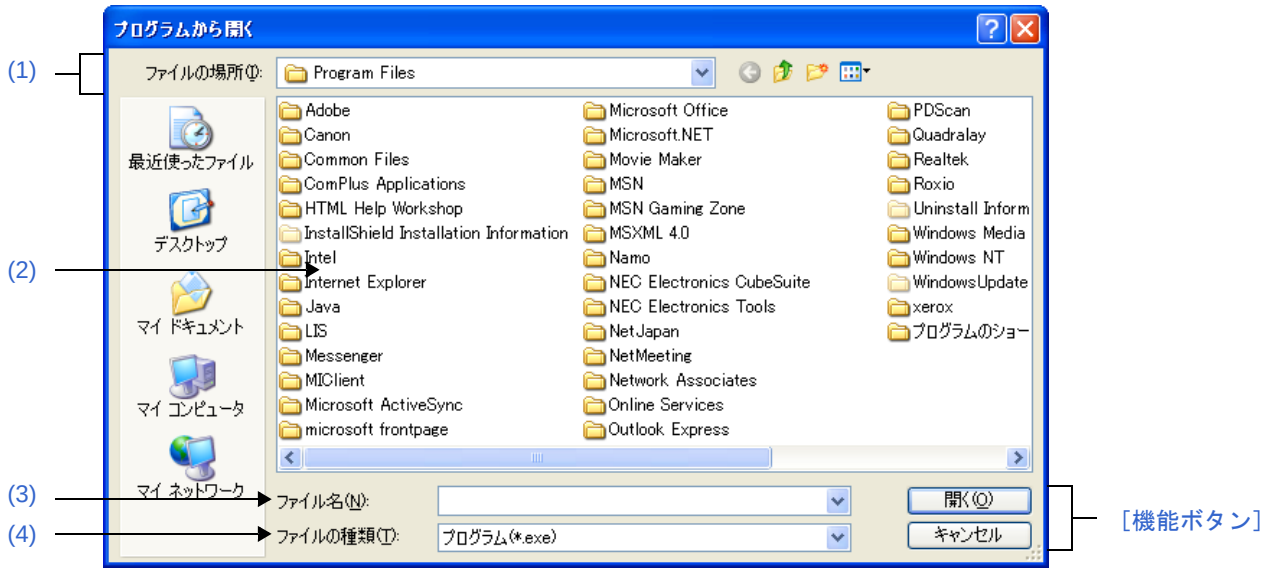
［機能ボタン］

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.53 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”）が選択されます。

(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］で選択され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

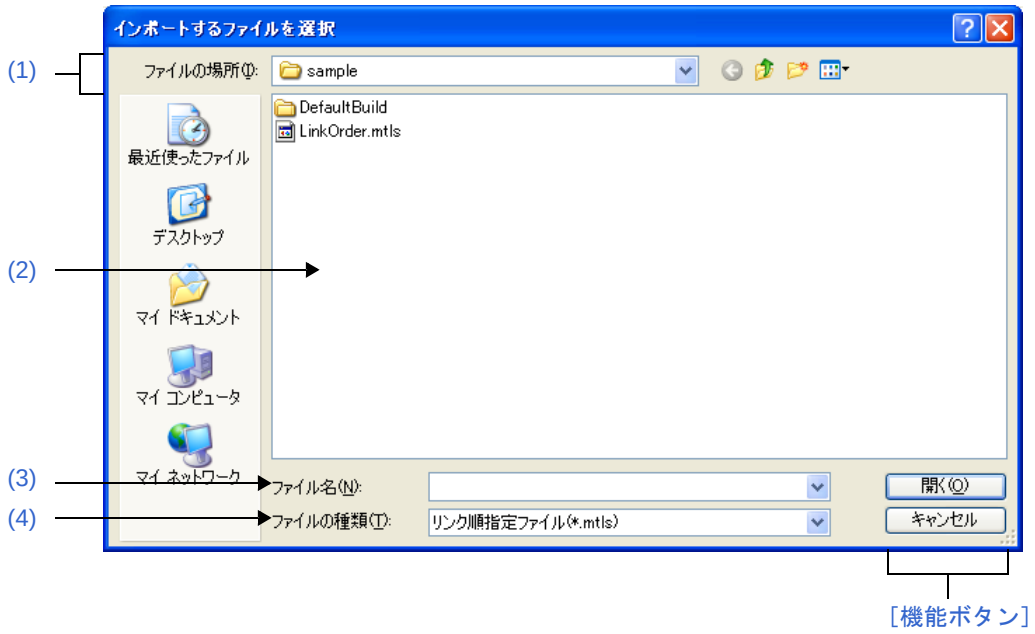
［機能ボタン］

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.54 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

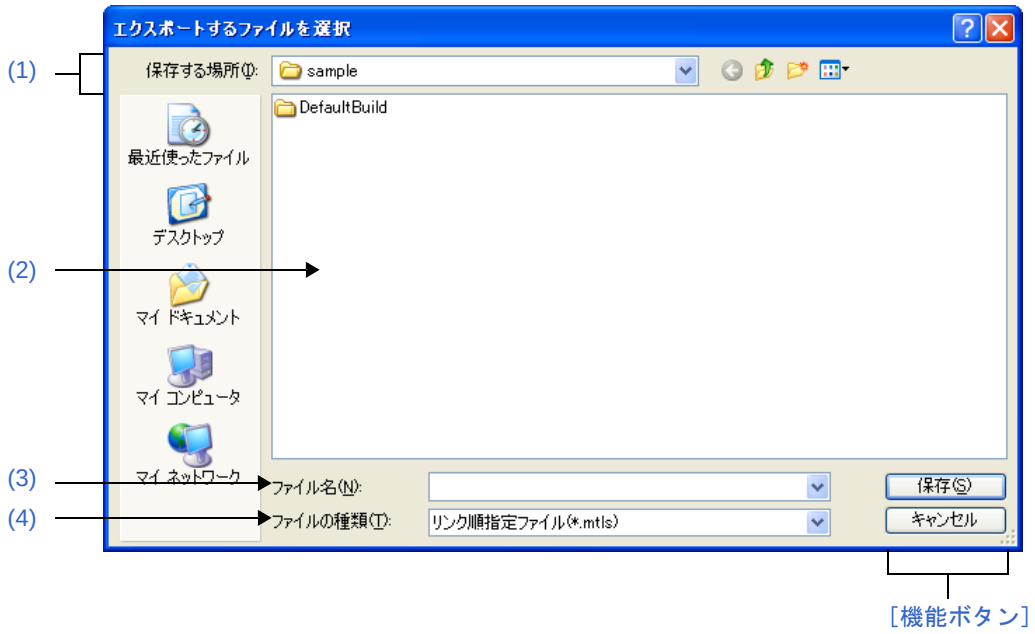
［機能ボタン］

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.55 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。

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仕様についての詳細を説明します。

B.1 RX ファミリ用 C/C++ コンパイラ

RX ファミリ用 C/C++ コンパイラは，C 言語，C99 言語，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。
RX ファミリ用 C/C++ コンパイラでは，1 つのドライバがプリプロセッサからリンクまでの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。

(2) プリプロセッサ
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。

(3) 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。

(4) 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。

(5) 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。

(6) 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。

(7) 最適化リンケージエディタ
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。

B.1.1 入出力ファイル

RX ファミリ用 C/C++ コンパイラの入出力ファイルを以下に示します。

表 B.1 RX ファミリ用 C/C++ コンパイラ用入出力ファイル

ファイル種別	拡張子	入出力	説明
C ソースプログラムファイル	.c	入力	C 言語，C99 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。
C++ ソースプログラムファイル	.cpp .cc .cp	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。
インクルードファイル	任意	入力	ソース・ファイルで参照するファイル C 言語，C99 言語，C++ 言語，もしく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。
C プログラム用プリプロセッサ展開ファイル	.p	出力	入力 C 言語または C99 言語ソース・プログラム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 -output=prep オプション指定時に出力します。
C++ プログラム用プリプロセッサ展開ファイル	.pp	出力	入力 C++ 言語ソース・プログラム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 -output=prep オプション指定時に出力します。

ファイル種別	拡張子	入出力	説明
アセンブリソースプログラムファイル	.src	出力	コンパイルにより，C,C99 または C++ ソースから生成したアセンブリ言語ファイル
	.src	入力	アセンブリ言語で記述したソース・ファイル
アセンブリプログラム用リストファイル	.lst	出力	アセンブル結果の情報を持つリスト・ファイル -listfile オプション指定時に出力します。 -show オプションで出力内容を選択します。
リロケータブルオブジェクトプログラムファイル	.obj	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
		入力	
アブソリュートロードモジュールファイル	.abs	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。
リンケージリストファイル	.map	出力	リンク結果の情報を持つリスト・ファイル -list オプション指定時に出力します。 -show オプションで出力内容を選択します。
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
		入力	
ライブラリリストファイル	.lbp	出力	ライブラリ作成結果の情報を持つリスト・ファイル -list オプション指定時に出力します。 -show オプションで出力内容を選択します。
ライブラリバックアップファイル	.lbk	出力	ライブラリジェネレータが既に存在するライブラリファイルに上書きする場合に、上書き前の内容を保存しておくファイルです。
ヘキサ・ファイル（モトローラ S フォーマット）	.mot	出力	ロード・モジュール・ファイルをモトローラ S フォーマットに変換したファイル
ヘキサ・ファイル（インテル（拡張）HEX フォーマット）	.hex	出力	ロード・モジュール・ファイルをインテル（拡張）HEX フォーマットに変換したファイル
ヘキサ・ファイル（バイナリフォーマット）	.bin	出力	ロード・モジュール・ファイルをバイナリフォーマットに変換したファイル
スタック情報ファイル	.sni	出力	スタック情報ファイル -stack オプション指定時に出力します。
デバッグ情報ファイル	.dbg	出力	デバッグ情報ファイル -sdebug オプション指定時に出力します。
拡張子 td のファイルで指定された定義を含むオブジェクトファイル	.rti	出力	拡張子 td のファイルで指定された定義を含むオブジェクトファイル
呼び出し情報ファイル	.cal	出力	呼び出し情報ファイル CallWalker で出力します。
外部シンボル割り付け情報ファイル	.bls	出力	外部シンボル割り付け情報ファイル リンク時 -map オプション指定時に出力します。
	.bls	入力	外部シンボル割り付け情報ファイル コンパイル時 -map オプションの入力ファイルとして指定します。
ジャンプテーブルファイル（アセンブリ言語）	.jmp	出力	外部定義シンボルへ分岐するジャンプテーブルのアセンブラソースファイル -jump_entries_for_pic オプション指定時に出力します。

ファイル種別	拡張子	入出力	説明
シンボルアドレスファイル（アセンブリ言語）	.fsy	出力	外部定義シンボルをアセンブラ制御命令で記述したアセンブラソースファイル -fsymbol オプション指定時に出力します。
C++ 言語機能サポートファイル	.td,.ti,.pi,.ii	出力	C++ 言語機能をサポートするための情報ファイルです。

B.1.2 操作方法

ここでは，RX ファミリ用 C/C++ コンパイラの操作方法について説明します。

なお、オプションはコマンドラインの左から右に順に取り込みます。異なる意味を持つオプションの指定を同時に指定した場合、エラーや警告がない場合は、後 (右側) に書いた方が有効です。エラーや警告が出る条件や、その結果はオプションにより異なりますので、詳しくは各オプションの説明でご確認ください。

(1) 各ツールの操作方法

(a) コンパイルドライバ（**ccrx**）

ccrx はコンパイルドライバの起動コマンドです。
本コマンド起動により，コンパイル，アセンブル，リンクを行うことができます。
- 入力ファイルの拡張子が「.s」「.src」「.S」「.SRC」のいずれかである場合，コンパイルドライバはそのファイルをアセンブリ言語ファイルと解釈して，アセンブラを起動します。
- 入力ファイルの拡張子が「.c」「.C」のいずれかである場合，コンパイルドライバはそのファイルを C 言語ソースファイルと解釈して，コンパイラを起動します。
- 入力ファイルの拡張子が「.cpp」「.CPP」「.cc」「.CC」「.cp」「.CP」のいずれかである場合，コンパイルドライバはそのファイルを C++ 言語ソースファイルと解釈して，コンパイラを起動します。

これら以外の拡張子のファイルは，C 言語ソースファイルとしてコンパイルします。

入力ファイルは複数同時に指定することができます。なお、C/C++ 言語ソースファイルを入力ファイルに複数同時に指定する場合を「一括コンパイル」と呼びます。

【コマンド記述形式】

```
ccrx [ △ < オプション > … ][ △ < ファイル名 >[ △ < オプション > …] …]
        < オプション > : -< オプション >[=< サブオプション >[=< サブオプション >]][,…]
```

(b) アセンブラ（**asrx**）

asrx は，アセンブラの起動コマンドです。

【コマンド記述形式】

```
asrx [ △ < オプション > … ][ △ < ファイル名 >[ △ < オプション > …] …]
        < オプション > : -< オプション >[=< サブオプション >][,…]
```

(c) 最適化リンケージエディタ（**rlink**）

rlink は，最適化リンケージエディタの起動コマンドです。
リンク処理だけではなく，以下に挙げる機能も含んでいます。

- リロケータブルファイル結合時の最適化
- ライブラリファイルの作成や編集
- モトローラ S 形式ファイル，インテル HEX 形式ファイル，およびバイナリファイルへのコンバート

【コマンド記述形式】

```
rlink [ △ < オプション > … ][ △ < ファイル名 >[ △ < オプション > …] …]
          < オプション > : -< オプション >[=< サブオプション >][,…]
```

(d) ライブラリジェネレータ（**lbgrx**）

ライブラリ・ジェネレータは、標準ライブラリを生成するツールです。標準ライブラリを生成する際、任意のコンパイル・オプションを指定することができます。
lbgrx は，ライブラリジェネレータの起動コマンドです。

【コマンド記述形式】

```
lbgrx [ △<オプション> … ]
          <オプション>：-<オプション>[=<サブオプション>][,…]
```

(2) 操作方法例

(a) コンパイル，アセンブル，リンクを **1** コマンドで実施する場合
以下の手順すべてを 1 コマンドで実施します。

- C/C++ 言語ソースファイル（tp1.c と tp2.c）を ccrx でコンパイルする
- コンパイル後，asrx でアセンブルする
- アセンブル後，rlink でリンクして，アブソリュートファイル（tp.abs）を作成する

【コマンド記述】

```
ccrx -isa=rxv1 -output=abs=tp.abs tp1.c tp2.c
```

備考 1. output オプションの出力形式指定を "-output=sty" に変えると，リンク後のファイルをモトローラ S 形式ファイルとして生成します。

備考 2. アブソリュートファイル生成過程で生じる中間ファイル（アセンブリ言語ファイルや，リロケータブルファイル）は残りません。生成されるファイルは，output オプションで指示した形式のファイルのみです。

備考 3. ccrx に対して，アセンブラ，最適化リンケージエディタにのみ有効なアセンブルオプションやリンクオプションを指示したい場合には，-asmcmd オプション，-lnkcmd オプション，-asmopt オプション，-lnkopt オプションを使用して指示してください。

備考 4. リンク対象のオブジェクトは，0 番地から配置します。セクションの並び順は保証されません。配置アドレスやセクションの配置順序を指示したい場合には，-lnkcmd オプション，-lnkopt オプションを使用して最適化リンケージエディタへオプション指示してください。

(b) コンパイルとアセンブルを **1** コマンドで実施する場合
以下の手順を 1 コマンドで実施し，別コマンドでリンカを起動して，tp.abs を作成します。

- C/C++ 言語ソースファイル（tp1.c と tp2.c）を ccrx でコンパイルする
- コンパイル後，asrx でアセンブルして，リロケータブルファイル（tp1.obj, tp2.obj）を作成する

【コマンド記述】

```
ccrx -isa=rxv1 -output=obj tp1.c tp2.c
rlink -form=abs -output=tp.abs -subcommand=cmd.sub tp1.obj tp2.obj
```

備考 1. ccrx に対して "-output=obj" オプションを指示すると，ccrx はリロケータブルファイルを生成します。

備考 2. リロケータブルファイル名を変更する場合は，ccrx へ C/C++ 言語ソースファイルをひとつずつ入力する必要があります。

備考 3. rlink の form オプションを，"-form=sty" に変えると，リンク後のファイルをモトローラ S 形式ファイルとして生成します。

(c) コンパイル，アセンブル，リンクを各々別コマンドで実施する場合
以下の個々の手順を，それぞれ 1 コマンドで実施します。

- C/C++ 言語ソースファイル（tp1.c と tp2.c）を ccrx でコンパイルして，アセンブリ言語ファイル（tp1.src, tp2.src）を作成する
- アセンブリ言語ファイル（tp1.src, tp2.src）を asrx でアセンブルして，リロケータブルファイル（tp1.obj, tp2.obj）を生成する
- リロケータブルファイル（tp1.obj, tp2.obj）を rlink でリンクして，アブソリュートファイル（tp.abs）を作成する

【コマンド記述】

```
ccrx -isa=rxv1 -output=src tp1.c tp2.c
asrx tp1.src tp2.src
rlink -form=abs -output=tp.abs -subcommand=cmd.sub tp1.obj tp2.obj
```

備考 ccrx に対して "-output=src" オプションを指示すると，ccrx はアセンブリ言語ファイルを生成します。

(d) アセンブルとリンクを **1** コマンドで実施する場合
以下の手順すべてを 1 コマンドで実施します。

- アセンブリ言語ファイル（tp1.src, tp2.src）を asrx でアセンブルする
- アセンブル後，rlink でリンクして，アブソリュートファイル（tp.abs）を作成する

【コマンド記述】

```
ccrx -isa=rxv1 -output=abs=tp.abs tp1.src tp2.src
```

備考 リンク対象のオブジェクトは，0 番地から配置します。セクションの並び順は保証されません。配置アドレスやセクションの配置順序を指示したい場合には，-lnkcmd オプション，-lnkopt オプションを使用して最適化リンケージエディタへオプション指示してください。

(e) アセンブルとリンクを別コマンドで実施する場合
以下の個々の手順を，それぞれ 1 コマンドで実施します。

- アセンブリ言語ファイル（tp1.src, tp2.src）を asrx でアセンブルして，リロケータブルファイル（tp1.obj, tp2.obj）を生成する
- リロケータブルファイル（tp1.obj, tp2.obj）を rlink でリンクして，アブソリュートファイル（tp.abs）を作成する

【コマンド記述 1】

```
ccrx -isa=rxv1 -output=obj tp1.src tp2.src
rlink -form=abs -output=tp.abs -subcommand=cmd.sub tp1.obj tp2.obj
```

【コマンド記述 2】

```
asrx -isa=rxv1 tp1.src tp2.src
rlink -form=abs -output=tp.abs -subcommand=cmd.sub tp1.obj tp2.obj
```

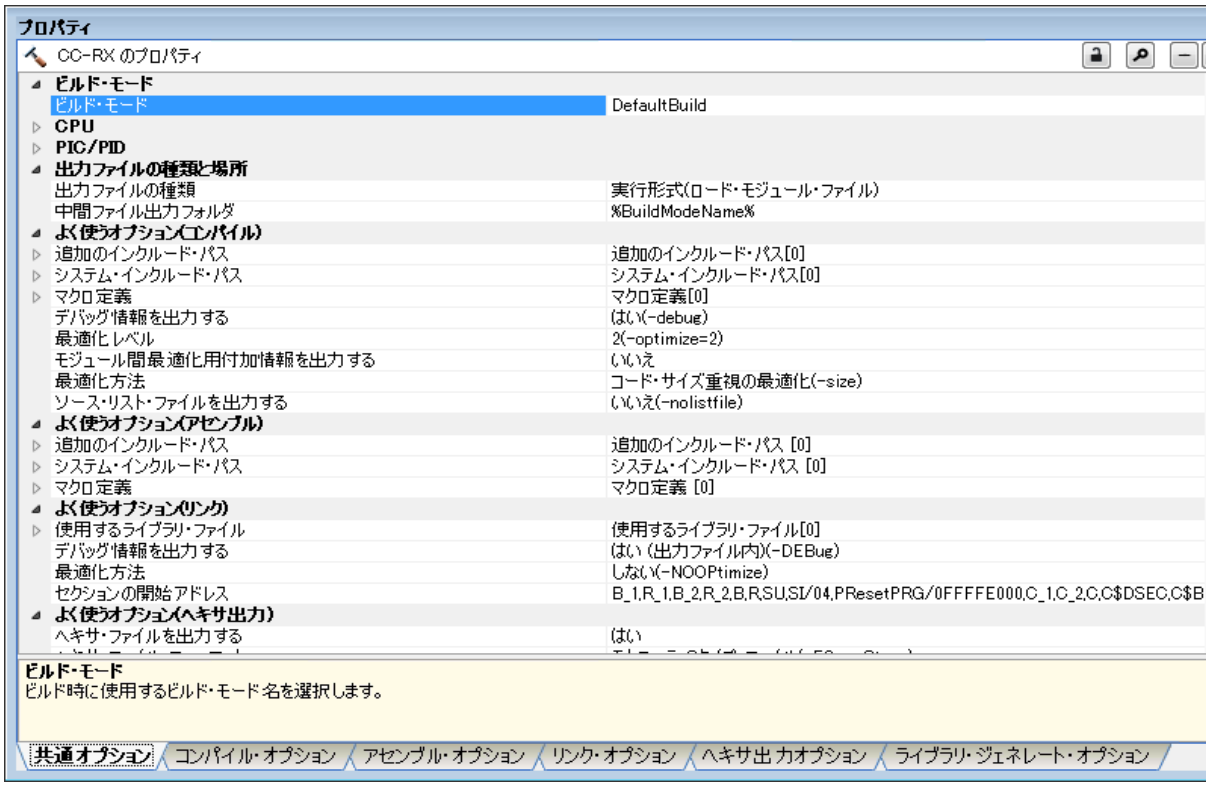
(3) CubeSuite+ でのオプション設定
CubeSuite+ から RX ファミリ用 C/C++ コンパイラのオプション CubeSuite+ のプロジェクト・ツリー パネルにおいて，ビルド・ツール・ノードを選択したのち，［表示］メニュー→［プロパティ］を選択すると，プロパティ パネルがオープンします。

(a) アプリケーション用のプロジェクトの場合
［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／［ライブラリ・ジェネレート・オプション］タブを選択します。

(b) ライブラリ用のプロジェクトの場合
［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／［ライブラリアン・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 RX ファミリ用 C/C++ コンパイラのオプションを設定することができます。

図 B.1 プロパティ パネル



(4) 環境変数 (コマンドプロンプト)
環境変数の一覧を以下に示します。

表 B.2 環境変数

No.	環境変数	説明	設定省略時の解釈
1	path	実行ファイルの格納ディレクトリを指定します。	省略不可
2	BIN_RX	ccrx を格納したディレクトリを指定します。	＜ccrx コマンドの格納ディレクトリ＞ lbgrx コマンド利用時は、省略不可
3	ISA_RX *1	命令セットアーキテクチャを選択します。 < 命令セットアーキテクチャ > RXV1 RXV2	省略時、値は設定されません。
4	INC_RX	コンパイラのインクルードファイル格納ディレクトリを指定します。	＜ccrx コマンドの格納ディレクトリ＞ \..\include
5	INC_RXA	アセンブラのインクルードファイル格納ディレクトリを指定します。	省略時、値は設定されません。
6	TMP_RX	テンポラリファイルを作成するディレクトリを指定します。	ccrx コマンド利用時は、%TEMP%

No.	環境変数	説明	設定省略時の解釈
7	HLNK_LIBRARY1 HLNK_LIBRARY2 HLNK_LIBRARY3	最適化リンケージエディタが使用するデフォルトライブラリ名を指定します。library オプションで指定したライブラリを優先してリンクします。その後未解決のシンボルがある場合、1,2,3 の順にデフォルトライブラリを検索します。	省略時、値は設定されません。
8	HLINK_TMP	最適化リンケージエディタがテンポラリファイルを作成するフォルダを指定します。この環境変数の指定がない場合は、カレントフォルダにテンポラリファイルを作成します。	省略時、値は設定されません。
9	HLINK_DIR	最適化リンケージエディタの入力ファイル格納フォルダを指定します。 input オプション、library オプションで指定したファイルの検索順序は、カレントフォルダ、HLNK_DIR 指定フォルダになります。ただし、ワイルドカードで指定したファイルは、カレントフォルダ内だけ検索します。	省略時、値は設定されません。
10	CPU_RX *1	CPU 種別を指定します。 <CPU 種別 > RX600 RX200	省略時、値は設定されません。

*1) 環境変数 ISA_RX と CPU_RX の両方を定義している場合は、ISA_RX の内容を優先します。

B.1.3 オプション

ここでは，RX ファミリ用 C/C++ コンパイラ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 → 「B.1.3.1 コンパイル・オプション」参照
アセンブル・フェーズ → 「B.1.3.2 アセンブル・オプション」参照
リンク・フェーズ → 「B.1.3.3 最適化リンケージエディタ（rlink）・オプション」参照
ライブラリ生成・フェーズ→「B.1.3.4 ライブラリジェネレータ・オプション」参照

B.1.3.1 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。

分類	オプション	説明
ソースオプション	-lang	ソース・ファイルをコンパイルする言語を選択します。
	-include	インクルード・ファイルの取り込み先フォルダを指定します。
	-preinclude	コンパイル単位の先頭にインクルードするファイルを指定します。
	-define	マクロ定義を指定します。
	-undefine	無効化するプリデファインド・マクロを指定します。
	-message	インフォメーションレベル・メッセージを出力します。
	-nomessage	抑止するインフォメーションレベル・メッセージ番号を指定します。
	-change_message	コンパイラ出力メッセージレベル変更します。
	-file_inline_path	(無効オプション) このオプションは V2.00 以降では使用できません。指定しても意味を持ちません。 【V.1.02 以前】ファイル間インライン展開ファイル取り込み先フォルダの指定
	-comment	コメント（/* */）のネストを許すかどうかを選択します。
	-check	M16C(R8C),H8(H8S,H8SX),SuperH ファミリ用の既存プログラムとの互換性をチェックします。
	-misra2004	MISRA-C:2004 ルールによるソースチェックをします。
	-ignore_files_misra	MISRA-C:2004 チェック対象外のファイルを指定します。
	-check_language_extension	拡張機能の使用によって部分抑止している MISRA-C:2004 ルールのチェックを有効にします。
オブジェクトオプション	-output	出力ファイル形式を選択します。
	-noline	プリプロセッサ展開時に #line の出力しません。
	-debug	オブジェクト・ファイルにデバッグ情報を出力します。
	-nodebug	オブジェクト・ファイルにデバッグ情報を出力しません。
	-section	変更するセクション名を指定します。
	-stuff	変数のアライメントに応じたセクションに配置します。
	-nostuff	変数のアライメントに応じたセクションに配置しません。
	-instalign4	分岐先を 4 バイトで命令実行向け整合します。
	-instalign8	分岐先を 8 バイトで命令実行向け整合します。
	-noinstalign	分岐先を命令実行向け整合しません。
	-nouse_div_inst	除算、剰余算に DIV,DIVU,FDIV 命令を使用しません。
リストオプション	-listfile	ソース・リスト・ファイルを出力します。
	-nolistfile	ソース・リスト・ファイルを出力しません。
	-show	ソース・リスト・ファイルの内容を指定します。

分類	オプション	説明
最適化オプション (1/2)	-optimize	最適化レベルを選択します。
	-goptimize	モジュール間最適化用付加情報を出力します。
	-speed	実行性能重視の最適化を実施します。
	-size	コード・サイズ重視の最適化を実施します。
	-loop	ループ展開の最大展開数を指定します。
	-inline	自動インライン展開行います。
	-noinline	自動インライン展開を行いません。
	-file_inline	(無効オプション) このオプションは V2.00 以降では使用できません。指定しても意味を持ちません。 【V.1.02 以前】ファイル間インライン展開対象ファイルの指定
	-case	switch 文のコード展開方式を選択します。
	-volatile	外部変数を volatile 化します。
	-novolatile	外部変数を volatile 化しません。
	-const_copy	const 修飾された外部変数の定数伝播を実施します。
	-noconst_copy	const 宣言された外部変数の定数伝播を実施しません。
	-const_div	整数型定数による除算および剰余算を変換します。
	-noconst_div	整数型定数による除算および剰余算を変換しません。
	-library	ライブラリ関数の実行方法を選択します。
	-scope	最適化範囲を複数に分割してコンパイルします。
	-noscope	最適化範囲を複数に分割しないでコンパイルします。
	-schedule	パイプライン処理を考慮した命令並べ替えを行います。
	-noschedule	命令並べ替えを行いません。

分類	オプション	説明
最適化オプション (2/2)	-map	外部変数アクセス最適化を行います。
	-smap	コンパイル単位内で定義された外部変数に対し、外部変数アクセス最適化を行います。
	-nomap	外部変数アクセス最適化を行いません。
	-approxdiv	浮動小数点定数除算の乗算化を行います。
	-enable_register	(無効オプション) このオプションは V2.00 以降では使用できません。指定しても意味を持ちません。 【V.1.02 以前】register 指定変数を優先的にレジスタ割り付け
	-simple_float_conv	浮動小数点型、整数型間の型変換を一部省略します。
	-fpu	浮動小数点演算命令を使用します。
	-nofpu	浮動小数点演算命令を使用しません。
	-alias	ポインタ指示先の型を考慮した最適化を実施するかどうかを選択します。
	-float_order	(無効オプション) このオプションは V2.00 以降では使用できません。指定しても意味を持ちません。 【V.1.02 以前】浮動小数点式の演算順序変更の最適化を行います。
	-ip_optimize	大域最適化を実施します。
	-merge_files	複数ソースのコンパイル結果をひとつのオブジェクトに出力します。
	-whole_program	指定ソースファイルをプログラム全体と仮定して最適化を実施します。

分類	オプション	説明
マイコンオプション	-isa	命令セット・アーキテクチャを選択します。
	-cpu	マイコン種別を選択します。
	-endian	エンディアン選択します。
	-round	浮動小数点定数演算の丸め方式を選択します。
	-denormalize	浮動小数点定数での非正規化数の扱いを選択します。
	-dbl_size	double 型、および long double 型の精度を選択します。
	-int_to_short	int を short に、unsigned int を unsigned short に置換します。
	-signed_char	char 型を signed char 型として扱います。
	-unsigned_char	char 型を unsigned char 型として扱います。
	-signed_bitfield	ビットフィールドの符号を signed で解釈します。
	-unsigned_bitfield	ビットフィールドの符号を unsigned で解釈します。
	-auto_enum	列挙型データのサイズを自動選択するかどうかを選択します。
	-bit_order	ビットフィールドメンバの並び順を選択します。
	-pack	構造体メンバ、クラスメンバのアライメント数を 1 にします。
	-unpack	構造体メンバ、クラスメンバのアライメント数をデータのアライメントに従います。
	-exception	例外処理機能を有効にします。
	-noexception	例外処理機能を無効にします。
	-rtti	C++ 実行時型情報（dynamic_cast、typeid）を有効または、無効を選択します。
	-fint_register	高速割り込み関数でのみ使用する汎用レジスタを選択します。
	-branch	分岐幅のサイズを選択します。
	-base	ROM,RAM 用ベースレジスタ選択します。
	-patch	CPU タイプ特有の問題を回避するかどうかを選択します。
	-pic	PIC 機能を有効にします。
	-pid	PID 機能を有効にします。
	-nouse_pid_register	PID レジスタをコード生成に使用しません。
	-save_acc	割り込み関数でアキュムレータ を退避・回復します。
アセンブル・リンクオプション	-asmcmd	asrx のオプションのサブコマンドファイルを指定します。
	-lnkcmd	rlink のオプションのサブコマンドファイルを指定します。
	-asmopt	asrx のオプションを指定します。
	-lnkopt	rlink のオプションを指定します。

分類	オプション	説明
その他オプション	-logo	コピーライトを出力します。
	-nologo	コピーライトの出力しません。
	-euc	入力プログラムの文字コードを EUC コードと解釈します。
	-sjis	入力プログラムの文字コードを SJIS コードと解釈します。
	-latin1	入力プログラムの文字コードを ISO-Latin1 コードと解釈します。
	-utf8	入力プログラムの文字コードを UTF-8 コードと解釈します。
	-big5	入力プログラムの文字コードを BIG5 コードと解釈します。
	-gb2312	入力プログラムの文字コードを GB2312 コードと解釈します。
	-outcode	出力アセンブリ言語ファイルの文字コードを選択します。
	-subcommand	コマンドオプションを取り込むファイルを指定します。

ソースオプション

＜コンパイル・オプション / ソースオプション＞

ソースオプションには，次のものがあります。

- -lang
- -include
- -preinclude
- -define
- -undefine
- -message
- -nomessage
- -change_message
- -file_inline_path (*) 無効オプション
- -comment
- -check
- -misra2004
- -ignore_files_misra
- -check_language_extension

-lang

<コンパイル・オプション / ソースオプション>

[指定形式]

```
-lang= { c| cpp| ecpp| c99 }
```

- 省略時解釈
  拡張子が cpp，cc，cp のときには C++ 言語ソースファイルとしてコンパイルし，それ以外の場合は C（C89）言語ソースファイルとしてコンパイルします。
  ただし，拡張子が src，s の場合は本オプション指定に関わらずアセンブリ言語ファイルとして扱います。

[詳細説明]

- ソースファイルの言語を指定します。
- lang=c オプション指定時は，C （C89）言語ソースファイルとしてコンパイルします。
- lang=cpp オプション指定時は，C++ 言語ソースファイルとしてコンパイルします。
- lang=ecpp オプション指定時は，Embedded C++ 言語ソースファイルとしてコンパイルします。
- lang=c99 オプション指定時は，C （C99）言語ソースファイルとしてコンパイルします。

[備考]

- Embedded C++ 言語仕様では，catch，const_cast，dynamic_cast，explicit，mutable，namespace，reinterpret_cast，static_cast，template，throw，try，typeid，typename，using，多重継承，仮想基底クラスをサポートしていません。これらを記述した場合，エラーメッセージを出力します。
- EC++ ライブラリを使用する場合は，必ず lang=ecpp オプションを指定してください。
- 一括コンパイル (入力ファイルに C/C++ 言語ソースファイルを複数同時に入力) では、個々の C/C++ 言語ソースファイルに異なる言語を指定することはできません。指定したい言語ごとに C/C++ 言語ソースを分け、それぞれの言語で本オプションを指定して一括コンパイルを行ってください。

-include

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-include = <パス名>[,…]
```

［詳細説明］

- インクルードファイルの存在するパス名を指定します。
  - パス名が複数ある場合にはカンマ（,）で区切って指定することができます。
- 「<」、「>」で囲まれたファイルの検索は，include オプション指定フォルダ，環境変数 INC_RX 指定フォルダの順序で行います。
- 「"」、「"」で囲まれたファイルの検索は，#include 行を記述しているファイルの格納フォルダ，include オプション指定フォルダ，環境変数 INC_RX 指定フォルダの順序で行います。
- include オプション指定フォルダが複数ある場合、パス名をコマンドラインに指定した順 (左から右の順) に検索します。
- 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［ソース］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］

-preinclude

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-preinclude = < ファイル名 >[,…]
```

［詳細説明］

- 指定したファイルの内容をコンパイル単位の先頭に取り込みます。
- ファイル名が複数ある場合にはカンマ（,）で区切って指定することができます。
- preinclude オプション指定フォルダが複数ある場合，左に指定したものから順に検索を行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［コンパイル単位の先頭にインクルードするファイル］

［備考］

- 本オプションを複数回指定した場合，指定したすべてのファイルが取り込み対象となります。

-define

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-define = <sub>[,…]
          <sub> : < マクロ名 > [= < 文字列 >]
```

［詳細説明］

- ソースファイル内で記述する #define と同等の効果を得ます。
- < マクロ名 >=< 文字列 > と記述することで < 文字列 > をマクロ名として定義できます。
- サブオプションに < マクロ名 > を単独で指定した場合は，そのマクロ名が定義されたものと仮定します。
- < 文字列 > には，名前または整定数を記述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。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［マクロ定義］

［備考］

- 本オプションで指定したマクロ名がソース中で #define により既に定義されている場合，#define を優先します。
- 本オプションを複数回指定した場合，指定したすべてのマクロ名が有効となります。

-undefine

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-undefine = <sub>[,…]
          <sub> : < マクロ名 >
```

［詳細説明］

- ＜マクロ名＞のプリデファインドマクロを無効化します。
- マクロ名が複数ある場合にはカンマ（,）で区切って指定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。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［無効化するプリデファインド・マクロ］

［備考］

- 指定可能なプリデファインドマクロについては，RX コーディング編の「マクロ名」の項目を参照ください。
- 本オプションを複数回指定した場合，指定したすべてのマクロ名が未定義となります。

-message

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-message
```

［詳細説明］

- インフォメーションレベル・メッセージを出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］

［備考］

- 本オプションを指定してアセンブラや最適化リンケージエディタのメッセージ出力を制御することはできません。
- 最適化リンケージエディタのメッセージについては，lnkcmd オプションにより，最適化リンケージエディタの message オプションおよび nomessage オプションを指定することで出力制御が可能です。

-nomessage

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-nomessage[= < エラー番号 > [- < エラー番号 >][,…]
```

［詳細説明］

- nomessage オプションを指定した場合，インフォメーションレベル・メッセージの出力を抑止します。
- サブオプションでエラー番号を指定すると，指定したインフォメーションレベル・メッセージの出力だけを抑止します。
- エラー番号が複数ある場合にはカンマ（,）で区切って指定することができます。
- < エラー番号 >-< エラー番号 > のようにハイフン（-）で抑止するエラー番号の範囲を指定することもできます。
- エラー番号には、インフォメーションを表す「M」から始まるメッセージ番号の、末尾 (右側) の 5 桁を指定してください。
  例) インフォメーションメッセージ M0523009 の場合
  -nomessage=23009
- 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［ソース］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］，［抑止するインフォメーションレベル・メッセージ番号］

［備考］

- 本オプションを指定してアセンブラや最適化リンケージエディタのメッセージ出力を制御することはできません。
- 最適化リンケージエディタのメッセージについては，lnkcmd オプションにより，最適化リンケージエディタの message オプションおよび nomessage オプションを指定することで出力制御が可能です。
- nomessage オプションを複数回指定した場合，指定したすべてのエラー番号について抑止します。
- 本オプションは、メッセージの番号 (コンポーネント番号を含む) が 0510000 ～ 0549999 であるもののみ対象とします。
- 番号が 0520000 ～ 0529999 以外の警告メッセージは、本オプションに番号を指定しても抑止できません。同時に -chagne_message を用いてインフォメーションに変更してください。

-change_message

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-change_message= <sub>[,…]
                    <sub> : < エラーレベル >[=< エラー番号 >[- < エラー番号 >][,…]]
                            < エラーレベル > : { information | warning | error }
```

［詳細説明］

- インフォメーション，ウォーニングのメッセージ・レベルを変更します。
- エラー番号が複数ある場合にはカンマ（,）で区切って指定することができます。
- エラー番号には、インフォメーションを表す「M」または警告を表す「W」から始まるメッセージ番号の、末尾 (右側) の 5 桁を指定してください。
  例) インフォメーションメッセージ M0523009 の場合
  -change_message=error=23009
- 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［ソース］カテゴリの［ウォーニング・メッセージをインフォメーションレベルに変更する］，［ウォーニングレベルのエラー番号］，［インフォメーション・メッセージをウォーニングレベルに変更する］，［インフォメーション・メッセージをウォーニングレベルに変更する］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニング・メッセージをエラーレベルに変更する］，［インフォメーション、ウォーニングレベルのエラー番号］

［例］

```
change_message=information= エラー番号
```

- ウォーニングレベルの指定エラー番号のみインフォメーションレベルに変更します。

```
change_message=warning= エラー番号
```

- インフォメーションレベルの指定エラー番号のみウォーニングレベルに変更します。

```
change_message=error= エラー番号
```

- インフォメーション，ウォーニングレベルの指定エラー番号のみエラーレベルに変更します。

```
change_message=information
```

- すべてのウォーニングメッセージをインフォメーションレベルに変更します。

```
change_message=warning
```

- すべてのインフォメーションメッセージをウォーニングレベルに変更します。

```
change_message=error
```

- すべてのインフォメーション，ウォーニングメッセージをエラーレベルに変更します。

［備考］

- インフォメーションレベルに変更したメッセージについては，nomessage オプション指定により出力を抑止できます。
- 本オプションを指定してアセンブラや最適化リンケージエディタのメッセージ出力を制御することはできません。
- 最適化リンケージエディタのメッセージについては，lnkcmd オプションにより，最適化リンケージエディタの message オプションおよび nomessage オプションを指定することで出力制御が可能です。
- 本オプションを複数回指定した場合，指定したすべてのエラー番号について有効になります。
- 警告メッセージおよびインフォメーションメッセージ以外のメッセージは対象外です。本オプションに指定しても無視します。
- misra2004 オプション指定時に表示する，MISRA 検出メッセージ（記号 (M) を表示）は本オプション制御対象外です。
- 一部のメッセージでエラー (E) や警告 (W) などのメッセージ種別が変化することがあります。メッセージ種別が変わっても、番号 (コンポーネント番号およびメッセージ番号) で、メッセージの意味は変わりません。

-file_inline_path

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-file_inline_path= < パス名 >[,…]
```

［詳細説明］

- V2.00 で、本オプションは無効になりました。指定した場合、無視されますが、従来バージョンとの互換性のためエラーにはなりません。

-comment

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-comment = { nest | nonest }
```

- 省略時解釈
  comment=nonest です。

［詳細説明］

- comment=nest を指定した場合，ネストしたコメントの記述を可能にします。
- comment=nonest を指定した場合，ネストしたコメントを記述するとエラーになります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［コメント (/* */) のネストを許す］

［例］

- comment=nest を指定した場合はすべてコメントと解釈しますが，comment=nonest を指定した場合は [1] でコメントが終わっていると解釈します。

```
/* This is an example of /* nested */ comment */
                                  [1]
```

-check

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-check = { nc | ch38 | shc}
```

［詳細説明］

- R8C, M16C ファミリ用 C/C++ コンパイラ，H8, H8S, H8SX ファミリ用 C/C++ コンパイラおよび SuperH ファミリ用 C/C++ コンパイラ向けにコーディングした C/C++ 言語ソースファイルを本コンパイラへ流用する際，互換性に影響するオプション指定，ソース記述をチェックすることができます。
- check=nc では，R8C,M16C ファミリ用 C コンパイラとの互換性をチェックします。
チェックされるオプションや型には，次のようなものがあります。
- オプション：signed_char, signed_bitfield, bit_order=left, endian=big, dbl_size=4
- inline，enum 型，#pragma BITADDRESS，#pragma ROM，#pragma PARAMETER，asm()
- -int_to_short の指定がないときに，signed short 範囲外の定数を int,signed int 型へ代入，あるいは unsigned short 範囲外の定数を int 型または unsigned int 型へ代入
- signed short, unsigned short 共範囲外の定数を long, long long 型へ代入
- signed short 範囲外の定数と int, short, char 型（char 型は符号付き除く）との比較式
- check=ch38 では，H8,H8S,H8S ファミリ用 C/C++ コンパイラとの互換性をチェックします。
チェックされるオプションや型には，次のようなものがあります。
- オプション：unsigned_char, unsigned_bitfield, bit_order=right, endian=little, dbl_size=4
- __asm，#pragma unpack
- signed long の最大値より大きな定数との比較式
- -int_to_short の指定がないときに，signed short 範囲外の定数を int,signed int 型へ代入，あるいは unsigned short 範囲外の定数を int 型または unsigned int 型へ代入
- signed short, unsigned short 共範囲外の定数を long, long long 型へ代入
- signed short 範囲外の定数と int, short, char 型（char 型は符号付き除く）との比較式
- check=shc では，SuperH ファミリ用 C/C++ コンパイラとの互換性をチェックします。
チェックされるオプションや型には，次のようなものがあります。
- オプション：unsigned_char, unsigned_bitfield, bit_order=right, endian=little, dbl_size=4 , round=nearest
- #pragma unpack
- volatile 修飾した変数
- 表示された項目により，それぞれ次の項目を確認ください。
- オプション：言語仕様で規定されていない実装依存の内容がコンパイラ間で異なっています。メッセージで出力されたオプションの選択を確認してください。
- 拡張仕様：プログラムの動作に影響を及ぼす可能性がある拡張仕様です。メッセージで出力された拡張仕様の記述を確認してください。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［既存プログラムとの互換性をチェックする］

［備考］

- dbl_size=4 が有効な時に，R8C, M16C ファミリ用 C コンパイラ，H8, H8S, H8SX ファミリ用 C/C++ コンパイラおよび SuperH ファミリ用 C/C++ コンパイラと浮動小数点関連の変換 / ライブラリの計算結果が異なる場合があります。

dbl_size=4 は，本コンパイラでは，double 型および long double 型を 32 ビットにしますが，各種 R8C, M16C ファミリ用 C コンパイラ（fdouble_32），H8, H8S, H8SX ファミリ用 C/C++ コンパイラ（double=float）および SuperH ファミリ用 C/C++ コンパイラ（double=float）では，double 型のみ 32 ビットにします。

- unsigned int 型と long 型をオペランドとする二項演算（加減乗除や比較など）に対する結果が，SuperH ファミリ用 C/C++ コンパイラと異なる場合があります。
  本コンパイラではオペランドを unsigned long 型に変換してから演算しますが，SuperH ファミリ用 C/C++ コンパイラ（ただし，strict_ansi を指定しないとき）では，signed long long 型に変換してから演算します。
- volatile 修飾した変数に対し，読み出しや書き込みのサイズが，SuperH ファミリ用 C/C++ コンパイラと異なる場合があります。
  volatile 修飾したビットフィールドは，本コンパイラでは宣言型より小さなサイズでアクセスすることがありますが，SuperH ファミリ用 C/C++ コンパイラでは宣言型のサイズ通りにアクセスします。
- 構造体およびビットフィールドメンバの割り付けについては，本オプションでメッセージを出力しません。割り付けを意識した宣言をしている場合には，RX コーディング編の「3.1.4 データの内部表現と領域」の項目を参照してください。
- R8C, M16C ファミリ用 C コンパイラ（fextend_to_int を指定しない）では，条件式で汎整数拡張を行わずに評価したコードを生成するので，本コンパイラの生成コードと動作が異なる場合があります。

-misra2004

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-misra2004= {
              all
              | apply=< ルール番号 >[,< ルール番号 >,...]
              | ignore=< ルール番号 > [,< ルール番号 >,...]
              | required
              | required_add=< ルール番号 >[,< ルール番号 >,...]
              | required_remove=< ルール番号 >[,< ルール番号 >,...]
              | < ファイル名 >}
```

［詳細説明］

- MISRA-C:2004 のルールチェック機能を有効にし，そのチェック対象を指定します。
- -misra2004=all オプション指定時は，サポートしているすべてのルールをチェック対象とします。
- -misra2004=apply< ルール番号 >[,< ルール番号 >,...] オプション指定時は，サポートしているルールのうち，指定されたルール番号をチェック対象とします。
- -misra2004=ignore=< ルール番号 >[,< ルール番号 >,...] オプション指定時は，サポートしているルールのうち，指定されたルール番号以外のルールをチェック対象とします。
- -misra2004=required オプション指定時は，サポートしているルールのうち，ルールの分類が “required” になっているルールをチェック対象とします。
- -misra2004=required_add=< ルール番号 >[,< ルール番号 >,...] オプション指定時は，サポートしているルールのうち，ルールの分類が “required” になっているルールと指定されたルール番号をチェック対象とします。
- -misra2004=required_remove=< ルール番号 >[,< ルール番号 >,...] オプション指定時は，サポートしているルールのうち，ルールの分類が “required” になっているルールから指定されたルール番号を除いたルール番号をチェック対象とします。
- -misra2004 =< ファイル名 > オプション指定時は，サポートしているルールのうち，指定されたファイル名に記載されたルール番号をチェック対象とします。ファイル名で指定されたファイル内の記述は，1 ルールを 1 行で記述します。
- ルール番号は，10 進数値およびピリオド（.）で指定してください。
- MISRA-C:2004 のチェック項目に該当した場合，次の形式でメッセージを表示します。
  ファイル名（*行番号*）: M0523028 Rule *ルール番号*: *メッセージ*
- -misra2004=< ファイル名 > の指定を複数回行った場合は、最後に指定したファイルのみ有効になります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［MISRA C ルール検査］カテゴリの［適用するルール］，［ルール番号記載ファイル］，［ルール番号］，［除外するルール番号］，［必須ルールの他にチェックするルール番号］，［必須ルールから除外するルール番号］

［備考］

- -misra2004 オプションは複数回指定できます。ただし、指定された種類 (-misra2004= 以降に指定する名前) が複数ある場合は、最後に記述した種類と同じものだけを有効とします。
  (指定例)
  ... -misra2004=ignore=2.2 -misra2004=apply=2.3
  -misra2004=required_add=4.1 -misra2004=apply=4.2
  -misra2004=apply=5.2 ...
  ignore, apply, required_add の 3 種類の -misra2004 が同時に指定されていますが、最後に連続して指定した 2 つの apply の指定のみが有効になります。その結果、ルール 4.2 と 5.2 がチェック対象になります。

- サポートしていないルール番号を，各オプションの < ルール番号 > に指定した場合はエラー C0523031 となり，そこで処理を中止します。
- -misra2004=< ファイル名 > で指定したルールファイルがオープンできない場合，エラー C0523029 となります。また，ルールファイルからルール番号が取り出せない場合は，エラー C0523030 となり，いずれの場合もそこで処理を中止します。
- -lang オプションで cpp,c99 または ecpp が選択された場合，本オプションを無視します。
- -output=prep と同時に指定した場合，本オプションを無視します。
- 本オプションでサポートしている MISRA-C:2004 のルール番号を次に示します。

[required であるルール番号]
2.2 2.3
4.1 4.2
5.2 5.3 5.4
6.1 6.2 6.4 6.5
7.1
8.1 8.2 8.3 8.5 8.6 8.7 8.11 8.12
9.1 9.2 9.3
10.1 10.2 10.3 10.4 10.5 10.6
11.1 11.2 11.5
12.3 12.4 12.5 12.7 12.8 12.9 12.10 12.12
13.1 13.3 13.4
14.2 14.3 14.4 14.5 14.6 14.7 14.8 14.9 14.10
15.1 15.2 15.3 15.4 15.5
16.1 16.3 16.5 16.6 16.9
18.1 18.4
19.3 19.6 19.8 19.11 19.14 19.15
20.4 20.5 20.6 20.7 20.8 20.9 20.10 20.11 20.12

[required ではないルール番号]
5.5 5.6
6.3
11.3 11.4
12.1 12.6 12.11 12.13
13.2
17.5
19.7 19.13

- #pragma などの拡張機能を用いたソースでは，ルールチェックの一部が抑止されます。
  抑止される内容は，check_language_extension オプションの項目を参照してください。
- misra2004 オプションで表示される MISRA 診断メッセージは，change_message オプションで制御することはできません。

-ignore_files_misra

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-ignore_files_misra=< ファイル名 >[,< ファイル名 >,...]
```

［詳細説明］

- MISRA-C:2004 のルールチェック対象外のソースファイルを指定します。
- 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［MISRA C ルール検査］カテゴリの［ルール・チェック対象外のファイル］

［備考］

- コマンドライン上で同一オプションを複数回指定した場合には，それぞれのオプションが有効になります。
- -misra2004 オプションの指定がない場合は，本オプションを無視します。
- 指定されたソースファイル名が，コンパイル対象でない場合は，指定されたソースファイル名を無視します。

-check_language_extension

＜コンパイル・オプション / ソースオプション＞

［指定形式］

```
-check_language_extension
```

［詳細説明］

- C/C++ 言語規格から独自に拡張した言語仕様が有効な場合に一部抑止される，MISRA2004 ルールチェックを有効にします。
- 本コンパイラでは，misra2004 オプションのデフォルトでは，次の場合はチェックが抑止されます。これをチェックしたい場合は，misra2004 オプション指定時に，check_language_extension オプションを指定してください。
- プロトタイプ宣言がない場合（ルール 8.1）で，該当関数に #pragma entry または #pragma interrupt のいずれかの指定があるとき。
- 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［MISRA C ルール検査］カテゴリの［拡張キーワードや拡張仕様をメッセージ出力する］

［例］

```
#pragma interrupt vfunc
extern void service(void);
void vfunc(void)
{
    service();
}
```

- 関数 vfunc にはプロトタイプ宣言がありませんが，#pragma interrupt の対象なので -misra2004=all を指定してコンパイルしても，-check_language_extension の指定がないと，ルール 8.1 のチェックメッセージが表示されません。

［備考］

- -misra2004 オプションの指定がない場合は，本オプションを無視します。

オブジェクトオプション

＜コンパイル・オプション / オブジェクトオプション＞

オブジェクトオプションには，次のものがあります。

- -output
- -noline
- -debug
- -nodebug
- -section
- -stuff
- -nostuff
- -instalign4
- -instalign8
- -noinstalign
- -nouse_div_inst

-output

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-output = <sub> [=< ファイル名 >]
        <sub> : { prep | src | obj | abs | hex | sty }
```

- 省略時解釈
  output=obj です。

［詳細説明］

- 出力ファイルの形式を指定します。
- サブオプションと出力ファイルの一覧を以下に示します。
- < ファイル名 > を指定しない場合は，先頭に入力したソースファイル名に以下表の拡張子をつけたファイルを作成します。

表 B.3 サブオプション出力形式

サブオプション	出力形式	ファイル名指定省略時の拡張子
prep	プリプロセッサ展開後のソースファイル	C（C89, C99）言語ソースファイル：p C++ 言語ソースファイル：pp
src	アセンブリ・ソース・ファイル	src
obj	リロケータブル・ファイル	obj
abs	アブソリュート・ファイル	abs
hex	インテル HEX 形式ファイル	hex
sty	モトローラ S 形式ファイル	mot

注 リロケータブルファイルは，アセンブラの出力ファイルです。
アブソリュートファイル，インテル HEX 形式ファイル，およびモトローラ S 形式ファイルは，最適化リンケージエディタの出力ファイルです。

- 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［オブジェクト］カテゴリの［出力ファイル形式］

［備考］

- 指定した形式のファイルを作成するための中間ファイルは，フォルダ指定があればそのフォルダに，フォルダ指定がなければカレントフォルダに作成します。

-noline

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-noline
```

［詳細説明］

- プリプロセッサ展開時に #line 出力を抑止します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［オブジェクト］カテゴリの［出力ファイル形式］

［備考］

- 本オプションは output=prep オプションの指定が無い場合は無効となります。

-debug

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-debug
```

［詳細説明］

- debug オプションを指定した場合，C ソース・レベル・デバッグに必要なデバッグ情報を出力します。
- debug オプションは，最適化オプションを指定した場合も有効となります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［オブジェクト］カテゴリの［デバッグ情報を出力する］

-nodebug

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-nodebug
```

［詳細説明］

- nodebug オプションを指定した場合，デバッグ情報を出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［オブジェクト］カテゴリの［デバッグ情報を出力する］

-section

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-section = <sub>[,…]
           <sub>: { P = < セクション名 > |
                    C = < セクション名 > |
                    D = < セクション名 > |
                    B = < セクション名 > |
                    L = < セクション名 > |
                    W = < セクション名 > }
```

- 省略時解釈
  section=P=P,C=C,D=D,B=B,L=L,W=W です。

［詳細説明］

- セクション名を指定します。
- section=P =< セクション名 > は，プログラム領域のセクション名を指定します。
- section=C =< セクション名 > は，定数領域のセクション名を指定します。
- section=D =< セクション名 > は，初期化データ領域のセクション名を指定します。
- section=B =< セクション名 > は，未初期化データ領域のセクション名を指定します。
- section=L =< セクション名 > は，リテラル領域のセクション名を指定します。
- section=W=< セクション名 > は，switch 文分岐テーブル領域のセクション名を指定します。
- < セクション名 > は，英字，数字，下線（_），または $ の列で，先頭が数字以外のものです。
- 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［オブジェクト］カテゴリの［プログラム領域のセクション名］，［定数領域のセクション名］，［初期化データ領域のセクション名］，［未初期化データ領域のセクション名］，［リテラル領域のセクション名］，［switch 文分岐テーブル領域のセクション名］

［備考］

- V.1.00 と同様に L セクションを出力せず，リテラル領域を C セクションに出力したい場合は，section=L=C を選択してください。
- L セクションを C セクションと同じ名前のセクション名に変更する場合を除き，領域が異なるセクションに同じセクション名を指定できません。
- セクション名長の翻訳限界については，「 翻訳限界」を参照してください。

-stuff

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-stuff
```

［詳細説明］

- stuff オプションを指定した場合，すべての変数をアライメント数に応じてアライメント数が 4 のセクション，2 のセクション，1 のセクションに配置します（表 B-3）。

表 B.4 **stuff** オプション指定時の，各変数と出力先セクションの関係

変数の種類	変数のアライメント数	変数が所属するセクション
const 修飾変数	4	C
	2	C_2
	1	C_1
初期値あり変数	4	D
	2	D_2
	1	D_1
初期値なし変数	4	B
	2	B_2
	1	B_1
switch 文分岐テーブル	4	W
	2	W_2
	1	W_1

- C，D，B は section オプションまたは #pragma section で指定したセクション名になります。
- W は section オプションで指定したセクション名になります。C セクション内で初期値がない変数が初期値のある変数の後に出力されることを除き，各セクション内のデータは定義順に出力されます。

［例］

```
int a;
char b=0;
const short c=0;
struct {
        char x;
        char y;
} ST;
```

```
        .SECTION                C_2,ROMDATA,ALIGN=2
        .glb        _c
_c:
        .word                   0000H
        .SECTION                D_1,ROMDATA
        .glb        _b
_b:
        .byte                   00H
        .SECTION                B,DATA,ALIGN=4
        .glb        _a
_a:
        .blkl       1
        .SECTION                B_1,DATA
        .glb        _ST
_ST:
        .blkb       2
```

［備考］

- - sutff オプションは B,D,C および W 以外のセクションに対しては無効です。

-nostuff

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-nostuff [= < セクション種別 >[,…]]
            < セクション種別 > : { B | D | C | W}
```

［詳細説明］

- nostuff オプションを指定した場合，指定した < セクション種別 > に属する変数をアライメント数が 4 のセクションに配置します。
- < セクション種別 > を省略した場合は，すべてのセクション種別の変数が対象になります。
- C，D，B は section オプションまたは #pragma section で指定したセクション名になります。
- W は section オプションで指定したセクション名になります。
- C セクション内で初期値がない変数が初期値のある変数の後に出力されることを除き，各セクション内のデータは定義順に出力されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［オブジェクト］カテゴリの［初期値なし変数をアライメント 4 のセクションに配置する］，［初期値あり変数をアライメント 4 のセクションに配置する］，［const 修飾変数をアライメント 4 のセクションに配置する］，［switch 文分岐テーブルをアライメント 4 のセクションに配置する］

［例］

```
int a;
char b=0;
const short c=0;
struct {
        char x;
        char y;
} ST;
```

```
        .SECTION                C,ROMDATA,ALIGN=4
        .glb        _c
_c:
        .word                   0000H
        .SECTION                D,ROMDATA,ALIGN=4
        .glb        _b
_b:
        .byte                   00H
        .SECTION                B,DATA,ALIGN=4
        .glb        _a
_a:
        .blkl       1
        .glb        _ST
_ST
        .blkb       2
```

［備考］

- nostuff オプションに B,D,C および W 以外のセクションを指定することはできません。

-instalign4

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-instalign4[={loop|inmostloop}]
```

- 省略時解釈
  分岐先の命令実行向け整合を行いません。

［詳細説明］

- 分岐先の命令実行向け整合を行います。
- instalign4 を指定した場合は 4 バイト配置アドレスを命令実行向け整合します。
- 命令実行向け整合とは，-instalign4 のサブオプションで選択した種類の分岐先の命令が，アライメント数（4）の倍数であるアドレスをまたいでいる場合（*1）にだけ整合を取るものです。
- 分岐先の種類は，-instalign4 のサブオプションの指定により次の 3 種類から選択します（*2）。
- 指定なし : 関数先頭，switch 文の case および default ラベル
- inmostloop: 各最内周ループの先頭，関数先頭，switch 文の case および default ラベル
- loop: 各ループの先頭，関数先頭，switch 文の case および default ラベル
- 本オプションを選択すると，プログラムセクションのアライメント数が 1 から 4 に変わります。
- 本オプションは，分岐先命令のアドレス整合することで RX CPU の命令キューを効率よく動作させ，プログラムの実行速度向上を図るものです。
- 次の用途を想定した仕様となっております。
- instalign4 … 命令キューが 32 ビットの CPU（主に RX200 シリーズ）で速度向上を図る場合
- 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［オブジェクト］カテゴリの［分岐先の命令実行向け整合］

注 1. 命令サイズがアライメント数以下の場合です。命令サイズがアライメント数よりも大きい場合は，またいでいる箇所が 2 以上の場合にだけ整合を取ります。

注 2. ここに挙げたもの以外の分岐先は，命令実行向け整合の対象ではありません。たとえば，loop を選択した場合はループの先頭は対象ですが，ループ内にあるループを構成しない if 文などの分岐先は対象ではありません。

-instalign8

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-instalign8[={loop|inmostloop}]
```

- 省略時解釈
  分岐先の命令実行向け整合を行いません。

［詳細説明］

- 分岐先の命令実行向け整合を行います。
- instalign8 を指定した場合は 8 バイトでそれぞれ配置アドレスを命令実行向け整合します。
- 命令実行向け整合とは，-instalign8 のサブオプションで選択した種類の分岐先の命令が，アライメント数（8）の倍数であるアドレスをまたいでいる場合（*1）にだけ整合を取るものです。
- 分岐先の種類は，-instalign8 のサブオプションの指定により次の 3 種類から選択します（*2）。
- 指定なし : 関数先頭，switch 文の case および default ラベル
- inmostloop: 各最内周ループの先頭，関数先頭，switch 文の case および default ラベル
- loop: 各ループの先頭，関数先頭，switch 文の case および default ラベル
- 本オプションを選択すると，プログラムセクションのアライメント数が 1 から 8 に変わります。
- 本オプションは，分岐先命令のアドレス整合することで RX CPU の命令キューを効率よく動作させ，プログラムの実行速度向上を図るものです。
- 次の用途を想定した仕様となっております。
- instalign8 … 命令キューが 64 ビットの CPU（主に RX600 シリーズ）で速度向上を図る場合
- 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［オブジェクト］カテゴリの［分岐先の命令実行向け整合］

注 1. 命令サイズがアライメント数以下の場合です。命令サイズがアライメント数よりも大きい場合は，またいでいる箇所が 2 以上の場合にだけ整合を取ります。

注 2. ここに挙げたもの以外の分岐先は，命令実行向け整合の対象ではありません。たとえば，loop を選択した場合はループの先頭は対象ですが，ループ内にあるループを構成しない if 文などの分岐先は対象ではありません。

［例］

- ＜C ソース＞

```
long a;
int f1(int num)
{
    return (num+1);
}
void f2(void)
{
    a = 0;
}
void f3(void)
{
}
```

- ＜出力コード＞
  [-instalign8 を指定してコンパイルした場合]
  下記は，各関数の先頭を，8 バイトで命令実行向け整合させた場合の例です。
  8 バイトの命令実行向け整合では，対象命令が 8 バイトの境界をまたがない限り，配置アドレスを変更しないため，
  実際に整合がかかるのは関数 f2 のアドレスだけです。

```
    .SECTION P,CODE,ALIGN=8
    .INSTALIGN 8
_f1:                        ; 関数 f1。配置アドレス =0000H
    ADD     #01H,R1         ; 2 バイト
    RTS                     ; 1 バイト
    .INSTALIGN 8
_f2:                        ; 関数 f2。配置アドレス =0008H * 整合有り
                            ; 0003H に 6 バイト命令が来ると，8 バイト境界を
                            ; またぐため，整合が取られる。
    MOV.L   #_a,R4          ; 6 バイト
    MOV.L   #0,[R4]         ; 3 バイト
    RTS                     ; 1 バイト
    .INSTALIGN 8
_f3:                        ; 関数 f3。配置アドレス =0012H
     RTS
    .END
```

-noinstalign

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-noinstalign
```

［詳細説明］

- 分岐先の命令実行向け整合を行いません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［オブジェクト］カテゴリの［分岐先の命令実行向け整合］

-nouse_div_inst

＜コンパイル・オプション / オブジェクトオプション＞

［指定形式］

```
-nouse_div_inst
```

［詳細説明］

- プログラム中のすべての除算および剰余算を，DIV 命令，DIVU 命令および FDIV 命令を使わないコードを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［オブジェクト］カテゴリの［除算、剰余算を DIV,DIVU,FDIV 命令で生成する］

［備考］

- 本オプション指定時は，DIV,DIVU および FDIV 命令を生成する代わりに，それぞれの命令に相当する処理を行うランタイム関数の呼び出しに置き換えます。
- このため，ROM サイズやコード実行速度といったコード効率が悪くなる場合があります。

リストオプション

＜コンパイル・オプション / リストオプション＞

リストオプションには，次のものがあります。

- -listfile
- -nolistfile
- -show

-listfile

＜コンパイル・オプション / リストオプション＞

［指定形式］

```
-listfile[={< ファイル名 >|< パス名 >}]
```

- 省略時解釈
  ソースファイルと同じファイル名で，拡張子が lst のソースリストファイルを作成します。

［詳細説明］

- listfile オプションを指定した場合，ソースリストファイルを出力します。＜ファイル名＞を指定することもできます。
- < ファイル名 > の代わりに、あらかじめ存在するフォルダを < パス名 > として指定することもできます。この場合は、ソースリストファイル名としてコンパイルまたはアセンブル対象のソースファイルの拡張子を .lst に変更したものを、< パス名 > に選択したフォルダに出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［リスト］カテゴリの［ソース・リスト・ファイルを出力する］
- ［備考］
- 本オプションを指定してリンケージリストを出力することはできません。
- リンケージリストを出力するには，lnkcmd オプションにより最適化リンケージエディタの list オプションを指定してください。
- コンパイラが出力する情報は，ソースリストに書き込まれます。
- ソース・リスト・ファイルのフォーマットについては，「アセンブリ・リスト・ファイル」を参照ください。
- < パス名 > を使用する場合は、パス名に該当するフォルダはあらかじめ作成しておいてください。存在しない場合は、listfile には < パス名 > の代わりに < ファイル名 > が選択されたものとみなします。

-nolistfile

＜コンパイル・オプション / リストオプション＞

［指定形式］

```
-nolistfile
```

［詳細説明］

- ソースリストファイルは出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［リスト］カテゴリの［ソース・リスト・ファイルを出力する］

-show

＜コンパイル・オプション / リストオプション＞

［指定形式］

```
-show=<sub>[,…]
      <sub> : { source | conditionals | definitions | expansions }
```

［詳細説明］

- ソースリストファイルの内容の設定を行います。
- サブオプションと指定内容の一覧を以下に示します。

表 B.5 サブオプション指定一覧

サブオプション	内容
source	C/C++ ソースを出力
conditionals	条件アセンブルで条件が偽となる行も含めて出力
definitions	.DEFINE を置き換える以前の情報を出力
expansions	アセンブラマクロ記述展開行を出力

- 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［リスト］カテゴリの［C/C++ ソースを出力する］，［条件アセンブルで偽の行を出力する］，［.DEFINE 置換前の情報を出力する］，［アセンブラ・マクロ記述展開行を出力する］

［備考］

- 本オプションは listfile オプション指定時のみ有効です。
- コンパイラが出力する情報は，ソースリストに書き込まれます。ソースリストファイルのフォーマットについては，「アセンブリ・リスト・ファイル」を参照ください。

最適化オプション

＜コンパイル・オプション / 最適化オプション＞
最適化オプションには，次のものがあります。

- -optimize
- -goptimize
- -speed
- -size
- -loop
- -inline
- -noinline
- -file_inline (*) 無効オプション
- -case
- -volatile
- -novolatile
- -const_copy
- -noconst_copy
- -const_div
- -noconst_div
- -library
- -scope
- -noscope
- -schedule
- -noschedule
- -map
- -smap
- -nomap
- -approxdiv
- -enable_register (*) 無効オプション
- -simple_float_conv
- -fpu
- -nofpu
- -alias
- -float_order (*) 無効オプション
- -ip_optimize
- -merge_files
- -whole_program

-optimize

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-optimize = { 0 | 1 | 2 | max }
```

- 省略時解釈
- -optimize=2 です。

［詳細説明］

- 最適化レベルを指定します。
- optimize=0 を指定した場合，最適化を実施しません。これにより，デバッグ情報を高い精度で出力でき，ソースレベルデバッグがしやすくなります。
- optimize=1 を指定した場合，自動変数のレジスタ割付，関数出口ブロックの統合，統合可能な複数命令の統合など，一部最適化を実施します。これにより，optimize=0 指定時よりもコードサイズを削減できます。
- optimize=2 を指定した場合，全般的に最適化を実施します。ただし，実施する最適化の内容は，size/speed オプションの選択によって異なります。
- optimize=max を指定した場合，実施可能な最適化を最大限に行います。たとえば，最適化の適用範囲を最大限に拡大したり，speed オプション指定時には，大規模なループ展開を可能にします。最適化の効果が期待できる反面，コンパイル時間の増大や，speed オプション指定時のコードサイズの大幅な増加など，副作用を伴う場合があります。
- 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［最適化］カテゴリの［最適化レベル］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［最適化レベル］

［備考］

- 各種最適化オプションの説明で，デフォルトが記述されていないものは，optimize オプションと speed, size オプションの指定値によりデフォルトが変化することを意味します。
- デフォルトについての詳細は，speed, size オプションを参照ください。

-goptimize

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-goptimize
```

［詳細説明］

- モジュール間最適化時に使用する付加情報を，出力ファイル内部に生成します。
- 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［モジュール間最適化用付加情報を出力する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［モジュール間最適化用付加情報を出力する］

-speed

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-speed
```

［詳細説明］

- speed オプションを指定した場合，実行性能重視の最適化を実施します。
- 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］カテゴリの［最適化方法］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［最適化方法］

［備考］

- speed オプションを指定した場合，optimize オプションの指定により，以下オプションが指定されているとみなします。

＜ optimize=max 指定時＞

	ループ展開	インライン展開	定数除算の乗算化	命令並び換え	const 修飾変数の定数伝播	最適化範囲分割	外部変数アクセス最適化	ポインタ指示先の型を考慮した最適化
speed	loop=8	inline=250	const_div	schedule	const_copy	noscope	map 注 nomap 注	alias=ansi

注　入力が C/C++ ソースで，かつ出力の指定が output=abs か mot の場合は map がデフォルトに，それ以外では nomap がデフォルトとなります。

＜ optimize=2 指定時＞

	ループ展開	インライン展開	定数除算の乗算化	命令並び換え	const 修飾変数の定数伝播	最適化範囲分割	外部変数アクセス最適化	ポインタ指示先の型を考慮した最適化
speed	loop=2	inline=100	const_div	schedule	const_copy	scope	nomap	alias=noansi

＜ optimize=0 または optimize=1 指定時＞

	ループ展開	インライン展開	定数除算の乗算化	命令並び換え	const 修飾変数の定数伝播	最適化範囲分割	外部変数アクセス最適化	ポインタ指示先の型を考慮した最適化
speed	loop=1	noinline	const_div	noschedule	noconst_copy	scope	nomap	alias=noansi

-size

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-size
```

［詳細説明］

- size オプションを指定した場合，コードサイズ重視の最適化を実施します。
- 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］カテゴリの［最適化方法］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［最適化方法］

［備考］

- size オプションを指定した場合，optimize オプションの指定により，以下オプションが指定されているとみなします。ただし，以下オプションを明示的に指定した場合は指定したオプションが有効になります。

＜ optimize=max 指定時＞

	ループ展開	インライン展開	定数除算の乗算化	命令並び換え	const 修飾変数の定数伝播	最適化範囲分割	外部変数アクセス最適化	ポインタ指示先の型を考慮した最適化
size	loop=1	inline=0	noconst_div	schedule	const_copy	noscope	map 注 nomap 注	alias=ansi

注　入力が C/C++ ソースで，かつ出力の指定が output=abs か mot の場合は map がデフォルトに，それ以外では nomap がデフォルトとなります。

＜ optimize=2 指定時＞

	ループ展開	インライン展開	定数除算の乗算化	命令並び換え	const 修飾変数の定数伝播	最適化範囲分割	外部変数アクセス最適化	ポインタ指示先の型を考慮した最適化
size	loop=1	noinline	noconst_div	schedule	const_copy	scope	nomap	alias=noansi

＜ optimize=0 または optimize=1 指定時＞

	ループ展開	インライン展開	定数除算の乗算化	命令並び換え	const 修飾変数の定数伝播	最適化範囲分割	外部変数アクセス最適化	ポインタ指示先の型を考慮した最適化
size	loop=1	noinline	noconst_div	noschedule	noconst_copy	scope	nomap	alias=noansi

-loop

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-loop[=< 数値 >]
```

- 省略時解釈
  loop=2 です。

［詳細説明］

- ループ展開の最適化を行うかどうかを指定します。
- loop オプションを指定した場合，ループ文（for, while, do-while）を展開します。
- < 数値 > で，最大で何倍の展開を行うかを指定することができます。< 数値 > は 1 ～ 32 の整数を指定することができます。< 数値 > を指定しなかった場合は 2 とします。
- 本オプションの省略時解釈は，optimize オプションと speed, size オプションの指定に従います。詳細は，speed, size オプション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［最適化］カテゴリの［ループ展開］，［最大展開数］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［ループ展開］，［最大展開数］

［備考］

- optimize=0 または optimize=1 のときは本オプションの指定は無効です。

-inline

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-inline[=< 数値 >]
```

- 省略時解釈
  inline=100 です。

［詳細説明］

- 関数の自動インライン展開を行います。
- < 数値 > として使える値の範囲は 0 ～ 65535 です。
- inline オプションを指定した場合，自動インライン展開を行います。ただし，#pragma noinline を指定した関数はインライン展開を行いません。
- < 数値 > で，関数サイズが何 % 増加するまでインライン展開を行うかを指定できます。例えば，inline=100 を指定した場合，関数サイズが 100% 増加するまで（2 倍まで）インライン展開を行います。
- 本オプションの省略時解釈は，optimize オプションと speed, size オプションの指定に従います。詳細は，speed, size オプション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［最適化］カテゴリの［自動インライン展開を行う］，［関数サイズの最大増加率］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［自動インライン展開を行う］，［関数サイズの最大増加率］

［備考］

- #pragma inline を指定した関数，および inline 指定子付きの関数は，オプションの指定に関わらず，展開を試みます。
- 確実に関数をインライン展開したい場合は，#pragma inline を関数に指定してください。
- 本オプションの選択もしくは inline 指定子を関数に指定しても，コンパイラで効率が悪くなると判断したときは，インライン展開を行わないことがあります。

-noinline

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-noinline
```

［詳細説明］

- 関数の自動インライン展開を行いません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［自動インライン展開を行う］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［自動インライン展開を行う］

［備考］

- #pragma inline を指定した関数，および inline 指定子付きの関数は，オプションの指定に関わらず，展開を試みます。

-file_inline

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-file_inline=< ファイル名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- V2.00 で、本オプションは無効になりました。指定した場合、無視されますが、従来バージョンとの互換性のためエラーにはなりません。

［備考］

- C(C99) ソースの場合は、代替機能である -merge_files オプションが使用できます。-file_inline に指定していたファイル (-file_inline_path を併用していた場合は、パス名を含めて) を、ソースファイルのひとつとして追加してください。
- -merge_files 機能には、いくつか注意点があります。-merge_files オプションの [備考] 欄も参照してください。

-case

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-case= { ifthen | table | auto }
```

- 省略時解釈
  case=auto です。

［詳細説明］

- switch 文のコード展開方式を指定します。
- case=ifthen を指定した場合，switch 文を if_then 方式で展開します。if_then 方式は，switch 文の評価式の値と case ラベルの値を比較し，一致すれば case ラベルの文へ飛ぶ処理を case ラベルの回数繰り返す展開方式です。この方式は，switch 文に含まれる case ラベルの数に比例してオブジェクトコードのサイズが増大します。
- case=table を指定した場合，switch 文をテーブル方式で展開します。テーブル方式は，case ラベルの飛び先を分岐テーブルに確保し，1 回の分岐テーブルの参照で switch 文の評価式と一致する case ラベルの文へ飛ぶ展開方式です。この方式は，switch 文に含まれる case ラベルの数に比例して分岐テーブルのサイズが増えますが，実行速度は常に一定です。分岐テーブルは，switch 文分岐テーブル領域のセクションに出力されます。
- case=auto を指定した場合，if_then 方式，テーブル方式いずれかをコンパイラが自動的に選択します。
- 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］カテゴリの［switch 文のコード展開方式］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［switch 文のコード展開方式］

［備考］

- case=table 指定時に作成される分岐テーブルは，nostuff オプション指定時は W セクションに出力されますが，nostuff オプション指定がない場合は，switch 文の規模により W，W_2 または W_1 セクションのいずれかに振り分けて出力されます。

-volatile

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-volatile
```

［詳細説明］

- volatile を指定した場合，すべての外部変数を volatile 宣言したものとして扱います。したがって，外部変数のアクセス回数，アクセス順序は C/C++ 言語ソースファイルで記述した通りになります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［外部変数を volatile 化する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［外部変数を volatile 化する］

［備考］

- 本オプションで各変数に付加された volatile 宣言は、RX 用のデバッグツールでは表示されません。

-novolatile

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-novolatile
```

［詳細説明］

- novolatile を指定した場合，volatile 修飾のない外部変数に対して最適化を行います。したがって，外部変数のアクセス回数，アクセス順序が C/C++ 言語ソースファイルで記述した場合と異なることがあります。
- 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［最適化］カテゴリの［外部変数を volatile 化する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［外部変数を volatile 化する］

-const_copy

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-const_copy
```

- 省略時解釈
  optimize=2 または optimize=max オプションを指定した場合は const_copy です。

［詳細説明］

- const_copy を指定した場合，const 修飾型外部変数についても定数伝播を行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［const 宣言された外部変数の定数伝播を実施する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［const 宣言された外部変数の定数伝播を実施する］

［備考］

- C++ 言語ソースファイルの const 修飾型変数については，本オプションで制御することはできません（常に定数伝播されます）。

-noconst_copy

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-noconst_copy
```

- 省略時解釈
  本オプションの省略時解釈は，optimize=1 または optimize=0 オプションを指定した場合は noconst_copy です。

［詳細説明］

- noconst_copy を指定した場合，const 修飾型外部変数の定数伝播を抑止します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［const 宣言された外部変数の定数伝播を実施する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［const 宣言された外部変数の定数伝播を実施する］

［備考］

- C++ 言語ソースファイルの const 修飾型変数については，本オプションで制御することはできません（常に定数伝播されます）。

-const_div

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-const_div
```

- 省略時解釈
  speed オプションを指定した場合は const_div です。

［詳細説明］

const_div を指定した場合，ソースファイル中の整数型定数による除算および剰余算を，乗算やビット演算 (シフトやビット論理積) を用いた命令列に変換します。
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］カテゴリの［整数型定数による除算および剰余算の変換方法］
［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［整数型定数による除算および剰余算の変換方法］

［備考］

シフト演算のみで行える定数乗算，およびビット論理積のみで行える剰余算は，const_div オプションの制御対象外となります。

-noconst_div

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-noconst_div
```

- 省略時解釈
  size オプションを指定した場合は noconst_div です。

［詳細説明］

- noconst_div を指定した場合，ソースファイル中の整数型定数による除算および剰余算に、それぞれに対応する除算命令および剰余算命令 (2 のべき乗定数値による符号なし整数の除算および剰余算を除く) を用います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［整数型定数による除算および剰余算の変換方法］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［整数型定数による除算および剰余算の変換方法］

［備考］

- シフト演算のみで行える定数乗算，およびビット論理積のみで行える剰余算は，noconst_div オプションの制御対象外となります。

-library

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-library = { function | intrinsic }
```

- 省略時解釈
  library=intrinsic です。

［詳細説明］

- library=function を指定した場合，ライブラリ関数をすべて関数呼び出しします。
- library=intrinsic を指定した場合，abs()，fabsf() およびストリング操作命令が使用できるライブラリ関数を命令展開します。
- 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］カテゴリの［ライブラリ関数の展開方法］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［ライブラリ関数の展開方法］

［備考］

- library=intrinsic とともに -isa=rxv2 が指定されている場合は、sqrtf 関数または -dbl_size=4 指定時の sqrt 関数のそれぞれの呼び出しを FSQRT 命令に置き換えます。ただし、FSQRT 命令に展開された場合は、変数 errno の内容を変更しません。

-scope

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-scope
```

- 省略時解釈
  optimize=max オプションを指定した場合以外は scope です。

［詳細説明］

- scope を指定した場合，サイズの大きい関数について，最適化範囲を複数に分割してコンパイルします。
- 本オプションは，プログラムによって実行性能に影響しますので，性能チューニング時に試してください。
- 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［最適化］カテゴリの［最適化範囲を複数に分割してコンパイルする］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［最適化範囲を複数に分割してコンパイルする］

-noscope

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-noscope
```

- 省略時解釈
  optimize=max オプションを指定した場合は noscope です。

［詳細説明］

- noscope を指定した場合，最適化範囲を分割せずにコンパイルします。最適化範囲が広がることによりコンパイル速度は遅くなりますが，一般的にはオブジェクト性能が向上します。ただし，レジスタ数が不足するとオブジェクト性能が低下する場合があります。
- 本オプションは，プログラムによって実行性能に影響しますので，性能チューニング時に試してください。
- 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［最適化］カテゴリの［最適化範囲を複数に分割してコンパイルする］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［最適化範囲を複数に分割してコンパイルする］

-schedule

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-schedule
```

- 省略時解釈
  optimize=2 または optimize=max オプションを指定した場合は schedule です。

［詳細説明］

- schedule を指定した場合，パイプライン処理を考慮した命令並べ替えを行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［パイプライン処理を考慮した命令並べ替えを行う］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［パイプライン処理を考慮した命令並べ替えを行う］

-noschedule

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-noschedule
```

- 省略時解釈
  optimize=1 または optimize=0 オプションを指定した場合は noschedule です。

［詳細説明］

- noschedule を指定した場合，命令並べ替えを行いません。基本的に C/C++ 言語ソースファイルで記述した順番で処理を行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［パイプライン処理を考慮した命令並べ替えを行う］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［パイプライン処理を考慮した命令並べ替えを行う］

-map

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-map[=< ファイル名 >]
```

- 省略時解釈
  optimize=max オプションを指定した場合は map です。
- ［詳細説明］
- 外部変数アクセス最適化を行います。
- 最適化リンケージエディタが生成する外部シンボル割り付け情報を元にベースアドレスを設定し，外部変数もしくは静的変数のアクセスをベースアドレス相対で行うコードを生成します。
- map オプションによる外部変数アクセス最適化を使用する場合は，output オプションの指定により使い方が異なります。
- [output=abs または mot の場合]
  map のみ指定してください（optimize=max 指定時は不要）。自動的にコンパイル・リンクを 2 回行い，外部シンボル割り付け情報を元にベースアドレスを設定したコード生成を行います。
- [output=obj の場合]
  ソースファイルを本オプションを指定しないで一度コンパイルし，最適化リンケージエディタでのリンク時に map=< ファイル名 > を指定して外部シンボル割り付け情報ファイルを作成し，再度 ccrx に map=< ファイル名 > を指定してコンパイルしてください。
- 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［最適化］カテゴリの［外部変数アクセス最適化を行う］

［例］

- ＜C ソース＞

```
long A,B,C;
void func()
{
    A = 1;
    B = 2;
    C = 3;
}
```

- ＜出力コード＞

```
_func:
    MOV.L       #_A,R4      ; A のアドレスをベースアドレスに設定
    MOV.L       #1,[R4]
    MOV.L       #2,4[R4]    ; A のアドレスをベースとして，B にアクセス
    MOV.L       #3,8[R4]    ; A のアドレスをベースとして，C にアクセス
```

［備考］

- 外部変数もしくは静的変数の定義順を変更した場合は，外部シンボルアドレス情報ファイルを生成し直す必要があります。map オプション以外で 1 回目のコンパイル時に指定したオプションと異なるオプションを指定した場合と，関数内の処理を追加した場合は，動作は保証しません。これらの場合は必ず外部シンボルアドレス情報ファイルを生成し直してください。
- C/C++ ソースをコンパイルする場合のみ適用されます。コンパイル時に output=src を用いて作成，またはアセンブリ言語で記述されたプログラムには適用されません。

- map オプションと smap オプションを同時に指定した場合は，map オプションが有効となります。
- プログラムセクションの次に連続してデータセクションを配置すると，外部変数アクセス最適化が無効になる，あるいは，最適化が十分に機能しない場合があります。
- 最適化を最大限に機能させるためには，連続して複数のセクションを配置させる場合，プログラムセクションを末尾に配置してください。
- 以下に具体例を示します。

- P を 0x100 番地から，C1,C2 を P の直後，C3 を 0x400 番地から配置したとします。
- この場合，P セクションと連続して C1,C2 セクションが配置されているため，C2 の後方に配置してください。C3 セクションは配置関係が連続していないため無関係です。

-smap

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-smap
```

［詳細説明］

- コンパイル対象ファイル内で定義された外部変数もしくは静的変数についてベースアドレスを設定し，アクセスをベースアドレス相対で行うコードを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［外部変数アクセス最適化を行う］

［例］

- ＜C ソース＞

```
long A,B,C;
void func()
{
     A = 1;
     B = 2;
     C = 3;
}
```

- ＜出力コード＞

```
_func:
    MOV.L       #_A,R4      ; A のアドレスをベースアドレスに設定
    MOV.L       #1,[R4]
    MOV.L       #2,4[R4]    ; A のアドレスをベースとして，B にアクセス
    MOV.L       #3,8[R4]    ; A のアドレスをベースとして，C にアクセス
```

［備考］

- C/C++ ソースをコンパイルする場合のみ適用されます。コンパイル時に output=src を用いて作成，またはアセンブリ言語で記述されたプログラムには適用されません。
- map オプションと smap オプションを同時に指定した場合は，map オプションが有効となります。

-nomap

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-nomap
```

- 省略時解釈
  optimize=0，optimize=1，または optimize=2 オプションを指定した場合は nomap です。

［詳細説明］

- nomap オプションを指定した場合，外部変数アクセス最適化を行いません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［外部変数アクセス最適化を行う］

［例］

- ＜C ソース＞

```
long A,B,C;
void func()
{
    A = 1;
    B = 2;
    C = 3;
}
```

- ＜出力コード＞

```
_func:
   MOV.L       #_A,R4
   MOV.L       #1,[R4]
   MOV.L       #_B,R4
   MOV.L       #2,[R4]
   MOV.L       #_C,R4
   MOV.L       #3,[R4]
```

-approxdiv

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-approxdiv
```

- 省略時解釈
  浮動小数点定数除算を，定数の逆数の乗算に変換しません。

［詳細説明］

- 浮動小数点定数除算を，定数の逆数の乗算に変換します。
- 変数 ÷ 除数 という式があるとき，除数が定数の場合は，変数 ×(除数の逆数) に変換したコードを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［浮動小数点定数除算の乗算化を行う］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［浮動小数点定数除算の乗算化を行う］

［備考］

- 本オプションを指定した場合，浮動小数点定数除算の実行速度は向上しますが，演算の精度と演算の順序が変わる場合がありますので注意が必要です。

-enable_register

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-enable_register
```

［詳細説明］

- V2.00 で、本オプションは無効になりました。指定した場合、無視されますが、従来バージョンとの互換性のためエラーにはなりません。

-simple_float_conv

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-simple_float_conv
```

［詳細説明］

- 浮動小数点型の型変換処理の一部を省略します。
- 本オプション選択時は，次の浮動小数点の型変換を行う生成コードが変化します。
  a) 32bit 浮動小数点型から符号無し整数型への変換
  b) 符号無し整数型から 32bit 浮動小数点型への変換
  c) 32bit 浮動小数点型を経由した，整数型から 64bit 浮動小数点型への変換
- 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］カテゴリの［浮動小数点型 <-> 符号無し整数型の範囲チェックを省略する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［浮動小数点型 <-> 符号無し整数型の範囲チェックを省略する］

［例］

＜ a) 32bit 浮動小数点型から符号無し整数型への変換＞
-fpu の指定が有効でない場合は該当しません。

```
unsigned long func1(float f)
{
    return ((unsigned long)f);
}

オプション非指定時：
 _func1:
         FCMP    #4F000000H,R1
         BLT     L12
         FADD    #0CF800000H,R1
 L12:
         FTOI    R1,R1
         RTS

オプション指定時：
 _func1:
         FTOI    R1,R1
         RTS
```

＜ b) 符号無し整数型から 32bit 浮動小数点型への変換＞
-fpu の指定が有効でない場合は該当しません。

```
float func2(unsigned long u)
{
       return ((float)u);
}

オプション非指定時：
 _func2:
         BTST   #31,R1
         BEQ    L15
         SHLR   #1,R1,R14
         AND    #1,R1
         OR     R14,R1
         ITOF   R1,R1
         FADD   R1,R1
         BRA    L16
 L15:
         ITOF   R1,R1
 L16:
         RTS

オプション指定時：
 _func2:
         ITOF   R1,R1
         RTS
```

＜ c) 32bit 浮動小数点型を経由した，整数型から 64bit 浮動小数点型への変換＞
-dbl_size=8 の指定が有効でない場合は該当しません。

```
 double func3(long l)
 {
       return (double)(float)l;
 }
オプション非指定時：
_func3:
        ITOF    R1,R1
        BRA     __COM_CONVfd

オプション指定時：
        BRA     __COM_CONV32sd
```

［備考］

- 本オプション指定時，該当する型変換の処理に対するコード性能は向上しますが，変換結果が C,C++ 言語規格と異なる場合がありますので，ご注意ください。
- -optimize=0 のとき c) は無効になります。

-fpu

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-fpu
```

- 省略時解釈
  ISA（*1）で命令セットアーキテクチャを選択した場合は fpu です。
  CPU に RX200 を選択（*1）した場合は nofpu，それ以外は fpu です。

注 *1）isa オプションまたは環境変数 ISA_RX による選択を指します。
*2) cpu オプションまたは環境変数 CPU_RX による選択を指します。

［詳細説明］

- fpu を指定した場合，FPU 命令を使用したコード生成を行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  ［共通オプション］タブの［最適化］カテゴリの［浮動小数点演算命令を使用する］

［備考］

- FPU 命令の具体的な内容については，RX コーディング編の「インストラクション」を参照してください。
- CPU として RX200 が選択されている場合，fpu を指定するとエラーになります。

-nofpu

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-nofpu
```

- 省略時解釈
  ISA（*1）で命令セットアーキテクチャを選択した場合は fpu です。
  CPU に RX200 を選択（*1）した場合は nofpu，それ以外は fpu です。

注 *1）isa オプションまたは環境変数 ISA_RX による選択を指します。
*2) cpu オプションまたは環境変数 CPU_RX による選択を指します。

［詳細説明］

- nofpu を指定した場合，FPU 命令を使用しないコード生成を行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
  ［共通オプション］タブの［最適化］カテゴリの［浮動小数点演算命令を使用する］

-alias

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-alias = { noansi | ansi }
```

- 省略時解釈
  alias=noansi です。
- ［詳細説明］
- ポインタ指示先の型を考慮した最適化を実施するかどうかを選択します。
- alias=ansi を指定した場合，ANSI 規格に基づき，ポインタ指示先の型を考慮した最適化を行います。一般には，alias=noansi を指定した場合よりもオブジェクト性能が向上しますが，alias=ansi と alias=noansi とで実行結果が異なる場合があります。
- alias=noansi を指定した場合は V.1.00 と同様で，ANSI 規格に基づくポインタ指示先の型を考慮した最適化を行いません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［最適化］カテゴリの［ポインタ指示先の型を考慮した最適化を実施する］
- ［ライブラリ・ジェネレート・オプション］タブの［最適化］カテゴリの［ポインタ指示先の型を考慮した最適化を実施する］

［例］

```
long x;
long n;
void func(short * ps)
{
    n = 1;
    *ps = 2;
    x = n;
}
```

- ＜alias=noansi 指定時＞
  *ps = 2; によって，n の値が書き換わる可能性があるとみなし（A）で n の値を再ロードします。

```
_func:
     MOV.L      #_n,R4
     MOV.L      #1,[R4]     ;  n = 1;
     MOV.W      #2,[R1]     ;  *ps = 2;
     MOV.L      [R4],R5     ;  （A）n を再ロードする
     MOV.L      #_x,R4
     MOV.L      R5,[R4]
     RTS
```

- ＜alias=ansi 指定時＞
  *ps と n は型が異なるため，*ps = 2; では n の値は変化しないと判断し，（B）で，n = 1 で代入に使用した値を再利用します（もし *ps = 2; によって n の値が書き換わる場合，結果は変わります）。

```
_func:
    MOV.L       #_n,R4
    MOV.L       #1,[R4]     ;  n = 1;
    MOV.W       #2,[R1]     ;  *ps = 2;
    MOV.L       #_x,R4
    MOV.L       #1,[R4]     ;  (B) n = 1 で代入に使用した値を再利用する
    RTS
```

[備考]

- optimize=0 または optimize=1 が有効な場合に alias オプションを選択すると，alias=ansi の選択は無視され，常に alias=noansi が選択されたものとしてコード生成します。

-float_order

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-float_order
```

- 省略時解釈
  浮動小数点演算式の演算順序変更の最適化を行いません。

［詳細説明］

- V2.00 で、本オプションは無効になりました。指定した場合、無視されますが、従来バージョンとの互換性のため
  エラーにはなりません。

-ip_optimize

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-ip_optimize
```

［詳細説明］

- 大域最適化を実施します。
- 手続き間別名解析を利用した最適化
- 引数・戻り値の定数伝播 など

［例］

例 1.

- ＜C ソース＞

```
static int func1(int *a, int *b) {
    *a=0;
    *b=1;
    return *a;
}
int x[2];
int func2() {
    return func1(x, x+1);
}
```

- <ip_optimize の指定がない場合のアセンブリ出力 >

```
; -optimize=2 -size
__$func1:
  MOV.L #00000000H, [R1]
  MOV.L #00000001H, [R2]
  MOV.L [R1], R1
  RTS

_func2:
  MOV.L #_x, R1
  ADD #04H, R1, R2
  BRA __$func1
```

- <ip_optimize 指定時のアセンブリ出力 >

```
; -optimize=2 -size

__$func1:
  MOV.L #00000000H, [R1]
  MOV.L #00000001H, [R2]
  MOV.L #00000000H, R1
  RTS

_func2:
  MOV.L #_x, R1
  ADD #04H, R1, R2
  BRA __$func1
```

例 2.

- ＜C ソース＞

```
static int func(int x, int y, int z) {
    return x-y+z;
}
int func2() {
    return func(3,4,5);
}
```

- <ip_optimize の指定がない場合のアセンブリ出力 >

```
; -optimize=2 -size

__$func:
  ADD R3, R1
  SUB R2, R1
  RTS
_func2:
  MOV.L #00000005H, R3
  MOV.L #00000004H, R2
  MOV.L #00000003H, R1
  BRA __$func
```

- <ip_optimize 指定時のアセンブリ出力 >

```
; -optimize=2 -size

__$func:
  MOV.L #00000004H, R1
  RTS
_func2:
  MOV.L #00000005H, R3
  MOV.L #00000004H, R2
  MOV.L #00000003H, R1
  BRA __$func
```

［備考］

- merge_files オプションと同時に指定することにより、ファイル間の最適化も行うことができます。

-merge_files

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-merge_files
```

- ［詳細説明］
- 複数の C ソースファイルをマージしてコンパイルすることにより、1 つのオブジェクトファイルを出力します。
- output オプションでファイル名を指定した場合は指定した名前のファイルを生成し、指定しなかった場合は先頭に入力したソースファイル名に出力形式に応じた拡張子をつけたファイルを生成します。
- output オプションで出力形式に src または obj を指定した場合は、他の入力したソースファイル名に出力形式に応じた拡張子をつけた空ファイルを生成します。

［例］

```
ccrx -merge_files -output=src=files.obj file1.c file2.c file3.c
```

files.obj にオブジェクトを生成します。また、空ファイル file1.obj、file2.obj、file3.obj を生成します。

［備考］

- コンパイル対象ファイルが 1 つの場合は、本オプションは無効になります。
- output オプションで出力形式に prep を指定した場合、本オプションは無効になります。
- 本オプションと inline オプションを同時に指定した場合、ファイル間インライン展開も行うことができます。
- C++ または EC++ でコンパイルするコンパイル対象ファイルに対しては、本オプションは無効になります。
- プログラムに static 変数、static 関数を含む場合、次の制限があります。
- デバッガのウォッチウィンドウで、ファイル間で同じ名前の static 変数を表示する場合は、変数名だけでなくファイル名も指定してください。ファイル名がないと変数が特定できず表示内容が不定となります。
- ファイル間で同じ名前の static 関数があり、関数が属すセクションを rlink でオーバレイ化した場合、デバッガのオーバレイセクションの優先表示機能は使用できません。
- リンクマップファイル (.map) には、static 変数と static 関数は、元の名前では表示されず、代わりにコンパイラで変換した名前が表示されます。
- ソースファイル間で、同じ変数の宣言内容 (型指定子など) が異なる場合、コンパイル時にエラーとなる場合があります。
- 本オプションを指定して生成したオブジェクト・ファイルをリンクする際にリンク・オプション -delete, -rename, -replace のいずれかを指定した場合，動作は保証しません。

-whole_program

＜コンパイル・オプション / 最適化オプション＞

［指定形式］

```
-whole_program
```

［詳細説明］

- コンパイル対象ファイルがプログラム全体であることを仮定し、コンパイル対象ファイル全てをマージして大域最適化を実施します。

［備考］

- 本オプションを指定した場合は、C++ 言語ソースファイルを入力ファイルに含めないでください。
- 本オプションを指定した場合は、-lang=cpp および -lang=ecpp を指定しないでください。
- 本オプションを指定した場合、ip_optimize オプションが有効になります。さらに複数のソースファイルを入力した場合は、merge_files オプションが有効になります。
- 本オプションを指定した場合、コンパイラは以下の条件を満たすことを前提にコンパイルを行います。条件を満たさない場合の動作は保証しません。
  - コンパイル対象ファイル内で定義された extern 変数の内容及びアドレスが、コンパイル対象ファイル以外で設定及び参照されることはない
  - コンパイル対象ファイル内からコンパイル対象ファイル以外の関数を呼び出したとしても、呼び出された関数からコンパイル対象ファイル内の関数が呼ばれることはない

［例］

```
[wp.c]
extern void g(void);
int func(void)
{
    static int a = 0;
    a++;      // (1) a に値を書き込む。
    g();      // (2) g() を呼び出す。
    return a; // (3) a を読み出す。
}
```

[whole_program 指定なし]
関数 g() は関数 func() を呼び出す可能性があるので、(2) の実行で変数 a は書き換わるとみなし、(3) では a の値を読み出すコードを生成します。

```
_func:
        PUSH.L R6
        MOV.L  #__$a$1,R6
        MOV.L  [R6],R14
        ADD    #1,R14
        MOV.L  R14,[R6]     ; (1)
        BSR    _g           ; (2)
        MOV.L  [R6],R1      ; (3)
        RTSD   #4,R6-R6
```

[whole_program 指定あり]
関数 g() は関数 func() 呼び出さないとみなすため、(2) の実行で変数 a は不変と判断します。これを利用し、(3) では a の値を読み出さず、代わりに (1) で a に書き込んだ値を使用するコードを生成します。

```
_func:
        PUSH.L R6
        MOV.L  #__$a$1,R14
        MOV.L  [R14],R6
        ADD    #1,R6
        MOV.L  R6,[R14]     ; (1)
        BSR    _g           ; (2)
        MOV.L  R6,R1        ; (3)
        RTSD   #4,R6-R6
```

マイコンオプション

＜コンパイル・オプション / マイコンオプション＞

マイコンオプションには，次のものがあります。

- -isa
- -cpu
- -endian
- -round
- -denormalize
- -dbl_size
- -int_to_short
- -signed_char
- -unsigned_char
- -signed_bitfield
- -unsigned_bitfield
- -auto_enum
- -bit_order
- -pack
- -unpack
- -exception
- -noexception
- -rtti
- -fint_register
- -branch
- -base
- -patch
- -pic
- -pid
- -nouse_pid_register
- -save_acc

-isa

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-isa={ rxv1 | rxv2 }
```

- 省略時解釈
  環境変数 ISA_RX の内容に従います。

［詳細説明］

- 生成する命令コードの命令セットアーキテクチャ (RXv1 または RXv2) を指定します。
- isa=rxv1 を指定した場合，RXv1 を選択します。
- isa=rxv2 を指定した場合，RXv2 を選択します。
- 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［CPU］カテゴリの［命令セット・アーキテクチャ］

［備考］

- -nofpu と -fpu のいずれの選択もない場合に -isa オプションを指定すると、-fpu を自動的に選択します。
- -isa オプションを省略した場合、-cpu オプション、環境変数 CPU_RX および 環境変数 ISA_RX のいずれもない場合は、エラーとなります。
- -cpu オプションと同時に指定することはできません。

-cpu

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-cpu={ rx600 | rx200 }
```

- 省略時解釈
  環境変数 CPU_RX に従います。

［詳細説明］

- マイコン種別を指定します。
- cpu=rx600 を指定した場合，RX600 向けの命令コードを生成します。
- cpu=rx200 を指定した場合，RX200 向けの命令コードを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［命令セット・アーキテクチャ］

［備考］

- 本オプションは、従来機能と互換性を保つためのものです。
- 今後、RX ファミリの製品展開で追加される品種については、命令セットアーキテクチャなどの選択は、-cpu オプションではなく、-isa オプションでの対応となります。新規にアプリケーション開発する際は、-isa オプションを指定してください。
- -cpu オプションは、次の記述により -isa オプションと -fpu,-nofpu オプションで置き換えることが可能です。*1
- -cpu=rx600 → -isa=rxv1 -fpu
- -cpu=rx200 → -isa=rxv1 -nofpu
- -cpu=rx200 を指定すると，-nofpu が自動的に選択されます。
- -cpu=rx200 と fpu は同時に指定することはできません。
- -nofpu と fpu のいずれの選択もない場合に -cpu=rx600 を指定すると，-fpu を有効にします。
- - cpu オプションを省略し、-isa オプション、環境変数 CPU_RX および 環境変数 ISA_RX のいずれも指定していない場合はエラーとなります。
- -isa オプションと同時に指定することはできません。

【注意】

- *1) ソースプログラムにプリデファインドマクロ __RX200、__RX600 の記述がある場合を除きます。

-endian

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-endian={ big  |  little }
```

- 省略時解釈
  endian=little です。

［詳細説明］

- endian=big を指定した場合，データのバイト並びが big endian になります。
- endian=little を指定した場合，データのバイト並びが little endian になります。
- #pragma endian 拡張子でも指定できます。オプションと #pragma 拡張子の両方が指定された場合には，#pragma 拡張子の指定を優先します。
- 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［CPU］カテゴリの［データのエンディアン］

［例］

-

［備考］

-

-round

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-round={ zero  |  nearest }
```

- 省略時解釈
  round=nearest です。

［詳細説明］

- 浮動小数点定数演算の丸め方式を選択します。
- round=zero を指定した場合，round to zero で丸めます。
- round=nearest を指定した場合，round to nearest で丸めます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［浮動小数点定数演算の丸め方式］

［例］

-

［備考］

- 本オプションでは，実行時の浮動小数点演算における丸め方式を変更することはできません。
- 本オプションのデフォルトの選択は，fpu, nofpu オプション選択の影響を受けません。

-denormalize

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-denormalize={ off  |  on }
```

- 省略時解釈
  denormalize=off です。

［詳細説明］

- 浮動小数点定数に非正規化数を記述した場合の扱いを指定します。
- denormalize=off を指定した場合，非正規化数を 0 として扱います。
- denormalize=on を指定した場合，非正規化数を非正規化数として扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［浮動小数点定数に非正規化数を記述した場合の扱い］

［例］

-

［備考］

- 本オプションでは，実行時の浮動小数点演算における非正規化数の扱いを変更することはできません。
- 本オプションは，fpu, nofpu オプション選択で自動的に有効になることはありません。

-dbl_size

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-dbl_size={4 | 8}
```

- 省略時解釈
  dbl_size=4 です。
- ［詳細説明］
- double 型，および long double 型の精度を指定します。
- dbl_size=4 を指定した場合，単精度浮動小数点型（4 バイト）として扱います。
- dbl_size=8 を指定した場合，倍精度浮動小数点型（8 バイト）として扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［double 型、および long double 型の精度］

［例］

-

［備考］

- dbl_size=4 を選択した場合，標準関数のうち mathf.h と math.h とで同じ仕様の関数（例 : sqrtf と sqrt など）を一体化して標準ライブラリが構築されます。このため dbl_size=4 選択時は，例えば mathf.h ヘッダの関数である sqrtf を呼び出すところを，RX のシミュレータやエミュレータでトレース（ステップ実行）すると，sqrtf ではなく同じ仕様を持つ math.h ヘッダの関数 sqrt が呼び出されたように見えることがあります。

-int_to_short

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-int_to_short
```

- 省略時解釈
  ソースファイル内の int を short に，unsigned int を unsigned short に置換せずコンパイルを行います。

［詳細説明］

- ソースファイル内の int を short に，unsigned int を unsigned short に置換してコンパイルを行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［int 型を short 型に置換する］

［例］

-

［備考］

- limits.h の INT_MAX，INT_MIN，および UINT_MAX は本オプションの変換対象外となります。
- C++ および EC++ コンパイル時は，本オプションは無効になります。C++，EC++ プログラム内から C プログラムを参照する可能性がある外部名に対して W0523041 を出力します。
- C 標準ヘッダをインクルードしたファイルを int_to_short オプションを指定して C++ または EC++ コンパイルした場合も，W0523041 が出力されることがあります。この場合は動作には問題ありませんので無視してください。
- C と C++（EC++）との間で共通にアクセスするデータは，int 型ではなく long 型または short 型で宣言してください。
- 本オプション有効時は、標準ライブラリの scanf 等の書式付き入力関数の呼び出しでは、%d および %u 変換で用いる引数には、必ず long および unsigned long 型変数のアドレスをそれぞれ渡してください。long を含まない int 型や unsigned 型変数のアドレスを渡すと、誤動作する場合があります。

-signed_char

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-signed_char
```

- 省略時解釈
  char 型を unsigned char 型として扱います。

［詳細説明］

- signed char 型として扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［char 型の符号］

［例］

-

［備考］

- char 型のビットフィールドメンバは，本オプションの制御対象外です。signed_bitfield および unsigned_bitfield オプションで制御してください。

-unsigned_char

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-unsigned_char
```

- 省略時解釈
  char 型を unsigned char 型として扱います。

［詳細説明］

- unsigned_char を指定した場合，unsigned char 型として扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［char 型の符号］

［例］

-

［備考］

- char 型のビットフィールドメンバは，本オプションの制御対象外です。signed_bitfield および unsigned_bitfield オプションで制御してください。

-signed_bitfield

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-signed_bitfield
```

- 省略時解釈
  符号なし型として扱います。

［詳細説明］

- 符号付き型として扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［ビットフィールド型の符号］

［例］

-

［備考］

-

-unsigned_bitfield

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-unsigned_bitfield
```

- 省略時解釈
  符号なし型として扱います。

［詳細説明］

- 符号なし型として扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［ビットフィールド型の符号］

［例］

-

［備考］

-

-auto_enum

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-auto_enum
```

- 省略時解釈
  列挙型サイズを signed long 型として処理します。

［詳細説明］

- enum 宣言した列挙型のデータを，列挙値が収まる最小型として処理します。
- 列挙型のとりうる値と型の関係を以下に示します。

表 B.6　列挙型のとりうる値と型の関係

列挙子		選択される型
最小値	最大値	
-128	127	signed char
0	255	unsigned char
-32768	32767	signed short
0	65535	unsigned short
上記以外		signed long

- 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［CPU］カテゴリの［列挙型のサイズを自動選択する］

［例］

-

［備考］

-

-bit_order

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-bit_order = { left | right }
```

- 省略時解釈
  bit_order=right です。

［詳細説明］

- ビットフィールドのメンバの並び順を指定します。
- bit_order=left を指定した場合は上位ビットからメンバを割り付けます。
- bit_order=right を指定した場合は下位ビットからメンバを割り付けます。
- #pragma bit_order 拡張子でも指定できます。オプションと #pragma の両方が指定された場合には，拡張子の指定を優先します。
- 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［CPU］カテゴリの［ビットフィールドメンバの並び順］

-pack

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-pack
```

- 省略時解釈
  構造体，クラスのアライメント数は，メンバの最大のアライメント数と同じになります。

［詳細説明］

- 構造体メンバ，クラスメンバのアライメント数を指定します。
- 構造体メンバのアライメント数は，#pragma pack 拡張子でも指定できます。オプションと #pragma の両方が指定された場合には，#pragma 拡張子の指定を優先します。構造体，クラスのアライメント数は，メンバの最大のアライメント数と同じになります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［構造体メンバのアライメントを 1 とする］

［例］

-

［備考］

- 本オプション指定時の構造体メンバのアライメント数を以下に示します。

表 B.7 **pack** オプション指定時の構造体メンバ，クラスメンバのアライメント数

メンバの型	pack	指定なし
(signed) char	1	1
(unsigned) short	1	2
(unsigned) int 注 , (unsigned) long, (unsigned) long long, 浮動小数点型 , ポインタ型	1	4

注 int_to_short オプションを指定した場合は，short と同じになります。

-unpack

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-unpack
```

- 省略時解釈
  構造体，クラスのアライメント数は，メンバの最大のアライメント数と同じになります。

［詳細説明］

- 構造体メンバ，クラスメンバのアライメント数を指定します。
- 構造体，クラスのアライメント数は，メンバの最大のアライメント数と同じになります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［構造体メンバのアライメントを 1 とする］

［例］

-

［備考］

- 本オプション指定時の構造体メンバのアライメント数を以下に示します。

表 B.8 un**pack** オプション指定時の構造体メンバ，クラスメンバのアライメント数

メンバの型	unpack	指定なし
(signed) char	1	1
(unsigned) short	2	2
(unsigned) int 注 , (unsigned) long, (unsigned) long long, 浮動小数点型 , ポインタ型	4	4

注 int_to_short オプションを指定した場合は，short と同じになります。

-exception

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-exception
```

- 省略時解釈
  C++ 例外処理機能（try, catch, throw）を無効にします。

［詳細説明］

- C++ 例外処理機能（try, catch, throw）を有効にします。
- コード性能が低下する可能性があります。
- 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［CPU］カテゴリの［C++ 例外処理機能 (try, catch, throw) を有効にする］

［例］

-

- ［備考］
- ファイル間で例外処理機能を有効にするには以下を行ってください。
- rtti=on を指定する。
- 最適化リンケージエディタで noprelink オプションを指定しない。
- exception オプションは C++ コンパイル時にのみ指定できます。lang=cpp の指定がなく，かつ入力ファイルの拡張子が .c または .p の場合，exception オプションを指定しても無視されます。

-noexception

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-noexception
```

- 省略時解釈
  C++ 例外処理機能（try, catch, throw）を無効にします。

［詳細説明］

- C++ 例外処理機能（try, catch, throw）を無効にします。
- 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［CPU］カテゴリの［C++ 例外処理機能 (try, catch, throw) を有効にする］

［例］

-

［備考］

- ファイル間で例外処理機能を有効にするには以下を行ってください。
- rtti=on を指定する。
- 最適化リンケージエディタで noprelink オプションを指定しない。
- noexception オプションは C++ コンパイル時にのみ指定できます。lang=cpp の指定がなく，かつ入力ファイルの拡張子が .c または .p の場合，noexception オプションは指定できません。指定するとエラーとなります。

-rtti

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-rtti={ on | off }
```

- 省略時解釈
  rtti=off です。

［詳細説明］

- 実行時型情報の有効 / 無効を指定します。
- rtti=on を指定した場合，dynamic_cast，typeid を有効にします。
- rtti=off を指定した場合，dynamic_cast，typeid を無効にします。
- 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［CPU］カテゴリの［C++ 実行時型情報 (dynamic_cast, typeid) を有効にする］

［例］

-

［備考］

- 本オプションを指定して作成したリロケータブルファイル（.obj）をライブラリに登録したり，最適化リンケージエディタでリロケータブル形式（.rel）で出力しないでください。シンボルの二重定義エラーや未定義エラーになることがあります。
- rtti=on は，C++ コンパイル時にのみ指定できます。lang=cpp の指定がなく，かつ入力ファイルの拡張子が .c または .p の場合，rtti=on を指定しても無視されます。

-fint_register

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-fint_register = {0 | 1 | 2 | 3 | 4 }
```

- 省略時解釈
  fint_register=0 です。

［詳細説明］

- 高速割り込み関数（#pragma interrupt で割り込み仕様に高速割り込み指定（fint）のある関数）でのみ使用する汎用レジスタを指定します。高速割り込み関数以外では，指定されたレジスタは使用しません。本オプションで指定した汎用レジスタは，高速割り込み関数内では退避回復なしで使用できるため，高速割り込み関数の高速化が見込めます。反面，他の関数で使用可能な汎用レジスタが減るため，プログラム全体のレジスタ割付効率は低下します。
- オプションとレジスタの関係を以下に示します。

表 B.9 オプションとレジスタの関係

オプション	高速割り込み専用レジスタ
fint_register=0	なし
fint_register=1	R13
fint_register=2	R12, R13
fint_register=3	R11, R12, R13
fint_register=4	R10, R11, R12, R13

- 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［CPU］カテゴリの［高速割り込み関数でのみ使用する汎用レジスタ］

［例］

-

［備考］

- 高速割り込み関数以外で，本オプションで指定したレジスタを使用した場合の動作は保証しません。本オプションの指定の対象となるレジスタが，base オプションで指定されていた場合，エラーとなります。

-branch

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-branch = { 16 | 24 | 32 }
```

- 省略時解釈
  branch=24 です。

［詳細説明］

- 分岐幅を指定します。
- branch =16 を指定した場合，分岐幅が 16bit 以内であるとしてコンパイルします。
- branch =24 を指定した場合，分岐幅が 24bit 以内であるとしてコンパイルします。
- branch =32 を指定した場合，分岐幅を指定しません。
- 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［CPU］カテゴリの［分岐幅］

-base

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-base = {  rom=< レジスタ >
         | ram=< レジスタ >
         | < アドレス値 > = < レジスタ >}
                          < レジスタ >:= {R8 ～ R13}
```

- ［詳細説明］
- プログラム全体で，ベースアドレスとして固定で使用する汎用レジスタを指定します。
- base=rom=< レジスタ A> を指定した場合は，const 変数のアクセスを指定したレジスタ A 相対で行うことを試みます。ただし，定数領域セクション全体の大きさが 64KB ～ 256KB*1 以内でなければなりません。
- base=ram=< レジスタ B> を指定した場合は，初期化変数および未初期化変数のアクセスは指定したレジスタ B 相対で行います。ただし，RAM データ全体の大きさが 64KB ～ 256KB*1 以内でなければなりません。
- < アドレス値 >=< レジスタ C> を指定した場合は，コンパイル時に割り付けアドレスが確定している領域のうち，アドレス値から 64KB ～ 256KB*1 以内の領域のアクセスを，指定したレジスタ C 相対で行います。

注 *1 この値は，アクセスする変数のサイズにより，64KB から 256KB の間で変化します。

- 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［CPU］カテゴリの［ROM 用ベースレジスタ］，［RAM 用ベースレジスタ］，［アドレス値を設定するベースレジスタのアドレス］，［アドレス値を設定するベースレジスタのレジスタ］

［例］

-

［備考］

- 異なる領域に対して同じレジスタを指定することはできません。
- レジスタはそれぞれの領域で 1 個だけ指定可能です。fint_register オプションで指定したレジスタを本オプションで指定した場合，エラーとなります。
- pid オプション選択時は，base=rom=< レジスタ > を選択できません。選択すると，警告として W0523039 メッセージを表示して base=rom=< レジスタ > の選択を無効とします。

-patch

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-patch ={rx610 }
```

［詳細説明］

- CPU の品種ごとに特有の問題を回避します。
- -patch=rx610 を指定すると，RX610 グループで問題となる，MVTIPL 命令を生成コードに使用しません。
- -patch=rx610 を指定しなければ，組み込み関数 set_ipl の呼び出しに対する生成コードは，MVTIPL 命令を含んだものとなります。
- 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［CPU］カテゴリの［CPU タイプ特有の問題を回避する］

-pic

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-pic
```

- 省略時解釈
  プログラムセクションを PIC（位置独立コード）としてコード生成しません。

［詳細説明］

- プログラムセクションを PIC（位置独立コード）としてコード生成します。
- PIC においては，関数呼び出しはすべて BSR または BRA 命令を用いて行い，また関数のアドレスを取得するときは PC からの相対アドレスを用いるようにします。これによって，PIC はリンク後に任意のアドレスに配置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。
- ［共通オプション］タブの［PIC/PID］カテゴリの［PIC 機能を有効にする］

［例］

- 関数呼び出し （ただし，branch=32 の場合）

```
void func()
{
        sub();
}

[-pic なし ]
_func:
            MOV.L       #_sub,R14
            JMP         R14

[-pic あり ]
_func:
            MOV.L       #_sub-L11,R14
L11:
            BRA         R14
```

- 関数アドレスの取得

```
void func1(void);
void (*f_ptr)(void);
void func2(void)
{
     f_ptr = func1;
}

[-pic なし ]
_func2:
            MOV.L       #_f_ptr,R4
            MOV.L       #_func1,[R4]
            RTS

[-pic あり ]
_func2:
            MOV.L       #_f_ptr,R4
L11:
            MVFC        PC,R14
            ADD         #_func1-L11,R14
            MOV.L       R14,[R4]
            RTS
```

［備考］

- C++ または EC++ コンパイル時は，pic オプションを選択できません。選択すると，警告として W0523039 メッセージを表示して pic の選択を無効とします。
- PIC である関数のアドレスを，静的初期化の初期化式に使用しないでください。エラー E0523026 となります。
- PIC アドレスを静的初期化に用いる例：

```
void pic_func1(void), pic_func2(int), pic_func3(int);     /* PIC になる        */
void (*fptr1_for_pic) = pic_func1;      /* PIC アドレスを静的初期化で使用： エラー */
struct PIC_funcs{ int code; void (*fptr)(int); };
struct PIC_funcs pic_funcs[] = {
    { 2, pic_func2 },                   /* PIC アドレスを静的初期化で使用： エラー */
    { 3, pic_func3 },                   /* PIC アドレスを静的初期化で使用： エラー */
};
```

- PIC 機能を利用するアプリケーションプログラムのスタートアップを作成する際は，「スタートアップ」ではなく，RX コーディング編の「アプリケーションのスタートアップ」を参照ください。
- PIC 機能については，RX コーディング編の「PIC/PID 機能の利用」の項目も参照してください。

-pid

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-pid[={ 16|32 }]
```

- 省略時解釈
  定数領域セクション C, C_2 および C_1，リテラルセクション L と switch 文分岐テーブルセクション W, W_2 および W_1 を PID（位置独立データ）としません。

［詳細説明］

- 定数領域セクション C, C_2 および C_1，リテラルセクション L と switch 文分岐テーブルセクション W, W_2 および W_1 を PID（位置独立データ）とします。
- PID のアクセスは，すべて PID レジスタからの相対アドレスで行います。これにより，PID はリンク後に任意のアドレスに配置することができます。
- PID 機能を実現するためには，汎用レジスタを 1 本消費します。
- ＜ PID レジスタ＞
- 下記の表の規則に基づき，fint_register オプションの指定に応じて R9 から R13 のうちの 1 本を選択します。なお，fint_register の指定がない場合は R13 を選択します。

表 B.10 **fint_register** オプションと **PID** レジスタの関係

fint_register オプション	PID レジスタ
fint_register 指定なし	R13
fint_register=0	
fint_register=1	R12
fint_register=2	R11
fint_register=3	R10
fint_register=4	R9

- PID レジスタは，PID のアクセスに使用する用途以外には使用されません。
- ＜パラメータ＞
- パラメータは，PID レジスタから定数領域セクションをアクセスする際の，オフセットの最大幅のビット数を 16 または 32 で選択します。
- オフセット幅を省略して pid オプションを選択した場合の解釈は，pid=16 です。pid=16 では，PID レジスタがアクセスできる定数領域セクションのサイズが 64KB ～ 256KB（アクセス幅により変動する）に制限されます。pid=32 では，PID レジスタがアクセスできる定数領域セクションのサイズに制限はありませんが，PID をアクセスするコードのサイズが増大します。
- なお，pid=32 と 外部シンボル割り付け情報が有効な map オプションを同時指定した場合は，割り付け情報により，PID レジスタによるアクセスが可能な場合は pid=16 と同じコードを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［PIC/PID］カテゴリの［PID 機能を有効にする］

［例］

- const 修飾した外部参照シンボルをアクセス

```
extern const int pid;
int work;
void func1()
{
        work = pid;
}
[-pid なし ]
_func1:
            MOV.L       #_pid,R4
            MOV.L       [R4],R5
            MOV.L       #_work,R4
            MOV.L       R5,[R4]
            RTS
[-pid=16 あり ] （* ただし，PID レジスタが R13 の場合）
_func1:
            MOV.L       _pid-__PID_TOP:16[R13],R5
            MOV.L       #_work,R4
            MOV.L       R5,[R4]
            RTS
            .glb        __PID_TOP
[-pid=32 あり ] （* ただし，PID レジスタが R13 の場合）
_func1:
            ADD         #(_pid-__PID_TOP),R13,R6
            MOV.L       [R6],R5
            MOV.L       #_work,R4
            MOV.L       R5,[R4]
            RTS
            .glb        __PID_TOP
```

- const 修飾した外部定義シンボルのアドレスを取得

```
extern const int pid = 1000;
const int *ptr;
void func2()
{
        ptr = &pid;
}
[-pid なし ]
_func2:
            MOV.L       #_ptr,R4
            MOV.L       #_pid,[R4]
            RTS
[-pid あり ] （* ただし，PID レジスタが R13 の場合）
_func2:
            ADD         #(_pid-__PID_TOP),R13,R5
            MOV.L       #_ptr,R4
            MOV.L       R5,[R4]
            RTS
            .glb        __PID_TOP
```

［備考］

- PID である領域のアドレスを，静的初期化の初期化式に使用しないでください。エラー E0523027 となります。

- PID アドレスを静的初期化に用いる例 :

```
extern const int pid_data1;             /* PID になる                        */
const int *ptr1_for_pid = &pid_data1;   /* PID のアドレスで静的初期化 : エラー */
const int pid_data4[] = {1,2,3,4};      /* PID になる                        */
const int *ptr2_for_pid = pid_data4;    /* PID のアドレスで静的初期化 : エラー */
```

- PID 機能を利用するアプリケーションプログラムのスタートアップを作成する際は，「スタートアップ」ではなく，「アプリケーションのスタートアップ」を参照ください。
- pid オプション選択時は，同一の外部変数は，必ずファイル間で const 修飾を統一してください。これは，pid オプションは const 修飾のある変数を PID にするためです。もし const 修飾の統一が不明な外部変数がある場合は，pid オプション（PID 機能）は使用しないでください。
- pid オブション選択時に，ファイル指定のある map オプションを有効にした場合，ファイル間で同じ外部変数に対して const 修飾が統一されていない外部参照変数に対して警告 W0530809 を表示することがあります。表示された変数は PID として扱います。
- C++ または EC++ コンパイル時は，pid オプションを選択できません。選択すると，警告 W0523039 を表示して pid の選択を無効とします。
- pid オプション選択時は，base=rom=< レジスタ > を選択できません。選択すると，警告 W0523039 を表示して base=rom=< レジスタ > の選択を無効とします。
- pid オプションで選択された PID レジスタが，base オプションでも選択された場合は警告 W0511113 になります。
- pid オプションと nouse_pid_register オプションを同時に選択するとエラー E0511150 になります。
- アプリケーションおよび PID 機能の内容については，「PIC/PID 機能の利用」を参照してください。

-nouse_pid_register

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-nouse_pid_register
```

- 省略時解釈
  なし

［詳細説明］

- PID レジスタを使用せずにコード生成を行います。
- PID レジスタは、fint_register オプションの指定内容に基づき、pid オプションと同じ規則で選択します。PID レジスタとして選択されるレジスタは、pid オプションの項目を参照してください。
- PID 機能が有効なアプリケーションプログラムから呼び出されるマスタプログラムは，本オプションでアセンブルする必要があります。このとき，アプリケーションに fint_register オプションの選択がある場合は，マスタプログラムにも同じパラメータの fint_register を選択してください。
- 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［PIC/PID］カテゴリの［PID レジスタをコード生成に使用する］

［例］

-

［備考］

- nouse_pid_register オプションと pid オプションを同時に選択するとエラー E0511150 になります。
- PID レジスタとして選択されるレジスタが，base オプションでも選択された場合は警告 W0511149 になります。
- PID 機能の詳細については，「PIC/PID 機能の利用」の項目を参照してください。

-save_acc

＜コンパイル・オプション / マイコンオプション＞

［指定形式］

```
-save_acc
```

- 省略時解釈
  割り込み関数に対して，アキュムレータ（ACC, ACC0, ACC1）の退避・回復コードを生成しません。

［詳細説明］

- 割り込み関数に対して，アキュムレータ（ACC, ACC0, ACC1）の退避・回復コードを生成します。
- ACC に対する退避・回復コードは、ISA(*1) に RXv1 を選択するか、CPU でマイコン種別を選択 (*2) した場合に生成します。
- ACC0 および ACC1 に対する退避・回復コードは、ISA(*1) に RXv2 を選択した場合に生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［割り込み関数で ACC を退避・回復する］

注 *1）isa オプションまたは環境変数 ISA_RX による選択を指します。
*2) cpu オプションまたは環境変数 CPU_RX による選択を指します。

［備考］

- 生成される退避・回復コードは，#pragma interrupt に acc を選択したときに生成されるコードと同じものです。実際の退避・回復コードは，「#pragma キーワード」の #pragma interrupt (acc，noacc) の項目を参照ください。
- 本オプション指定時は、割り込み時でもアキュムレータの値が保持されるため、C/C++ の演算式に対して MACW 命令などのアキュムレータを用いる命令を生成することがあります。

アセンブル・リンクオプション

＜コンパイル・オプション / アセンブル・リンクオプション＞

アセンブル，リンクオプションには，次のものがあります。

- -asmcmd
- -lnkcmd
- -asmopt
- -lnkopt

-asmcmd

＜コンパイル・オプション / アセンブル・リンクオプション＞

［指定形式］

```
-asmcmd=< ファイル名 >
```

- 省略時解釈
  なし

［詳細説明］

- asrx に渡すアセンブラオプションをサブコマンドファイルにより指定します。

［例］

```
ccrx -isa=rxv1 -asmcmd=file.sub sample.c
```

と記述した場合，以下の 2 行のコマンド記述と同じ意味になります。

```
ccrx -isa=rxv1 -output=src sample.c
asrx -isa=rxv1 -subcommand=file.sub sample.src
```

［備考］

- 本オプションを複数回指定した場合，指定したすべてのサブコマンドファイルが有効となります。

-lnkcmd

＜コンパイル・オプション / アセンブル・リンクオプション＞

［指定形式］

```
-lnkcmd=< ファイル名 >
```

- 省略時解釈
  なし

［詳細説明］

- rlink に渡すリンクオプションをサブコマンドファイルにより指定します。

［例］

```
ccrx -isa=rxv1 -output=abs=tp.abs –lnkcmd=file.sub tp1.c tp2.c
```

と記述した場合，以下の 3 行のコマンド記述と同じ意味になります。

```
ccrx -isa=rxv1 -output=src tp1.c tp2.c
asrx -isa=rxv1 tp1.src tp2.src
rlink -subcommand=file.sub -form=abs -output=tp tp1.obj tp2.obj
```

［備考］

- 本オプションを複数回指定した場合，指定したすべてのサブコマンドファイルが有効となります。
- -lnkcmd オプションに渡すサブコマンドファイルの内容は、最適化リンケージエディタの -subcommand オプションの項目を参照ください。

-asmopt

＜コンパイル・オプション / アセンブル・リンクオプション＞

［指定形式］

```
-asmopt=["] ＜アセンブラオプション＞ ["]
```

- 省略時解釈
  なし

［詳細説明］

- asrx に渡すアセンブラオプションを文字列により指定します。
- 空白をパラメータに含むオプションを指定する場合は，ダブルクォーテーション（"）で囲んで指定します。

［例］

```
ccrx -isa=rxv1 -asmopt="-chkpm" sample.c
```

と記述した場合，以下の 2 行のコマンド記述と同じ意味になります。

```
ccrx -isa=rxv1 -output=src sample.c
asrx -isa=rxv1 -chkpm sample.src
```

［備考］

- 本オプションを複数回指定した場合，指定したすべてのアセンブラオプションが有効となります。

-lnkopt

＜コンパイル・オプション / アセンブル・リンクオプション＞

［指定形式］

```
-lnkopt=["] ＜リンクオプション＞ ["]
```

- 省略時解釈
  なし

［詳細説明］

- rlink に渡すリンクオプションを文字列により指定します。
- 空白をパラメータに含むオプションを指定する場合は，ダブルクォーテーション（"）で囲んで指定します。

［例］

```
ccrx -isa=rxv1 -output=abs=tp.abs -lnkopt="-start=P,C,D/100,B/8000" tp1.c tp2.c
```

と記述した場合，以下の 3 行のコマンド記述と同じ意味になります。

```
ccrx -isa=rxv1 -output=src tp1.c tp2.c
asrx -isa=rxv1 tp1.src tp2.src
rlink -start=P,C,D/100,B/8000 -form=abs -output=tp tp1.obj tp2.obj
```

［備考］

- 本オプションを複数回指定した場合，指定したすべてのリンクオプションが有効となります。
- ひとつの -lnkopt で渡すことのできるリンクオプションはひとつだけです。複数のリンクオプションを渡す場合は、-lnkopt をその数だけ指定してください。

その他オプション

＜コンパイル・オプション / その他オプション＞
その他オプションには，次のものがあります。

- -logo
- -nologo
- -euc
- -sjis
- -latin1
- -utf8
- -big5
- -gb2312
- -outcode
- -subcommand

-logo

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-logo
```

- 省略時解釈
  コピーライト表示が出力されます。

［詳細説明］

- コピーライト表示が出力されます。

［例］

-

［備考］

-

-nologo

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-nologo
```

- 省略時解釈
  コピーライト表示が出力されます。

［詳細説明］

- nologo オプション指定時は，コピーライトの表示の出力が抑止されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［その他］カテゴリの［コピーライトを出力する］

［例］

-

［備考］

-

-euc

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-euc
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を SJIS コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を euc コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

［例］

-

［備考］

-

-sjis

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-sjis
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を SJIS コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を SJIS コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

［例］

-

［備考］

-

-latin1

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-latin1
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を SJIS コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を latin1 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

［例］

-

［備考］

-

-utf8

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-utf8
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を SJIS コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を utf8 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

［例］

-

［備考］

- utf8 オプションは，lang=c99 オプション指定時のみ有効です。

-big5

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-big5
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を BIG5 コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を big5 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

［例］

-

［備考］

- big5 を指定した場合は、outcode にも同じ文字コードを指定しなければなりません。

-gb2312

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-gb2312
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を GB2312 コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を gb2312 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［コンパイル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

［例］

-

［備考］

- gb2312 を指定した場合は、outcode にも同じ文字コードを指定しなければなりません。

-outcode

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-outcode = { euc | sjis | latin1 | utf8 | big5 | gb2312 }
```

- 省略時解釈
  outcode=sjis です。

［詳細説明］

- 文字列，文字定数内の文字を指定した文字コードで出力します。
- オプションと文字コードの関係を以下に示します。

表 B.11 オプションと文字コードの関係（**outcode**）

オプション	文字コード
euc	EUC コード
sjis	SJIS コード
latin1	ISO-Latin1 コード
utf8	UTF-8 コード
big5	Big5 コード
gb2312	GB2312 コード

- 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［オブジェクト］カテゴリの［出力アセンブリ言語ファイルの文字コード］

［例］

-

［備考］

- outcode=utf8 の指定は、lang=c99 オプション指定時のみ有効です。
- outcode=big5 または outcode=gb2312 を指定する場合は、それぞれ big5 または gb2312 オプションもあわせて指定しなければなりません。

-subcommand

＜コンパイル・オプション / その他オプション＞

［指定形式］

```
-subcommand=< サブコマンドファイル名 >
```

- 省略時解釈
  なし

［詳細説明］

- subcommand オプション指定時は，コンパイラ起動時のコンパイラオプションをサブコマンドファイルで指定します。
- サブコマンドファイル中の書式は，コマンドラインの書式と同一です。

［例］

-

［備考］

- 本オプションを複数回指定した場合，指定したすべてのサブコマンドファイルが有効となります。

B.1.3.2 アセンブル・オプション

分類	オプション	説明
ソースオプション	-include	インクルード・ファイルの取り込み先フォルダを指定します。
	-define	マクロ定義を指定します。
	-chkpm	特権命令のチェックします。
	-chkfpu	浮動小数点演算命令のチェックします。
	-chkdsp	DSP 機能命令のチェックします。
オブジェクトオプション	-output	リロケータブル・ファイル名を指定します。
	-debug	オブジェクト・ファイルにデバッグ情報を出力します。
	-nodebug	オブジェクト・ファイルにデバッグ情報を出力しません。
	-goptimize	モジュール間最適化用付加情報を出力します。
	-fpu	FPU 命令が有効であるリロケータブルファイルを生成します。
	-nofpu	FPU 命令が有効ではないリロケータブルファイルを生成します
リストオプション	-listfile	アセンブル・リスト・ファイル出力します。
	-nolistfile	アセンブル・リスト・ファイル出力しません。
	-show	ソース・リスト・ファイルの内容を指定します。
マイコンオプション	-isa	命令セット・アーキテクチャを選択します。
	-cpu	マイコン種別を選択します。
	-endian	エンディアン選択します。
	-fint_register	高速割り込み関数でのみ使用する汎用レジスタを選択します。
	-base	ROM,RAM 用ベースレジスタ選択します。
	-patch	CPU タイプ特有の問題を回避するかどうかを選択します。
	-pic	PIC 機能を有効にします。
	-pid	PID 機能を有効にします。
	-nouse_pid_register	PID レジスタをコード生成に使用しません。
その他オプション	-logo	コピーライトを出力します。
	-nologo	コピーライトを出力しません。
	-subcommand	コマンドオプションを取り込むファイルを指定します。
	-euc	入力プログラムの文字コードを EUC コードと解釈します。
	-sjis	入力プログラムの文字コードを SJIS コードと解釈します。
	-latin1	入力プログラムの文字コードを ISO-Latin1 コードと解釈します。
	-big5	入力プログラムの文字コードを BIG5 コードと解釈します。
	-gb2312	入力プログラムの文字コードを GB2312 コードと解釈します。

ソースオプション

＜アセンブル・オプション / ソースオプション＞
ソースオプションには，次のものがあります。

- -include
- -define
- -chkpm
- -chkfpu
- -chkdsp

-include

＜アセンブル・オプション / ソースオプション＞

［指定形式］

```
-include = < パス名 >[,…]
```

- 省略時解釈
  インクルードファイルの検索は，カレントフォルダ，環境変数 INC_RXA 指定フォルダの順序で行います。

［詳細説明］

- インクルードファイルの存在するパス名を指定します。
- パス名が複数ある場合にはカンマ（,）で区切って指定することができます。
- インクルードファイルの検索は，カレントフォルダ，include オプション指定フォルダ，環境変数 INC_RXA 指定フォルダの順序で行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［ソース］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］

［例］

フォルダ c:￥usr￥inc と c:￥usr￥rxc をインクルードファイルパスとして検索します。

```
asrx -include=c:￥usr￥inc,c:￥usr￥rxc test.src
```

-define

＜アセンブル・オプション / ソースオプション＞

［指定形式］

```
-define = <sub>[,…]
          <sub> : < マクロ名 > = < 文字列 >
```

- 省略時解釈
  なし

［詳細説明］

- マクロ名を対応する文字列に置き換えます。
  （ソースファイル先頭で .DEFINE 指示命令を記述するのと同じです）
- 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［ソース］カテゴリの［マクロ定義］

［例］

-

［備考］

- define オプションと .DEFINE が同時指定された場合，.DEFINE を優先します。

-chkpm

＜アセンブル・オプション / ソースオプション＞

［指定形式］

```
-chkpm
```

- 省略時解釈
  なし

［詳細説明］

- 特権命令を記述するとウォーニング W0551011 を通知します。
- 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［その他］カテゴリの［特権命令をチェックする］

［例］

-

［備考］

- 特権命令の詳細な説明については，RX コーディング編の「インストラクション」を参照してください。

-chkfpu

＜アセンブル・オプション / ソースオプション＞

［指定形式］

```
-chkfpu
```

- 省略時解釈
  なし

［詳細説明］

- 本オプションを指定した場合，浮動小数点演算命令を記述するとウォーニング W0551012 を通知します。
- 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［その他］カテゴリの［浮動小数点演算命令をチェックする］

［例］

-

［備考］

- 浮動小数点演算命令の詳細な説明については，RX コーディング編の「インストラクション」を参照してください。

-chkdsp

＜アセンブル・オプション / ソースオプション＞

［指定形式］

```
-chkdsp
```

- 省略時解釈
  なし

［詳細説明］

- 本オプションを指定した場合，DSP 機能命令を記述するとウォーニング W0551013 を通知します。
- 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［その他］カテゴリの［DSP 機能命令をチェックする］

［例］

なし

［備考］

- DSP 機能命令の詳細な説明については，RX コーディング編の「インストラクション」を参照してください。

オブジェクトオプション

＜アセンブル・オプション / オブジェクトオプション＞
オブジェクトオプションには，次のものがあります。

- -output
- -debug
- -nodebug
- -goptimize
- -fpu
- -nofpu

-output

＜アセンブル・オプション / オブジェクトオプション＞

［指定形式］

```
-output = < 出力ファイル名 >
```

- 省略時解釈
  ソースファイルと同じファイル名で拡張子が「obj」のリロケータブルファイルを出力します。

［詳細説明］

- 出力するリロケータブルファイル名は，出力ファイル名に拡張子がない場合，出力ファイル名に拡張子「.obj」を付加した文字列となり，拡張子がある場合，出力ファイル名の拡張子を「.obj」で置き換えた文字列となります。
- 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［オブジェクト］カテゴリの［出力フォルダ］

［例］

-

［備考］

-

-debug

＜アセンブル・オプション / オブジェクトオプション＞

［指定形式］

```
-debug
```

- 省略時解釈
  リロケータブルファイル内にデバッグ情報を出力しません。

［詳細説明］

- リロケータブルファイル内にデバッグ情報を出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［オブジェクト］カテゴリの［デバッグ情報を出力する］

［例］

-

［備考］

-

-nodebug

＜アセンブル・オプション / オブジェクトオプション＞

［指定形式］

```
-nodebug
```

- 省略時解釈
  リロケータブルファイル内にデバッグ情報を出力しません。

［詳細説明］

- リロケータブルファイル内にデバッグ情報を出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［オブジェクト］カテゴリの［デバッグ情報を出力する］

［例］

-

［備考］

-

-goptimize

＜アセンブル・オプション / オブジェクトオプション＞

［指定形式］

```
-goptimize
```

- 省略時解釈
  モジュール間最適化用付加情報を出力しません。

［詳細説明］

- モジュール間最適化用付加情報を出力します。
- 本オプションを指定したファイルは，リンク時にモジュール間最適化の対象になります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［オブジェクト］カテゴリの［モジュール間最適化用付加情報を出力する］

［例］

-

［備考］

-

-fpu

＜アセンブル・オプション / オブジェクトオプション＞

［指定形式］

```
-fpu
```

- 省略時解釈
  ISA（*1）で命令セットアーキテクチャを選択した場合は fpu です。
  CPU に RX200 を選択（*1）した場合は nofpu，それ以外は fpu です。

注 *1）isa オプションまたは環境変数 ISA_RX による選択を指します。
*2) cpu オプションまたは環境変数 CPU_RX による選択を指します。

［詳細説明］

- FPU 命令が有効であるリロケータブ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。
  ［共通オプション］タブの［最適化］カテゴリの［浮動小数点演算命令を使用する］

［備考］

- CPU に RX200 を選択した場合，fpu を指定するとエラーになります。
- FPU 命令の具体的な内容については，RX コーディング編の「インストラクション」を参照してください。

-nofpu

＜アセンブル・オプション / オブジェクトオプション＞

［指定形式］

```
-nofpu
```

- 省略時解釈
  ISA（*1）で命令セットアーキテクチャを選択した場合は fpu です。
  CPU に RX200 を選択（*1）した場合は nofpu，それ以外は fpu です。

注 *1）isa オプションまたは環境変数 ISA_RX による選択を指します。
*2) cpu オプションまたは環境変数 CPU_RX による選択を指します。

［詳細説明］

- FPU 命令が有効ではないリロケータブ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。
  ［共通オプション］タブの［最適化］カテゴリの［浮動小数点演算命令を使用する］

［備考］

- FPU 命令の具体的な内容については，RX コーディング編の「インストラクション」を参照してください。
- 本オプションを指定した場合、浮動小数点演算命令と、制御レジスタ FPSW の記述をエラーとします。

リストオプション

＜アセンブル・オプション / リストオプション＞
リストオプションには，次のものがあります。

- -listfile
- -nolistfile
- -show

-listfile

＜アセンブル・オプション / リストオプション＞

［指定形式］

```
-listfile[=< ファイル名 >]
```

- 省略時解釈
  アセンブルリストファイルは出力しません。

［詳細説明］

- アセンブルリストファイルを出力します。＜ファイル名＞を指定することもできます。
- ＜ファイル名＞は，「ファイル名の付け方」に従って指定できます。
- listfile オプションで＜ファイル名＞を指定しない場合には，ソースファイルと同じファイル名で，拡張子が「lst」のアセンブルリストファイルが作成されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［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］

［例］

-

［備考］

-

-nolistfile

＜アセンブル・オプション / リストオプション＞

［指定形式］

```
-nolistfile
```

- 省略時解釈
  アセンブルリストファイルは出力しません。

［詳細説明］

- アセンブルリストファイルは出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［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］

［例］

-

［備考］

-

-show

＜アセンブル・オプション / リストオプション＞

［指定形式］

```
-show=<sub>[,…]
       <sub> : { conditionals | definitions | expansions }
```

- 省略時解釈
  なし

［詳細説明］

- アセンブラが出力するリスト内容の設定を行います。各指定をした場合に出力される内容は以下の通りです。

表 B.12 **show** オプション指定一覧

出力種別	内容
conditionals	条件アセンブルで条件が偽となった行もアセンブルリストファイルに出力します。
definitions	.DEFINE で置き換える以前の情報でアセンブルリストファイルに出力します。
expansions	マクロ記述展開行をアセンブルリストファイルに出力します。

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［リスト］カテゴリの［条件アセンブルで偽の行を出力する］，［.DEFINE 置換前の情報を出力する］，［アセンブラ・マクロ記述展開行を出力する］

［例］

-

［備考］

-

マイコンオプション

＜アセンブル・オプション / マイコンオプション＞
マイコンオプションには，次のものがあります。

- -isa
- -cpu
- -endian
- -fint_register
- -base
- -patch
- -pic
- -pid
- -nouse_pid_register

-isa

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-isa = { rxv1 | rxv2 }
```

- 省略時解釈
  環境変数 ISA_RX の内容に従います。

［詳細説明］

- 生成する命令コードの命令セットアーキテクチャ (RXv1 または RXv2) を指定します。
- -isa=rxv1 を指定した場合，RXv1 向けのリロケータブルファイルを生成します。
- -isa=rxv2 を指定した場合，RXv2 向けのリロケータブ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。
- ［共通オプション］タブの［CPU］カテゴリの［命令セット・アーキテクチャ］

［備考］

- 選択した命令セットアーキテクチャでサポートされない命令の記述はエラーとなります。
- -nofpu と -fpu のいずれの選択もない場合に isa オプションを指定すると、fpu が自動的に選択されます。
- -isa オプションを省略し、-cpu オプション、環境変数 CPU_RX および 環境変数 ISA_RX のいずれも指定もない場合は、エラーとなります。
- -cpu オプションと同時に指定することはできません。

-cpu

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-cpu = { rx600 | rx200 }
```

- 省略時解釈
  環境変数 CPU_RX に従います。

［詳細説明］

- マイコン種別を指定します。
- -cpu=rx600 を指定した場合，RX600 向けのリロケータブルファイルを生成します。
- -cpu=rx200 を指定した場合，RX200 向けのリロケータブ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。
- ［共通オプション］タブの［CPU］カテゴリの［命令セット・アーキテクチャ］

［備考］

- 本オプションは、従来機能と互換性を保つためのものです。
- 今後、RX ファミリの製品展開で追加される品種については、命令セットアーキテクチャなどの選択は、-cpu オプションではなく、-isa オプションでの対応となります。新規にアプリケーション開発する際は、-isa オプションを用いてください。
- cpu オプションは、次の記述により -isa オプションと -fpu,-nofpu オプションで置き換えることが可能です。*1
- - cpu=rx600 → -isa=rxv1 -fpu
- - cpu=rx200 → -isa=rxv1 -nofpu
- rx200 を指定した場合，RX200 でサポートされていない浮動小数点演算命令の記述および制御レジスタ FPSW の記述をエラーとします。
- -cpu オプションを省略し、-isa オプション、環境変数 CPU_RX および 環境変数 ISA_RX のいずれの指定もない場合はエラーとなります。
- -isa オプションと同時に指定することはできません。

【注意】

- *1) ソースプログラムにプリデファインドマクロ __RX200、__RX600 の記述がある場合を除きます。

-endian

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-endian={ big | little }
```

- 省略時解釈
  -endian=little です。

［詳細説明］

- endian=big を指定した場合，データのバイト並びが BigEndian になります。
- endian=little を指定した場合，データのバイト並びが LittleEndian 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。
- ［共通オプション］タブの［CPU］カテゴリの［データのエンディアン］

［例］

-

［備考］

-

-fint_register

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-fint_register = {0 | 1 | 2 | 3 | 4 }
```

- 省略時解釈
  fint_register=0 です。

［詳細説明］

- コンパイラの同名のオプションで指定された，高速割り込み専用で使用する汎用レジスタの情報を，リロケータブルファイルに出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［高速割り込み関数でのみ使用する汎用レジスタ］

［例］

-

［備考］

- 本オプションは，プロジェクト全体で指定を統一してください。指定が異なる場合の動作は保証しません。
- 高速割り込み専用に指定した汎用レジスタを，アセンブリ言語ファイルにおいて，高速割り込み以外の用途で使用しないでください。使用した場合の動作は保証しません。
- 本オプションの指定の対象となるレジスタが，base オプションで指定されていた場合，エラーとなります。

-base

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-base = {  rom=< レジスタ >
         | ram=< レジスタ >
         | < アドレス値 > = < レジスタ >}
                          < レジスタ >:= {R8 ～ R13}
```

- 省略時解釈
  なし

［詳細説明］

- コンパイラの同名のオプションで指定された，ベースアドレスとして固定で使用する汎用レジスタの情報を，リロケータブルファイルに出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［ROM 用ベースレジスタ］，［RAM 用ベースレジスタ］，［アドレス値を設定するベースレジスタのアドレス］，［アドレス値を設定するベースレジスタのレジスタ］

［例］

-

［備考］

- 本オプションは，プロジェクト全体で指定を統一してください。指定が異なる場合の動作は保証しません。
- 本オプションで指定した汎用レジスタをベースレジスタ以外の用途に使用しないでください。使用した場合の動作は保証しません。
- 異なる領域に対して同じ汎用レジスタを指定した場合はエラーとなります。
- fint_register オプションで指定した汎用レジスタを本オプションで指定した場合はエラーとなります。

-patch

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-patch = { rx610 }
```

- 省略時解釈
  なし

［詳細説明］

- CPU の品種ごとに特有の問題を回避します。
- -patch=rx610 を指定した場合，RX610 グループで問題となる MVTIPL 命令を未定義命令として扱います。MVTIPL は命令と認識されずにエラーメッセージ E0552113 が出力されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［CPU］カテゴリの［CPU タイプ特有の問題を回避する］

［例］

-

［備考］

-

-pic

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-pic
```

- 省略時解釈
  PIC 機能が有効ではない状態でコード生成されたことを表すリロケータブルオブジェクトを生成します。

［詳細説明］

- PIC 機能が有効な状態でコード生成されたことを表すリロケータブルオブジェクトを生成します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［PIC/PID］カテゴリの［PIC 機能を有効にする］

［例］

-

［備考］

- 本オプションに矛盾するコードをアセンブリコードに記述しても，チェックされません。
- PIC 機能が有効なリロケータブルオブジェクトは，PIC 機能が有効でないリロケータブルオブジェクトとはリンクできません。
- PIC 機能については，「PIC/PID 機能の利用」の項目も参照してください。

-pid

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-pid[={16|32}]
```

- 省略時解釈
  PID 機能が有効ではない状態でコード生成されたことを表すリロケータブルオブジェクトを生成します。

［詳細説明］

- PID 機能が有効な状態でコード生成されたことを表すリロケータブルオブジェクトを生成します。
- ＜PID レジスタ＞
  下記の表の規則に基づき，fint_register オプションの指定に応じて R9 から R13 のうちの 1 本を選択します。なお，fint_register の指定がない場合は R13 を選択します。

表 B.13 **fint_register** オプションと **PID** レジスタの関係

fint_register オプション	PID レジスタ
fint_register 指定なし	R13
fint_register=0	
fint_register=1	R12
fint_register=2	R11
fint_register=3	R10
fint_register=4	R9

- PID レジスタは，PID のアクセスに使用する用途以外には使用されません。
- ＜パラメータ＞
  パラメータの意味は，コンパイラの同名オプションと同じです。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［共通オプション］タブの［PIC/PID］カテゴリの［PID 機能を有効にする］

［例］

-

［備考］

- 本オプションに矛盾するコードをアセンブリコードに記述しても，チェックされません。
- PID 機能が有効なリロケータブルオブジェクトは，PID 機能が有効でないリロケータブルオブジェクトとはリンクできません。
- pid オプションで選択された PID レジスタが，base オプションでも選択された場合はエラー F0553111 になります。
- pid オプションと nouse_pid_register オプションを同時に選択するとエラー F0553103 になります。
- PID 機能については，「PIC/PID 機能の利用」の項目も参照してください。

-nouse_pid_register

＜アセンブル・オプション / マイコンオプション＞

［指定形式］

```
-nouse_pid_register
```

- 省略時解釈
  なし

［詳細説明］

- PID レジスタを使用しないでコード生成されたことを表すリロケータブルオブジェクトを生成します。
- アセンブリソースファイル中で PID レジスタを使用している場合に，エラーとして E0552058 メッセージを出力します。なお、PID レジスタの名称がアセンブラ言語仕様の「代用レジスタ名」と同一の場合は、本オプションを指定してもエラーにはなりません。
- PID 機能が有効なアプリケーションプログラムから呼び出されるマスタプログラムは，本オプションでアセンブルする必要があります。このとき，アプリケーションに fint_register オプションの選択がある場合は，マスタプログラムにも同じパラメータの fint_register を選択してください。
- 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［PIC/PID］カテゴリの［PID レジスタをコード生成に使用する］

［例］

-

［備考］

- nouse_pid_register オプションと pid オプションを同時に選択するとエラー F0553103 になります。
- nouse_pid_register オプションで選択されたレジスタが，base オプションでも選択された場合はエラー F0553112 になります。
- PID 機能の詳細については，「PIC/PID 機能の利用」の項目を参照してください。

その他オプション

＜アセンブル・オプション / その他オプション＞
その他オプションには，次のものがあります。

- -logo
- -nologo
- -subcommand
- -euc
- -sjis
- -latin1
- -big5
- -gb2312

-logo

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-logo
```

- 省略時解釈
  コピーライト表示が出力されます。

［詳細説明］

- コピーライト表示が出力されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［その他］カテゴリの［コピーライトを出力する］

［例］

-

［備考］

-

-nologo

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-nologo
```

- 省略時解釈
  コピーライト表示が出力されます。

［詳細説明］

- コピーライトの表示の出力が抑止されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［その他］カテゴリの［コピーライトを出力する］

［例］

-

［備考］

-

-subcommand

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-subcommand=< ファイル名 >
```

- 省略時解釈
  なし

［詳細説明］

- subcommand オプション指定時は，アセンブラ起動時のアセンブラオプションをサブコマンドファイルで指定します。サブコマンドファイル中の書式は，コマンドラインの書式と同一です。

［例］

- ＜サブコマンドファイル opt.sub の内容＞

```
-listfile
-debug
```

- ＜コマンドライン指定＞
  (1) のコマンドライン指定を行うと，アセンブラで (2) のように解釈されます。

```
(1) asrx -endian=big -subcommand=opt.sub test.src
(2) asrx -endian=big -listfile -debug test.src
```

-

-euc

- ＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-euc
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を sjis コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を euc コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

-

-

-sjis

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-sjis
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を sjis コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を sjis コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

-

-

-latin1

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-latin1
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を sjis コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を latin1 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

-

-

-big5

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-big5
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を BIG5 コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を big5 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

-

-gb2312

＜アセンブル・オプション / その他オプション＞

［指定形式］

```
-gb2312
```

- 省略時解釈
  文字列，文字定数およびコメント内の文字を GB2312 コードで扱います。

［詳細説明］

- 文字列，文字定数およびコメント内の文字を gb2312 コードで扱います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［アセンブル・オプション］タブの［ソース］カテゴリの［入力プログラムの文字コード］

B.1.3.3 最適化リンケージエディタ（rlink）・オプション

分類	オプション	説明
入力オプション	-Input	リロケータブル・ファイルを指定します。
	-library	ライブラリ・ファイルを指定します。
	-binary	バイナリ・ファイルを指定します。
	-define	シンボル定義を指定します。
	-entry	エントリシンボル / アドレスを指定します。
	-noprelink	プレリンカを起動しません。
出力オプション	-form	出力ファイル形式を選択します。
	-debug	デバッグ情報をロード・モジュール・ファイル内に出力します。
	-sdebug	デバッグ情報を .dbg ファイルに出力します。
	-nodebug	デバッグ情報を出力しません。
	-record	レコードサイズを選択します。
	-rom	ROM から RAM へマップするセクションを指定します。
	-output	出力ファイル名を指定します。
	-map	外部シンボル割り付け情報ファイルを出力します。
	-space	出力範囲のメモリの空き領域をデータで充填します。
	-message	インフォメーションレベル・メッセージの出力します。
	-nomessage	出力抑止メッセージを指定します。
	-msg_unused	参照されない外部定義シンボルをメッセージ出力します。
	-byte_count	データ・レコードのバイト数を指定します。
	-crc	CRC コードの出力形式を指定します。
	-padding	終端にパディングを埋め込みます。
	-vectn	可変ベクタの特定ベクタ番号へのアドレスを指定します。（RX ファミリ、M16C ファミリ向け）
	-vect	可変ベクタの空き領域へのアドレスを指定します。（RX ファミリ、M16C ファミリ向け）
	-jump_entries_for_pic	ジャンプ・テーブル・ファイルを出力します。（RX ファミリの PIC 機能向け）
リストオプション	-list	リンケージ・リスト・ファイルを出力します。
	-show	リンケージ・リスト・ファイルの出力する内容を選択します。

分類	オプション	説明
最適化オプション	-optimize	リンク時最適化内容を選択します。
	-nooptimize	リンク時最適化を行いません。
	-samesize	共通コード統合の対象となる最小サイズを指定します。
	-symbol_forbid	未参照シンボル削除抑止シンボルを指定します。
	-samecode_forbid	共通コード統合抑止シンボルを指定します。
	-section_forbid	最適化抑止セクションを指定します。
	-absolute_forbid	最適化抑止アドレス範囲を指定します。
セクションオプション	-start	セクションの開始アドレスを指定します。
	-fsymbol	外部定義シンボルをファイル出力するセクションを指定します。
	-aligned_section	セクション・アライメントを 16 バイトにします。
ベリファイオプション	-cpu	アドレスの整合性をチェックします。
	-contiguous_section	セクション分割の対象外セクションを指定します。
その他オプション	-s9	S9 レコードを終端に出力します。
	-stack	スタック使用量情報ファイルを出力します。
	-compress	デバッグ情報を圧縮します。
	-nocompress	デバッグ情報を圧縮しません。
	-memory	リンク時に使用するメモリ量を選択します。
	-rename	変更するシンボル名、セクション名を指定します。
	-delete	削除するシンボル名、モジュール名を指定します。
	-replace	置換するライブラリ・モジュール指定します。
	-extract	ライブラリファイルから抽出するモジュールを指定します。
	-strip	アブソリュートファイル、ライブラリファイルのデバッグ情報を削除します。
	-change_message	変更するインフォメーション、ウォーニング、エラーレベルのメッセージレベル、メッセージを指定します。
	-hide	ローカルシンボル名情報を削除します。
	-total_size	リンク後の合計セクションサイズを標準出力に表示します。
サブコマンドファイルオプション	-subcommand	コマンドオプションを取り込むファイルを指定します。
残りのオプション	-logo	コピーライトを出力します。
	-nologo	コピーライトの出力しません。
	-end	END より前に指定したオプション列を実行します。
	-exit	オプション指定の終了を指定します。

入力オプション

＜最適化リンケージエディタ（rlink）・オプション / 入力オプション＞

入力オプションには，次のものがあります。

- -Input
- -library
- -binary
- -define
- -entry
- -noprelink

-Input

＜最適化リンケージエディタ（rlink）・オプション / 入力オプション＞

［指定形式］

```
-Input = ＜サブオプション＞[{,| △ }…]
         ＜サブオプション＞ : ＜ファイル名＞[(＜モジュール名＞[,…])]
```

- 省略時解釈
  なし

［詳細説明］

- 入力ファイルを指定します。複数ある場合にはカンマ（,）または空白文字で区切って指定します。
- ワイルドカード（*,?）も指定できます。ワイルドカードで指定した文字列はアルファベット順に展開します。数字と英文字は数字が先，英大文字と英小文字は英大文字が先になります。
- 入力ファイルとして指定できるのは，コンパイラ，アセンブラ出力オブジェクトファイル，最適化リンケージエディタ出力のリロケータブルファイルおよびアブソリュートファイルです。またライブラリ名 (＜モジュール名＞) の形式で，ライブラリ内モジュールを入力ファイルとして指定することもできます。モジュール名は拡張子なしで指定します。
- 入力ファイル名に拡張子の指定がない場合，モジュール名がない時は「obj」，モジュール名がある時は「lib」を仮定します。
- 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］カテゴリの［オブジェクト・モジュール・ファイル］
- ［ライブラリアン・オプション］タブの［入力］カテゴリの［オブジェクト・モジュール・ファイル］

［例］

```
input=a.obj lib1(e)                ; a.obj と lib1.lib 内のモジュール e を入力します。
input=c*.obj                       ; c で始まる拡張子 obj のファイルをすべて入力します。
```

［備考］

- form=object および extract 指定時，本オプションは無効です。
- コマンドライン上で入力ファイルを指定する場合は，input 無しで指定します。

-library

＜最適化リンケージエディタ（rlink）・オプション / 入力オプション＞

［指定形式］

```
-library= < ファイル名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- ライブラリファイルを指定します。複数ある場合にはカンマ（,）で区切って指定します。
- ワイルドカード（*,?）も指定できます。ワイルドカードで指定した文字列はアルファベット順に展開します。数字と英文字は数字が先，英大文字と英小文字は英大文字が先になります。
- 入力ファイル名に拡張子の指定がない場合は，「lib」を仮定します。
- form=library オプションまたは extract オプション指定時は，ライブラリファイルを編集対象ライブラリとして入力します。
- それ以外の場合は，入力ファイルとして指定されたファイル間でのリンケージ処理後に，未定義シンボルをライブラリファイルから検索します。
- ライブラリファイル内シンボルの検索は，ライブラリオプション指定ユーザライブラリファイル（指定順），ライブラリオプション指定システムライブラリファイル（指定順），デフォルトライブラリ（環境変数 HLNK_LIBRARY1,2,3）の順序で行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［リンク・オプション］タブの［入力］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］
- ［ライブラリアン・オプション］タブの［入力］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］

［例］

```
library=a.lib,b             ; a.lib と b.lib を入力します。
library=c*.lib              ; c で始まる拡張子 lib のファイルをすべて入力します。
```

［備考］

-

-binary

＜最適化リンケージエディタ（rlink）・オプション / 入力オプション＞

［指定形式］

```
-binary = ＜サブオプション＞[,…]
          ＜サブオプション＞ :
          ＜ファイル名＞(＜セクション名＞[:＜アライメント数＞][/＜セクション属性＞][,＜シンボル名＞])
                      ＜セクション属性＞ : CODE | DATA
                      ＜アライメント数＞ : 1 | 2 | 4 | 8 | 16 | 32 ( デフォルトは 1)
```

- 省略時解釈
  なし

［詳細説明］

- 入力バイナリファイルを指定します。複数ある場合にはカンマ（,）で区切って指定します。
- ファイル名に拡張子の指定がない場合は，「bin」を仮定します。
- 入力したバイナリデータは，指定したセクションのデータとして配置します。セクションのアドレスは start オプションで指定します。セクションは省略できません。
- またシンボルを指定することにより，定義シンボルとしてリンクすることもできます。C/C++ プログラムで参照している変数名の場合，プログラム中での参照名先頭に _ を付加します。
- 本オプションで指定したセクションには，セクション属性，アライメント数の指定が可能です。
- セクション属性は，CODE または DATA を指定できます。
- セクション属性の指定が無い場合，デフォルトとして書き込み，読み取り，実行すべての属性が有効になります。
- アライメント数に指定可能な値は 2 の累乗です。それ以外の値を指定することはできません。
- アライメント数の指定がない場合，デフォルト値として 1 が有効になります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［リンク・オプション］タブの［入力］カテゴリの［バイナリ・データ・ファイル］
- ［ライブラリアン・オプション］タブの［入力］カテゴリの［バイナリ・データ・ファイル］

［例］

```
input=a.obj
start=P,D*/200
binary=b.bin(D1bin),c.bin(D2bin:4,_datab)
form=absolute
```

- b.bin を D1bin セクションとして，0x200 番地から配置します。
- c.bin を D2bin セクション（アライメント数 4）として，D1bin の後に配置します。
- c.bin データを定義シンボル _datab としてリンクします。

［備考］

- form={object | library} または strip 指定時，本オプションは無効です。
- また入力オブジェクトファイル指定がない場合，本オプションは指定できません。

-define

＜最適化リンケージエディタ（rlink）・オプション / 入力オプション＞

［指定形式］

```
-define = < サブオプション >[,…]
          < サブオプション > : < シンボル名 > = {< シンボル名 > | < 数値 >}
```

- 省略時解釈
  なし

［詳細説明］

- 未定義シンボルを外部定義シンボルまたは数値で強制定義します。
- 数値は 16 進数で指定します。先頭が A ～ F の場合は先にシンボルを検索し，該当するシンボルがなければ数値として解釈します。先頭に 0 を付加した場合は常に数値と解釈します。シンボル名が C/C++ 変数名の場合，プログラム中での定義名先頭に _ を付加します。C++ 関数名の場合は（main 関数は除く）引数列を含めたプログラム中の定義名をダブルクォーテーションで囲んで指定します。ただし，引数が void の場合は，" 関数名 ()" で指定します。
- 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］カテゴリの［シンボル定義］

［例］

```
define=_sym1=data           ;_sym1 を外部定義シンボル data と同値として定義します。
define=_sym2=4000           ;_sym2 を 0x4000 として定義します。
```

［備考］

- form={object | relocate | library} 指定時，本オプションは無効です。

-entry

＜最適化リンケージエディタ（rlink）・オプション / 入力オプション＞

［指定形式］

```
-entry = {< シンボル名 > | < アドレス >}
```

- 省略時解釈
  なし

［詳細説明］

- 実行開始アドレスを外部定義シンボルまたはアドレスで指定します。
- アドレスは 16 進数で指定します。先頭が A ～ F の場合は先に定義シンボルを検索し，該当するシンボルがなければアドレスと判断します。先頭に 0 を付加した場合は常にアドレスと解釈します。
- シンボル名は，C 関数名の場合プログラム中での定義名先頭に _ を付加します。C++ 関数名の場合は（main 関数は除く）引数列を含めたプログラム中の定義名をダブルクォーテーションで囲んで指定します。ただし，引数が void の場合は，" 関数名 ()" で指定します。
- コンパイル，アセンブル時に entry シンボルを指定している場合，本オプション指定を優先します。
- 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］カテゴリの［実行開始アドレスを指定する］，［実行開始アドレス］

［例］

```
entry=_main           ;C/C++ の main 関数を実行開始アドレスとして設定します。
entry="init()"        ;C++ の init 関数を実行開始アドレスとして設定します。
entry=100             ;0x100 を実行開始アドレスとして設定します。
```

［備考］

- form={object | relocate | library} または strip 指定時，本オプションは無効です。
- 未参照シンボル削除最適化（optimize=symbol_delete）指定時には，実行開始アドレスは必ず必要です。指定がない場合は，未参照シンボル削除最適化指定は無効です。本オプションでアドレスを指定している場合は，未参照シンボル削除最適化を無効にします。

-noprelink

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-noprelink
```

- 省略時解釈
  プレリンカを起動します。

［詳細説明］

- プレリンカの起動を抑止します。
- プレリンカは，C++ テンプレートインスタンスの自動生成機能および実行時型検査機能をサポートします。C++ テンプレート機能および実行時型検査機能を使用していない場合は，noprelink オプションを指定してください。リンク時間が短くなります。
- 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［入力］カテゴリの［プレリンカを起動する］
- ［ライブラリアン・オプション］タブの［入力］カテゴリの［プレリンカを起動する］

［例］

-

［備考］

- extract または strip 指定時，本オプションは無効です。
- C++ テンプレート機能および実行時型検査機能を使用し，form=lib または，form=rel を指定する場合には，noprelink を指定しないでください。

出力オプション

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞
出力オプションには，次のものがあります。

- -form
- -debug
- -sdebug
- -nodebug
- -record
- -rom
- -output
- -map
- -space
- -message
- -nomessage
- -msg_unused
- -byte_count
- -crc
- -padding
- -vectn
- -vect
- -jump_entries_for_pic

-form

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-form = {Absolute | Relocate | Object | Library[={S|U}] | Hexadecimal | Stype | Binary}
```

- 省略時解釈
  form=absolute です。

［詳細説明］

- 出力形式を指定します。
- サブオプションの一覧を表 B.13 に示します。

表 B.14 **form** オプションのサブオプション一覧

No	サブオプション名	内 容
1	absolute	アブソリュートファイルを出力します。
2	relocate	リロケータブルファイルを出力します。
3	object	オブジェクトファイルを出力します。extract オプションでライブラリから 1 個のモジュールをオブジェクトファイルとして取り出すときに使用します。
4	library	ライブラリファイルを出力します。 library=s 指定時，出力ライブラリファイルをシステムライブラリとします。 library=u 指定時，出力ライブラリファイルをユーザライブラリとします。 省略時解釈は，library=u です。
5	hexadecimal	インテル HEX 形式ファイルを出力します。インテル HEX フォーマットは「インテル HEX 形式ファイル」を参照してください。
6	stype	モトローラ S 形式ファイルを出力します。モトローラ S フォーマットは「モトローラ S 形式ファイル」を参照してください。
7	binary	バイナリ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。
- ［共通オプション］タブの [出力] カテゴリの［出力ファイルの種類］
- ［ヘキサ出力オプション］タブの［出力］カテゴリの［ヘキサ・ファイル・フォーマット］

［備考］

出力形式と入力ファイル，他オプションとの関係を表 B-14 に示します。

表 B.15 出力形式と入力ファイル，他オプションとの関係

No	出力形式	指定オプション	入力可能なファイル形式	指定可能なオプション[注 1]
1	Absolute	strip あり	アブソリュートファイル	input, output
		上記以外	オブジェクトファイル リロケータブルファイル バイナリファイル ライブラリファイル	input, library, binary, debug/nodebug, sdebug, cpu, start, rom, entry, output, map, hide, optimize/nooptimize, samesize, symbol_forbid, samecode_forbid, section_forbid, absolute_forbid, compress, rename, delete, define, fsymbol, stack, noprelink, memory, msg_unused, show=symbol,reference,xreference,total_size, vector, jump_entries_for_pic, aligned_section
2	Relocate	extract あり	ライブラリファイル	library, output
		上記以外	オブジェクトファイル リロケータブルファイル バイナリファイル ライブラリファイル	input, library, debug/nodebug, output, hide, rename, delete, noprelink, msg_unused, show=symbol,xreference,total_size
3	Object	extract あり	ライブラリファイル	library, output
4	Hexadecimal Stype Binary		オブジェクトファイル リロケータブルファイル バイナリファイル ライブラリファイル	input, library, binary, cpu, start, rom, entry, output, map, space, optimize/nooptimize, samesize, symbol_forbid, samecode_forbid, section_forbid, absolute_forbid, rename, delete, define, fsymbol, stack, noprelink, record, s9[注 2], byte_count[注 3], memory, msg_unused, show=symbol,reference,xreference,total_size, vector, jump_entries_for_pic, aligned_section
			アブソリュートファイル	input, output, record, s9[注 2], byte_count[注 3], show=symbol, reference, xreference
5	Library	strip あり	ライブラリファイル	library, output, memory[注 4], show=symbol, section
		extract あり	ライブラリファイル	library, output
		上記以外	オブジェクトファイル リロケータブルファイル	input, library, output, hide, rename, delete, replace, noprelink, memory[注 4], show=symbol, section

注 1. message/nomessage, change_message, logo/nologo, form, list, subcommand は常に指定できます。

注 2. s9 は出力形式が form=stype のときだけ指定できます。

注 3. byte_count は出力形式が form= hexadecimal のときだけ指定できます。

注 4. hide 指定する場合は使用できません。

-debug

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-debug
```

- 省略時解釈
  出力ファイル中にデバッグ情報を出力します。

［詳細説明］

出力ファイル中にデバッグ情報を出力します。

- output オプションで複数ファイル出力を指定時に debug オプションを指定したときは，sdebug オプションと解釈して，< 先頭出力ファイル名 >.dbg 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［出力］カテゴリの［デバッグ情報を出力する］
- ［ライブラリアン・オプション］タブの［出力］カテゴリの［デバッグ情報を出力する］

［例］

-

［備考］

- form={object | library | hexadecimal | stype | binary}，strip，または，extract 指定時，本オプションは無効です。

-sdebug

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-sdebug
```

- 省略時解釈
  出力ファイル中にデバッグ情報を出力します。

［詳細説明］

- < 出力ファイル名 >.dbg ファイルにデバッグ情報を出力します。
- form=relocate 指定時に sdebug オプションを指定したときは，debug オプションと解釈します。
- 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］カテゴリの［デバッグ情報を出力する］

［備考］

- form={object | library | hexadecimal | stype | binary}，strip，または，extract 指定時，本オプションは無効です。

-nodebug

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-nodebug
```

- 省略時解釈
  出力ファイル中にデバッグ情報を出力します。

［詳細説明］

- デバッグ情報を出力しません。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［リンク・オプション］タブの［出力］カテゴリの［デバッグ情報を出力する］
- ［ライブラリアン・オプション］タブの［出力］カテゴリの［デバッグ情報を出力する］

［例］

-

［備考］

- form={object | library | hexadecimal | stype | binary}，strip，または，extract 指定時，本オプションは無効です。

-record

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-record = {H16 | H20 | H32 | S1 | S2 | S3}
```

- 省略時解釈
  それぞれのアドレスに合わせて混在したデータレコードを出力します。

［詳細説明］

アドレス範囲に関係なく，一定のデータレコードで出力します。
指定したデータレコードより大きいアドレスが存在した場合，アドレスに合わせてデータレコードを選択します。
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［ロード・モジュール・ファイル変換］カテゴリの［レコードサイズを統一する］，
［レコードサイズを統一する］

［例］

-

［備考］

- form=hexadecimal または stype 指定がないとき，本オプションは無効です。

-rom

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-rom = ＜サブオプション＞[,…]
       ＜サブオプション＞ : <ROM セクション名 >=<RAM セクション名 >
```

- 省略時解釈
  なし

［詳細説明］

- 初期化データ領域の ROM 用，RAM 用領域を確保し，ROM セクション内定義シンボルを RAM セクション内アドレスになるようリロケーションします。
- ROM セクションには初期値のあるリロケータブルセクションを指定します。
- RAM セクションには存在しないセクションまたはサイズ 0 のリロケータブルセク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。
- ［リンク・オプション］タブの［出力］カテゴリの［ROM から RAM へマップするセクション］

［例］

```
rom=D=R
start=D/100,R/8000
```

- D セクションと同サイズの R セクションを確保し，D セクション内定義シンボルを R セクション上のアドレスでリロケーションします。

［備考］

- form={object | relocate | library} または strip 指定時，本オプションは無効です。

-output

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-output ＝ < サブオプション >[,…]
          < サブオプション > ： < ファイル名 >[=< 出力範囲 >]
          < 出力範囲 >: {< 先頭アドレス >-< 終了アドレス > | < セクション名 >[:…]}
```

- 省略時解釈
  < 先頭入力ファイル名 >.< デフォルト拡張子 > です。
- デフォルト拡張子は，次のようになります。
  form=absolute :「abs」，form=relocate :「rel」，form=object :「obj」，form=library :「lib」，form=hexadecimal :「hex」，form=stype :「mot」，form=binary :「bin」

［詳細説明］

- 出力ファイル名を指定します。form={absolute | hexadecimal | stype | binary} のときは，複数ファイルを指定できます。アドレスは 16 進数で指定します。先頭が A ～ F の場合は先にセクションを検索し，該当するセクションがなければアドレスと判断します。先頭に 0 を付加した場合は常にアドレスと解釈します。
- 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［出力ファイルの種類］，［よく使うオプション（ヘキサ出力）］カテゴリの［ヘキサ・ファイルを出力する］，［ヘキサ・ファイル・フォーマット］，［出力フォルダ］，［出力ファイル名］，［分割出力ファイル］
- ［リンク・オプション］タブの［出力］カテゴリの［出力フォルダ］，［出力ファイル名］
- ［ヘキサ出力オプション］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］，［分割出力ファイル］
- ［ライブラリアン・オプション］タブの［出力］カテゴリの［出力フォルダ］，［出力ファイル名］

［例］

```
output=file1.abs=0-ffff,file2.abs=10000-1ffff
```

- 0 ～ 0xffff 間を file1.abs に，0x10000 ～ 0x1ffff 間を file2.abs に出力します。

```
output=file1.abs=sec1:sec2,file2.abs=sec3
```

- sec1,sec2 セクションを file1.abs に，sec3 セクションを file2.abs に出力します。

［備考］

- マイコン種別が RX ファミリでビッグエンディアンのときに，セクション単位で出力する場合は，セクションサイズを 4 の倍数にしてください。

-map

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-map [= < ファイル名 >]
```

- 省略時解釈
  なし

［詳細説明］

- コンパイラが外部変数アクセス最適化で使用する外部変数割り付け情報ファイルを出力します。
- < ファイル名 > を指定しなかった場合は，output オプションで指定したファイル名，もしくは先頭入力ファイル名で，拡張子が bls のファイルを出力します。
- 外部変数割り付け情報ファイル作成時の変数宣言順と，再コンパイル後のオブジェクトを読み込んだ時の変数宣言順が変わっている場合はエラーを出力します。
- 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［出力］カテゴリの［外部シンボル割り付け情報ファイルを出力する］

［例］

-

［備考］

- form={absolute | hexadecimal | stype | binary} を指定した場合のみ，本オプションは有効です。

-space

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-space [ = {< 数値 > | Random}]
```

- 省略時解釈
  なし

［詳細説明］

- 出力範囲のメモリの空き領域を，ユーザが指定するデータで充填します。
- 充填するデータとしては，乱数，もしくは 16 進数の数値を指定することができます。
- 空きエリアを埋める方法は，output オプション指定時の出力範囲指定方法によって下記のように異なります。
- 出力範囲 : セクション指定
- 指定されたセクション間に空きが存在した場合に指定データを出力
- 出力範囲 : アドレス範囲指定
- 指定された範囲内に空きが存在した場合に指定データを出力
- 出力データサイズは，1,2,4 バイト単位で有効となります。出力データサイズは space オプションに指定する 16 進数の数値で決まります。3 バイトデータを指定した場合，上位桁を 0 拡張し 4 バイトのデータとして扱われます。また，奇数桁データを指定した場合も，上位桁に 0 拡張して偶数桁入力として扱われます。
- 空きエリアのサイズが出力データサイズの倍数でない場合，出力できるだけ出力し，メッセージによる警告を行います。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［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［出力範囲のメモリの空き領域をデータで充填する］，［空きエリア出力データ］

［例］

-

［備考］

- 本オプションにてサブオプションの指定がされなかった場合は，空きエリアへの出力は行いません。
- 本オプションは form={ binary |stype | hexadecimal} オプションを指定した場合にのみ有効となります。
- output オプションによる出力範囲指定がされなかった場合は，本オプション指定は無効となります。

-message

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-message
```

- 省略時解釈
  nomessage (インフォメーションレベルメッセージの出力を抑止) です。

［詳細説明］

- インフォメーションレベルメッセージを出力します。
- 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［出力］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］
- ［ライブラリアン・オプション］タブの［出力］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］

［例］

-

［備考］

-

-nomessage

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-nomessage [= <サブオプション>[,…]]
             <サブオプション> : <エラー番号>[-<エラー番号>]
```

- 省略時解釈
  nomessage (インフォメーションレベルメッセージの出力を抑止) です。

［詳細説明］

- インフォメーションレベルメッセージの出力を抑止します。またエラー番号を指定すると，指定したエラー番号のメッセージ出力を抑止できます。ハイフン（-）を使用して抑止するエラー番号の範囲を指定することもできます。エラー番号としてウォーニング，エラーレベルメッセージ番号を指定した場合，change_message でインフォメーションレベルに変更したと仮定し，メッセージ出力を抑止します。
- エラー番号には，コンポーネント番号（05），発生フェーズ（6）に続けて出力される 4 桁の数値（M0560004 の場合は 0004）を指定します。4 桁の数値の先頭が 0 で始まる場合は，0 を省略することができます（M0560004 の場合は 4）。
- 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］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］，［抑止するインフォメーションレベル・メッセージ番号］
- ［ライブラリアン・オプション］タブの［出力］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］，［抑止するインフォメーションレベル・メッセージ番号］

［例］

- M0560004、M0560200 ～ M0560203 および M0561300 のメッセージ出力を抑止します。

```
nomessage=4,200-203,1300
```

-msg_unused

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-msg_unused
```

- 省略時解釈
  なし

［詳細説明］

- 本オプションを指定した場合，リンク処理の中で一度も参照されることのなかった外部定義シンボルを，メッセージ出力によってユーザに知らせます。
- 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］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］
- ［ライブラリアン・オプション］タブの［出力］カテゴリの［インフォメーションレベル・メッセージ出力を有効にする］

［例］

```
rlink -msg_unused a.obj
```

［備考］

- 入力ファイルが absolute 形式の場合，本オプション指定は無効です。
- メッセージ出力させるためには，同時に message オプションの指定が必要です。
- コンパイル時にインライン展開された関数に対してメッセージ出力する場合があります。その場合，関数定義に static 宣言することで，メッセージ出力を抑えることができます。
- 以下のいずれかに該当する場合，参照関係の解析が正しく行うことができず，メッセージ出力により通知される情報が不正確となります。
- 同一ファイル内の定数シンボルへの参照
- コンパイル時に最適化が有効で，直下の関数を呼び出す場合

-byte_count

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-byte_count=< 数値 >
```

- 省略時解釈
  バイト数最大値は FF として Intel-Hex ファイルを生成します。

［詳細説明］

- Intel-Hex 形式ファイルを生成する際に，データレコードのバイト数最大値を指定するためのオプションです。
- バイト数としては，1byte の 16 進数（01 ～ FF）を指定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。
- ［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［データ・レコードのバイト数を指定する］，［データ・レコードのバイト数最大値］

［例］

```
byte_count=10
```

［備考］

- 生成するファイル形式が Intel-Hex 形式（form=hex）ではない場合，本オプションは無効です。

-crc

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-CRc = ＜サブオプション＞
       ＜サブオプション＞: ＜出力位置＞=＜計算範囲＞[/＜多項式＞][:＜エンディアン＞]
                           ＜出力位置＞: ＜アドレス＞
                           ＜計算範囲＞: ＜先頭アドレス＞-＜終了アドレス＞[,...]
                           ＜多項式＞  : { CCITT | 16 }
                           ＜エンディアン＞: {BIG|LITTLE}
```

- 省略時解釈
  なし

［詳細説明］

- 計算範囲で指定された内容を下位アドレスから上位アドレスの順で CRC（Cyclic Redundancy Check）演算を行い，計算結果を出力位置のアドレスに出力します。
- エンディアンは，RX ファミリの場合に指定可能なオプションです。エンディアンを指定した場合は，エンディアンにしたがって，計算結果を出力位置のアドレスに出力します。指定しない場合は，アブソリュートファイルのエンディアンで計算結果を出力位置のアドレスに出力します。
- 多項式は CRC-CCITT または CRC-16 を選択できます。（デフォルトは CRC-CCITT）
- 多項式

```
CRC-CCITT
    X^16+X^12+X^5+1
        ビット表現（10001000000100001）
CRC-16
    X^16+X^15+X^2+1
        ビット表現（11000000000000101）
```

- 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［CRC 演算結果を出力する］，［出力アドレス］，［計算範囲］

［例］

- rlink *.obj -form=stype -start=P1,P2/1000,P3/2000
  -crc=2FFE=1000-2FFD -output=out.mot=1000-2FFF

- crc オプション：-crc=2FFE=1000-2FFD
  - 0x1000 ～ 0x2FFD の領域に対して CRC 演算を行い，その結果を 0x2FFE 番地に出力します。
  - 計算範囲にある空き領域は space オプションが指定されていない場合，space=0xFF が指定されていると仮定して，CRC 演算を行います。
- output オプション：-output=out.mot=1000-2FFF
  - space オプションが指定されていないため，空きの領域は「out.mot」ファイルに出力されません。CRC 演算は，空き領域では 0xFF で計算を行いますが，0xFF を埋めることはありません。

注 1. CRC 出力位置は，計算範囲に含むことは出来ません。

注 2. CRC 出力位置は output オプションの出力範囲に含まれている必要があります。

- rlink *.obj -form=stype -start=P1/1000,P2/1800,P3/2000
  -space=7F -crc=2FFE=1000-17FF,2000-27FF
  -output=out.mot=1000-2FFF

- crc オプション：-crc=2FFE=1000-2FFD,2000-27FF
  - 0x1000 ～ 0x17FF と 0x2000 ～ 0x27FF の２つの領域に対して CRC 演算を行い，その結果を 0x2FFE 番地に出力します。
  - CRC 演算は計算対象として，連続していない複数の計算範囲を指定できます。
- space オプション：-space=7F
  - 指定された計算範囲の空き領域は space オプションの値（0x7F）で計算されます。
- output オプション：-output=out.mot=1000-2FFF
  - space オプションが指定されているため，空き領域は「out.mot」ファイルに出力されます。空き領域は 0x7F で充填されます。

    注 1. CRC 演算の計算順は計算範囲の指定順ではありません。下位アドレスから上位アドレスの順に計算されます。

    注 2. crc オプションと space オプションを同時に指定する場合，space オプションに random または２バイト以上の値を指定することは出来ません。１バイトのデータを指定してください。

- rlink *.obj -form=stype -start=P1,P2/1000,P3/2000
  -crc=1FFE=1000-1FFD,2000-2FFF
  -output=flmem.mot=1000-1FFF

- crc オプション：-crc=1FFE=1000-1FFD,2000-2FFF
  - 0x1000 ～ 0x1FFD と 0x2000 ～ 0x2FFF の領域に対して CRC 演算を行い，その結果を 0x1FFE 番地に出力します。
  - 計算範囲にある空き領域は space オプションが指定されていない場合，space=0xFF が指定されていると仮定して，CRC 演算を行います。
- output オプション：-output=flmem.mot=1000-1FFF
  - space オプションが指定されていないため，空きの領域は「flmem.mot」ファイルに出力されません。
  - CRC 演算は，空き領域では 0xFF で計算を行いますが，0xFF を埋めることはありません。

［備考］

- 複数のアブソリュートファイル入力時は，本オプションは無効です。
- 出力形式が form={hexadecimal | stype} の場合に有効です。
- space オプションが指定されていない場合で，計算範囲に出力されない空き領域があるとき，空き領域には 0xFF が設定されているものとして CRC の計算が行われます。
- CRC 演算の計算範囲にオーバーレイ指定されている領域が含まれる場合はエラーになります。
- サンプルコード
- crc オプションで計算された CRC 演算結果を比較するためのサンプルコードです。
- サンプルコードのプログラムは，rlink の CRC 演算結果と一致します。
- 多項式 CRC-CCITT の場合（初期値 0xFFFF, MSB ファースト , 反転出力）

```
typedef  unsigned   char    uint8_t;
typedef  unsigned   short   uint16_t;
typedef  unsigned   long    uint32_t;
uint16_t CRC_CCITT(uint8_t *pData, uint32_t iSize)
{
    uint32_t  ui32_i;
    uint8_t       *pui8_Data;
    uint16_t  ui16_CRC = 0xFFFFu;
    pui8_Data = (uint8_t *)pData;
    for(ui32_i = 0; ui32_i < iSize; ui32_i++)
    {
        ui16_CRC  = (uint16_t)((ui16_CRC >> 8u) |
        ((uint16_t)((uint32_t)ui16_CRC << 8u)));
        ui16_CRC ^= pui8_Data[ui32_i];
        ui16_CRC ^= (uint16_t)((ui16_CRC & 0xFFu) >> 4u);
        ui16_CRC ^= (uint16_t) ((ui16_CRC << 8u) << 4u);
        ui16_CRC ^= (uint16_t)(((ui16_CRC & 0xFFu) << 4u) << 1u);
    }
    ui16_CRC = (uint16_t)( 0x0000FFFFul &
    ((uint32_t)~(uint32_t)ui16_CRC) );
    return ui16_CRC;
}
```

- 多項式 CRC-16 の場合（初期値 0x0000, LSB ファースト , 非反転出力）

```
#define  POLYNOMIAL  0xa001      // 生成多項式 CRC-16
typedef  unsigned    char     uint8_t;
typedef  unsigned    short    uint16_t;
typedef  unsigned    long     uint32_t;
uint16_t CRC16(uint8_t *pData, uint32_t iSize)
{
    uint16_t crcdData = (uint16_t)0;
    uint32_t data = 0;
    uint32_t i,cycLoop;
    for(i=0;i<iSize;i++){
        data = (uint32_t)pData[i];
        crcdData = crcdData ^ data;
        for (cycLoop = 0; cycLoop < 8; cycLoop++) {
            if (crcdData & 1) {
                crcdData = (crcdData >> 1) ^ POLYNOMIAL;
            } else {
                crcdData = crcdData >> 1;
            }
        }
    }
    return crcdData;
}
```

-padding

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-padding
```

- 省略時解釈
  なし

［詳細説明］

- セクションサイズが，セクションのアライメントの倍数となるように，セクション終端にデータを埋め込みます。
- 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］カテゴリの［セクション終端にパディング・データを埋め込む］

［例］

```
-start=P,C/0 -padding
```

P セクションのアライメント :4 バイト
P セクションのサイズ :0x06 バイト
C セクションのアライメント :1 バイト
C セクションのサイズ :0x03 バイト
の場合，
P セクションに 2 バイトのパディングデータを埋め込んで，サイズを 0x08 バイトにしてリンクする。

```
-start=P/0,C/7 -padding
```

P セクションのアライメント :4 バイト
P セクションのサイズ :0x06 バイト
C セクションのアライメント :1 バイト
C セクションのサイズ :0x03 バイト
の場合，
P セクションに 2 バイトのパディングデータを埋め込んで，サイズを 0x08 バイトにしてリンクすると，C セクションと重複してしまうため，E0562231 エラーを出力する。

［備考］

- 生成するパディングデータの値は 0x00 です。
- 絶対アドレスセクションには，パディングを行いませんので，絶対アドレスセクションはユーザにてサイズを調整してください。

-vectn

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-vectn = ＜サブオプション＞[,...]
         ＜サブオプション＞ :  ＜ベクタ番号＞ = {＜シンボル＞ | ＜アドレス＞}
```

- 省略時解釈
  なし

［詳細説明］

- 可変ベクタテーブルセクションの特定ベクタ番号に対して，オプションで指定されたアドレスを設定します。
- 本オプションを使用した場合，ソース上に割り込み関数記述がなくても，可変ベクタテーブルセクションを作成し，テーブルへアドレスを設定します。
- ＜ベクタ番号＞は，10 進数で 0 ～ 255 の範囲で指定してください。
- ＜シンボル＞は，対象関数の外部名で指定してください。
- ＜アドレス＞は，指定アドレスを 16 進数で指定してください。
- 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［出力］カテゴリの［特定ベクタ番号のアドレス］

［例］

```
-vectn=30=_f1,31=0000F100          ; ベクタ番号 30 番に _f1 のアドレスを，
                                   ; ベクタ番号 31 番に 0x0f100 を設定します
```

［備考］

ユーザが可変ベクタテーブルセクションをソースプログラムで作成している場合，可変ベクタテーブルの自動生成は行わないため，本オプションは無効になります。

-vect

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-vect={< シンボル >|< アドレス >}
```

- 省略時解釈
  なし

［詳細説明］

- 可変ベクタテーブルセクションで，アドレス未設定のベクタ番号に対してオプション指定のアドレスを設定します。
- 本オプションを使用した場合，ソース上の割り込み関数記述がなくても，可変ベクタテーブルセクションをリンカが作成し，テーブルへアドレスを設定します。
- < シンボル > は，対象関数の外部名を記述してください。
- < アドレス > は，設定するアドレスを 16 進数表記で記述してください。
- 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［出力］カテゴリの［可変ベクタの空き領域のアドレス］

［例］

-

［備考］

ユーザが可変ベクタテーブルセクションをソースプログラムで作成している場合，可変ベクタテーブルの自動生成は行わないため，本オプションは無効になります。
{< シンボル >|< アドレス >} の記述で，先頭を 0 と記述したものはすべてアドレスとして判断します。

-jump_entries_for_pic

＜最適化リンケージエディタ（rlink）・オプション / 出力オプション＞

［指定形式］

```
-jump_entries_for_pic = < セクション名 >[,...]
```

- 省略時解釈
  なし

［詳細説明］

- 指定セクション内の外部定義シンボルへ分岐するジャンプテーブルのアセンブラソースを出力します。
- ファイル名は，< 出力ファイル >.jmp 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。
- ［リンク・オプション］タブの［出力］カテゴリの［ジャンプ・テーブルを出力する］，［外部定義シンボルへ分岐するジャンプ・テーブルを出力するセクション］

［例］

- セクション sct2,sct3 の外部定義シンボルへ分岐するジャンプテーブルを test.jmp に出力します。

```
jump_entries_for_pic=sct2,sct3
output=test.abs
```

- [test.jmp の出力例]

```
;OPTIMIZING LINKAGE EDITOR GENERATED FILE 2009.07.19
            .glb _func01
            .glb _func02
            .SECTION    P,CODE
_func01:
            MOV.L       #1000H,R14
            JMP         R14
_func02:
            MOV.L       #2000H,R14
            JMP         R14
            .END
```

［備考］

- form={object | relocate | library} または strip 指定時，本オプションは無効です。
- 生成するジャンプテーブルは，P セクションへ出力します。
- セクション名に指定できるセクション種別は，プログラムセクションのみです。

リストオプション

＜最適化リンケージエディタ（rlink）・オプション / リストオプション＞

リストオプションには，次のものがあります。

- -list
- -show

-list

＜最適化リンケージエディタ（rlink）・オプション / リストオプション＞

［指定形式］

```
-list [= <ファイル名>]
```

- 省略時解釈
  なし

［詳細説明］

- リストファイル出力およびリストファイル名を指定します。
- リストファイル名を指定しない場合には，出力（または先頭出力）ファイルと同じファイル名で，拡張子がform=library または extract 指定時「lbp」，それ以外のとき「map」のリストファイルが作成されます。
- 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［リスト］カテゴリの［リンケージ・リスト・ファイルを出力する］
- ［ライブラリアン・オプション］タブの［リスト］カテゴリの［リンケージ・リスト・ファイルを出力する］

［例］

-

［備考］

-

-show

＜最適化リンケージエディタ（rlink）・オプション / リストオプション＞

［指定形式］

```
-show[= <sub>[,…]]
        <sub> : { symbol | reference | section | xreference | total_size | vector |
all }
```

- 省略時解釈
  なし

［詳細説明］

- リストの出力内容を指定します。
- サブオプションの一覧を表 B-16 に示します。
- 各リストの具体例については「リンケージリスト」,「ライブラリリスト」を参照してください。

表 B.16 **show** オプションのサブオプション一覧

No	出力形式	サブオプション名	意味
1	form=library	symbol	モジュール内シンボル名一覧を出力します。
		reference	指定できません。
		section	モジュール内セクション一覧を出力します。
		xreference	指定できません。
		total_size	指定できません。
		vector	指定できません。
		all	モジュール内シンボル名，セクション一覧を出力します。
2	form=library 以外	symbol	シンボルアドレス，サイズ，種別，最適化内容を出力します。
		reference	シンボルの参照回数を出力します。（form=relocate の時は指定できません）
		section	指定できません。
		xreference	クロスリファレンス情報を出力します。
		total_size	ROM 配置対象，RAM 配置対象ごとに，セクションの合計サイズを表示します。
		vector	ベクタ情報を出力します。（form=relocatable のときは指定できません）
		all	show=symbol,xreferrence,total_size 指定時と同内容を出力します。（form=rel） show=symbol,reference,xreference ,total_size 指定時と同内容を出力します。（form=abs） show=symbol,reference,xreference,total_size 指定時と同内容を出力します。（form=hex/stype/bin） form=obj のときは指定できません。

- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［リンク・オプション］タブの［リスト］カテゴリの［シンボル情報を出力する］，［シンボルの参照回数を出力する］，［クロスリファレンス情報を出力する］，［セクションの合計サイズを表示する］，［ベクタ情報を出力する］
- ［ライブラリアン・オプション］タブの［リスト］カテゴリの［シンボル情報を出力する］，［モジュール内セクション一覧を出力する］，［クロスリファレンス情報を出力する］，［セクションの合計サイズを表示する］，［ベクタ情報を出力する］

［例］

-

［備考］

- オプション form とオプション show および show=all で有効 / 無効になる組み合わせは以下のようになります。

		Symbol	Reference	Section	Xreference	Vector	Total_size
form=abs	show のみ	有効	有効	無効	無効	無効	無効
	show=all	有効	有効	無効	有効	有効	有効
form=lib	show のみ	有効	無効	有効	無効	無効	無効
	show=all	有効	無効	有効	無効	無効	無効
form=rel	show のみ	有効	無効	無効	無効	無効	無効
	show=all	有効	無効	無効	有効注	無効	有効
form=obj	show のみ	有効	有効	無効	無効	無効	無効
	show=all	無効	無効	無効	無効	無効	無効
form=hex/ bin/sty	show のみ	有効	有効	無効	無効	無効	無効
	show=all	有効	有効	無効	有効	有効注	有効注

注　入力ファイルが absolute 形式の場合は無効です。
クロスリファレンス情報の出力に関しては，下記制限があります。

- 入力ファイルが absolute 形式の場合，参照側アドレスの情報は出力されません。
- 同一ファイル内の，定数シンボルへの参照に関する情報は出力されません。
- コンパイル時に最適化が有効で，直下の関数を呼び出す場合についての情報は出力されません。
- 外部変数アクセス最適化が有効な場合，ベースとなるシンボルを除いて，変数の参照情報は出力されません。
- show=total_size で表示する情報は，別オプション total_size での表示内容と同じです。
- show=reference が有効な場合に，#pragma address で指定された変数の参照回数が 0 として出力されます。
- extract が指定されている場合、サブオプションを指定することはできません。

最適化オプション

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

最適化オプションには，次のものがあります。

- -optimize
- -nooptimize
- -samesize
- -symbol_forbid
- -samecode_forbid
- -section_forbid
- -absolute_forbid

-optimize

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-optimize[= <サブオプション>[,…]]
<サブオプション> : {SYmbol_delete | SAMe_code | SHort_format | Branch | SPeed | SAFe}
```

- 省略時解釈
  optimize です。

［詳細説明］

- コンパイル，アセンブル時に goptimize オプションを指定したファイルに対して最適化を行います。
- サブオプションがない場合は、すべての最適化を実行します。
  -optimize=symbol_delete,same_code,short_format,branch と同じ意味を持ちます。
- -optimize=speed は、オブジェクトスピード低下を招く可能性のある最適化以外を実行します。
  -optimize=symbol_delete,short_format,branch と同じ意味を持ちます。
- -optimize=safe は、変数や関数の属性によって制限される可能性のある最適化以外を実行します。
  -optimize=short_format,branch と同じ意味を持ちます。
- その他のサブオプションが意味する最適化の内容は、次の表の通りです。

表 B.17 **optimize** オプションのサブオプション一覧

サブオプション	意 味	最適化対象プログラム[注1]	
		RXC	RXA
symbol_delete	1 度も参照のない変数 / 関数を削除します。必ずコンパイル 時に #pragma entry を指定するか，rlink で entry オプションを指定してください。	○	×
same_code	複数の同一命令列をサブルーチン化します。	○	×
short_format	ディスプレースメント / イミディエートのコードサイズを短縮可能な場合，コードサイズがより小さくなる命令に置き換えます。	○	○
branch	プログラムの配置情報に基づいて，分岐命令サイズを最適化します。 symbol_delete 以外の他の最適化項目を実行すると，指定の有無に関わらず必ず実行します。	○	○

注意 1. RXC: RX ファミリ用 C/C++ プログラム，RXA: RX ファミリ用アセンブリプログラム

- 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［最適化］カテゴリの［最適化方法］，［未参照シンボルを削除する］，［複数の同一命令列をサブルーチン化する］，［コードサイズがより小さくなる命令に置き換える］，［分岐命令サイズを最適化する］

［備考］

- form=｛object|relocate|library｝または strip 指定時，本オプションは無効です。
- プログラム内に #pragma entry で実行開始関数を指定，または実行開始アドレス（entry）を指定していない場合，optimize=symbol_delete は無効になります。

-nooptimize

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-nooptimize
```

- 省略時解釈
  optimize です。

［詳細説明］

- リンク時最適化を行いません。
- 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［最適化］カテゴリの［最適化方法］

-samesize

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-samesize = <サイズ>
```

- 省略時解釈
  samesize=1E です。

［詳細説明］

- 共通コード統合最適化（optimize=same_code）で，最適化対象となる最小コードサイズを指定します。8 ～ 7FFF までの 16 進数で指定してください。
- 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［最適化］カテゴリの［最小コードサイズ］

［例］

-

［備考］

optimize=same_code の指定がないとき，本オプションは無効です。

-symbol_forbid

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-symbol_forbid = < シンボル名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- 未参照シンボル削除最適化を抑止します。
- C/C++ 変数名，C 関数名はプログラム中での定義名先頭に _ を付加します。C++ 関数の場合は，引数列を含めたプログラム中の定義名をダブルクォーテーションで囲んで指定します。ただし，引数が void の場合は，" 関数名 ()" で指定します。
- 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［最適化］カテゴリの［最適化による削除を抑止する未参照シンボル］

［例］

-

［備考］

最適化を使用しないリンク処理では，本オプションは無効です。

-samecode_forbid

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-samecode_forbid = < 関数名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- 共通コード統合最適化を抑止します。
- C/C++ 変数名，C 関数名はプログラム中での定義名先頭に _ を付加します。C++ 関数の場合は，引数列を含めたプログラム中の定義名をダブルクォーテーションで囲んで指定します。ただし，引数が void の場合は，" 関数名 ()" で指定します。
- 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［最適化］カテゴリの［最適化で統合を抑止する共通コード］

［例］

-

［備考］

最適化を使用しないリンク処理では，本オプションは無効です。

-section_forbid

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-section_forbid = <sub>[,…]
                  <sub>: [< ファイル名 >|< モジュール名 >](< セクション名 >[,…])
```

- 省略時解釈
  なし

［詳細説明］

- -section_forbid のサブオプションで指定したセクションの最適化を抑止します。入力ファイル名，もしくはライブラリモジュール名を同時に指定することで，最適化抑止対象をセクション全体だけでなく，指定したファイルに限定することも可能です。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［リンク・オプション］タブの［最適化］カテゴリの［最適化を抑止するセクション］

［例］

-

［備考］

- 最適化を使用しないリンク処理では，本オプションは無効です。
- パスを記述した入力ファイルを最適化抑止する場合，section_forbid オプションではファイル名にパスを記述してください。

-absolute_forbid

＜最適化リンケージエディタ（rlink）・オプション / 最適化オプション＞

［指定形式］

```
-absolute_forbid = <アドレス>[+ <サイズ>][,…]
```

- 省略時解釈
  なし

［詳細説明］

- -absolute_forbid のサブオプションで指定したアドレス＋サイズの範囲の最適化を抑止します。
- 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［最適化］カテゴリの［最適化を抑止するアドレス範囲］

［例］

-

［備考］

- 最適化を使用しないリンク処理では，本オプションは無効です。

セクションオプション

＜最適化リンケージエディタ（rlink）・オプション / セクションオプション＞
セクションオプションには，次のものがあります。

- -start
- -fsymbol
- -aligned_section

-start

＜最適化リンケージエディタ（rlink）・オプション / セクションオプション＞

［指定形式］

```
-start= <sub>[,...]
        <sub> : [(]< セクション名 > [{: | ,} < セクション名 >[,...] ][)][,...] [/< アドレス >]
```

- 省略時解釈
  セクションを 0 番地から割り付けます。

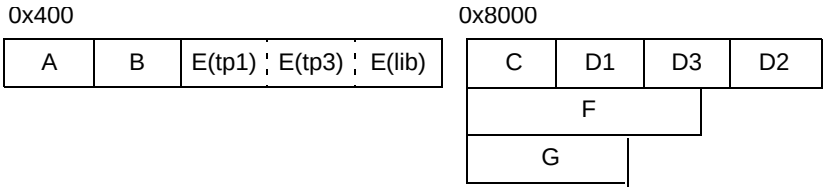
［詳細説明］

- セクションの開始アドレスを指定します。アドレスは 16 進数で指定してください。
- セクション名はワイルドカード（*）も指定できます。ワイルドカードで指定したセクションはオブジェクトファイルの入力順に展開します。
- セクションをコロン（:）で区切ることにより，複数のセクションを同一アドレスに割り付ける（セクションオーバーレイ配置）ことが可能です。
- 同一アドレスに割り付け指定したセクション間は，指定順に割り付けます。
- また，丸括弧 "()" で囲むことにより，オーバーレイ配置する対象セクションを変更できます。
- 同一セクション内オブジェクトは，入力ファイルの指定順，入力ライブラリの指定順に割り付けます。
- アドレスの指定がない場合は，0 番地から割り付けます。
- start オプションで指定していないセクションは，最終割り付けアドレスに続いて割り付けます。
- 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［セクション］カテゴリの［セクションの開始アドレス］

［例］

下記順番でオブジェクトを入力する場合のセクション配置を例に示します。
（括弧内は各オブジェクトが持つセクション）
tp1.obj(A,D1,E) -> tp2.obj(B,D3,F) -> tp3.obj(C,D2,E,G) -> lib.lib(E)

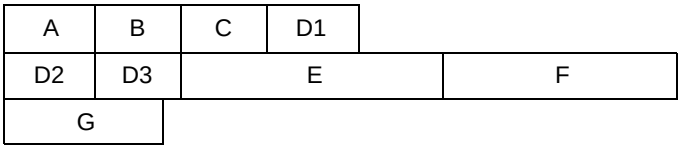
- -start=A,B,E/400,C,D*:F:G/8000



- ":" で区切った C,F,G セクションは，同一アドレスに割り付きます。
- ワイルドカードで記述したセクション（ここでは D で始まる名前のセクション）は，
  入力した順番で割り付きます。
- 同名セクション内（ここでは E セクション）は，入力したオブジェクトから順番に割り付きます。
- ライブラリ入力による同名セクション（ここでは E セクション）は，入力オブジェクトの次に割り付きます。

- -start=A,B,C,D1:D2,D3,E,F:G/400

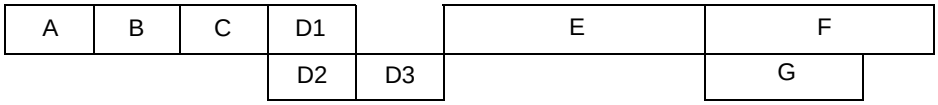
0x400



- ":" で区切った直後のセクション（この例の場合は A,D2,G）を先頭として，それぞれ先頭が同一アドレスに割り付きます。

- -start=A,B,C,(D1:D2,D3),E,(F:G)/400

0x400



- "()" で同一アドレス配置を括った場合，"()" の直前のセクション（この例の場合は C,E）の直後を先頭として，"()" 内の同一アドレス配置が行われます。
- "()" の直後のセクション（この場合 E）は，"()" 内の最後尾のセクションの直後に続けて配置されます。

［備考］

- form={object | relocate | library} または strip 指定時，本オプションは無効です。
- 括弧 "()" は，ネストして記述することはできません。
- 括弧 "()" 内では，少なくともひとつはコロン ":" の記述が必要です。コロン ":" を記述しない場合には，括弧 "()" は記述できません。
- 括弧 "()" を記述した場合，"()" 外にコロン ":" を記述することはできません。
- 括弧 "()" を使用して本オプションを記述した場合，リンカの最適化機能は無効になります。

-fsymbol

＜最適化リンケージエディタ（rlink）・オプション / セクションオプション＞

［指定形式］

```
-fsymbol = < セクション名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- 指定したセクション内外部定義シンボルをアセンブラ制御命令形式でファイルに出力します。
  ファイル名は，< 出力ファイル >.fsy 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。
- ［リンク・オプション］タブの［セクション］カテゴリの［外部定義シンボルをファイル出力するセクション］

［例］

- セクション sct2,sct3 の外部定義シンボルを test.fsy に出力します。

```
fsymbol=sct2,sct3
output=test.abs
```

- [test.fsy の出力例]

```
;RENESAS OPTIMIZING LINKER GENERATED FILE 2012.07.19
;fsymbol = sct2, sct3
;SECTION NAME = sct2
 .glb _f
_f .equ 00000000h
 .glb _g
_g .equ 00000016h
;SECTION NAME = sct3
 .glb _main
_main .equ 00000020h
 .end
```

［備考］

- form={object | relocate | library} または strip 指定時，本オプションは無効です。

-aligned_section

＜最適化リンケージエディタ（rlink）・オプション / セクションオプション＞

［指定形式］

```
-aligned_section = < セクション名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- 指定セクションのアライメント数を 16 byte に変更します。
- 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［セクション］カテゴリの［セクション・アライメント］

［備考］

- form={object | relocate | library} および extract，strip 指定時，本オプションは無効です。

ベリファイオプション

＜最適化リンケージエディタ（rlink）・オプション / ベリファイオプション＞
ベリファイオプションには，次のものがあります。

- -cpu
- -contiguous_section

-cpu

＜最適化リンケージエディタ（rlink）・オプション / ベリファイオプション＞

［指定形式］

```
-cpu = { ＜メモリ種別＞ = ＜アドレス範囲＞[,…] | STRIDE}
         ＜メモリ種別＞ = { ROm | RAm | FIX }
         ＜アドレス範囲＞ : ＜先頭アドレス＞ - ＜終了アドレス＞
```

- 省略時解釈
  なし

［詳細説明］

- cpu=stride 未指定時は，セクションの割り付けアドレスに対して，アドレス範囲に入らない場合は，エラーを出力します。
- cpu=stride 指定時は，セクションの割り付けアドレスに対して，アドレス範囲に入らない場合は，次の同メモリ種別に配置，または，分割して配置します。
- 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［ベリファイ］カテゴリの［セクションの割り付けアドレスをチェックする］，［メモリ種別のアドレス範囲］，［次の同メモリ種別に配置、または、分割して配置する］

［例］

- サブオプション stride を指定しない場合

```
start=D1,D2/100
cpu=ROM=100-1FF,RAM=200-2FF
```

- D1 が 100-1FF，D2 が 200-2FF の範囲に収まるとき，正常終了します。収まらないときエラーを出力します。

- サブオプション stride を指定した場合

```
start=D1,D2/100
cpu=ROM=100-1FF,RAM=200-2FF,ROM=300-3FF
cpu=stride
```

- D1,D2 が ROM 属性の領域に（セクションを分割して / 分割しないで）収まるとき，正常終了します。セクションを分割しても収まらないときリンクエラーになります。
- セクション割り付けが可能なアドレス範囲を 16 進数で指定してください。ROM/RAM の属性は，モジュール間最適化で使用します。
- メモリ種別 “FIX” には，アドレス固定の領域（I/O エリア等）を指定します。
- メモリ種別 “FIX” と，それ以外のメモリ種別のアドレス範囲が重複した場合は，メモリ種別 “FIX” を有効とします。
- サブオプション stride は，メモリ種別が，ROM または RAM で，アドレス範囲にセクションが収まらなかった場合に，セクションを分割して同じメモリ種別の領域に割り付けます。
- サブオプション stride で，セクションを分割する単位は，モジュール単位になります。

```
cpu=ROM=0-FFFF,RAM=10000-1FFFF
```

- セクションアドレスが，0-FFFF または 10000-1FFFF の間に入っているかチェックします。
- モジュール間最適化では，異なる属性間でのオブジェクトの移動は行いません。

```
cpu=ROM=100-1FF,ROM=400-4FF,RAM=500-5FF
cpu=stride
```

- セクションアドレスが，100-1FF の間に収まらなかった場合に，セクションをモジュール単位で分割して 400-4FF に割り付けます。

［備考］

- form={object | relocate | library} または strip 指定時，本オプションは無効です。
- cpu=stride および memory=low 指定時，無効になります。
- cpu=stride を指定し，B セクションが分割された場合，0 初期化するための情報として 8 バイト × 分割数分だけ C$BSEC セクションのサイズが増加します。

-contiguous_section

＜最適化リンケージエディタ（rlink）・オプション / ベリファイオプション＞

［指定形式］

```
-contiguous_section=< セクション名 >[,...]
```

- 省略時解釈
  なし

［詳細説明］

- cpu=stride が有効なときに，セクションを分割せずに同じメモリ種別の割り付け可能なアドレス領域に割り付けるセク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。
- ［リンク・オプション］タブの［ベリファイ］カテゴリの［分割対象外セクション］

［例］

```
start=P,PA,PB/100
cpu=ROM=100-1FF,ROM=300-3FF,ROM=500-5FF
cpu=stride
contiguous_section=PA
```

- セクション P を 100 番地に割り付けます。
- contiguous_section 指定したセクション PA が，1FF 番地までに割り付けることができない場合，セクション PA を分割せずに，300 番地から割り付けます。
- contiguous_section 指定してないセクション PB が，3FF 番地までに割り付けることができない場合，セクション PB を分割して，500 番地から割り付けます。

［備考］

- cpu オプションのサブオプションの stride が無効なとき，本オプションは無効です。

その他オプション

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞
その他オプションには，次のものがあります。

- -s9
- -stack
- -compress
- -nocompress
- -memory
- -rename
- -delete
- -replace
- -extract
- -strip
- -change_message
- -hide
- -total_size

-s9

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-s9
```

- 省略時解釈
  なし

［詳細説明］

- エントリアドレスが 0x10000 を超える場合でも，S9 レコードを終端に出力します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［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［S9 レコードを終端に出力する］

［例］

-

［備考］

- form=stype 指定がないとき，本オプションは無効です。

-stack

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-stack
```

- 省略時解釈
  なし

［詳細説明］

- スタック使用量情報ファイルを出力します。
- ファイル名は，< 出力ファイル名 >.sni 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。
- ［リンク・オプション］タブの［その他］カテゴリの［スタック使用量情報ファイルを出力する］

［例］

-

［備考］

- form={object | relocate | library} および strip 指定時，本オプションは無効です。

-compress

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-compress
```

- 省略時解釈
  デバッグ情報を圧縮しません。

［詳細説明］

- デバッグ情報を圧縮します。
- デバッグ情報を圧縮すると，デバッガのロード速度が速くなります。
- 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［デバッグ情報を圧縮する］

［例］

-

［備考］

- form={object | relocate | library | hexadecimal | stype | binary} または strip オプションを指定した場合，compress オプションは無効です。

-nocompress

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-nocompress
```

- 省略時解釈
  デバッグ情報を圧縮しません。

［詳細説明］

- デバッグ情報を圧縮しません。
- nocompress オプションを指定すると，compress オプションを指定したときに比べてリンク時間が短くなります。
- 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［デバッグ情報を圧縮する］

［例］

-

［備考］

-

-memory

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-memory = [ High | Low ]
```

- 省略時解釈
  memory=high です。

［詳細説明］

- リンク時に使用するメモリ量を指定します。
- memory=high オプションを指定した場合，従来通りの処理を行います。
- memory=low オプションを指定した場合，リンク時に必要な情報のロードを細かく行うことにより，使用するメモリ量の削減を行います。ファイルアクセスの頻度が増えるため，メモリ使用量が実装メモリを超えない状況では memory=high オプション指定より処理が遅くなります。
- 大規模なプロジェクトをリンクした際，最適化リンケージエディタのメモリ使用量が稼動マシンの実装メモリ量を越えてしまい，動作が遅くなっているような場合には memory=low オプション指定をお試しください。
- 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［メモリ使用量を削減する］
- ［ライブラリアン・オプション］タブの［その他］カテゴリの［メモリ使用量を削減する］

［備考］

- 下記オプションを指定した場合、-memory=low オプション指定は無効となります。
- form=absolute,hexadecimal,stype,binary 指定時
  compress,delete,rename,map,stack,cpu=stride
  list と show[={reference | xreference}] を同時指定
- form=library 指定時
  delete,rename,extract,hide,replace
- form=object,relocate 指定時
  extract
- optimize を指定時

また，入力ファイルや出力ファイル形式によっても無効となる組み合わせがあります。

-rename

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-rename = <サブオプション>[,…]
          <サブオプション> : { [<ファイル>](<名前>=<名前>[,…])
                             | [<モジュール>](<名前>=<名前>[,…])}
```

- 省略時解釈
  なし

［詳細説明］

- 外部シンボル名，セクション名を変更します。
- 特定のファイルまたは特定のライブラリ内モジュールに含まれるシンボル名，セクション名を変更することもできます。
- C/C++ 変数名の場合，プログラム中での定義名先頭に _ を付加します。
- 関数名を変更した場合の動作は保証できません。
- 指定した名前がセクション，シンボルの両方に存在した場合，シンボル名を優先します。
- 同一ファイル名，モジュール名が複数存在する場合は，先に入力した方を優先します。

［例］

```
rename=(_sym1=data)                ;_sym1 を data に変更します。
rename=lib1(P=P1)                  ; ライブラリモジュール lib1 内の P セクションを
                                   ;P1 セクションに変更します。
```

［備考］

- extract または strip 指定時，本オプションは無効です。
- form=absolute 指定時，入力されたライブラリのセクション名を変更することができません。
- コンパイル・オプション -merge_files と本オプションを組み合わせて使用した場合，動作は保証しません。

-delete

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-delete = <サブオプション>[,…]
          <サブオプション> : { [<ファイル>](<名前>[,…]) | <モジュール> }
```

- 省略時解釈
  なし

［詳細説明］

- 外部シンボル名またはライブラリモジュールを削除します。
- 特定のファイルに含まれるシンボル名，モジュールを削除することもできます。
- C/C++ 変数名，C 関数名はプログラム中での定義名先頭に _ を付加します。C++ 関数の場合は，引数列を含めたプログラム中の定義名をダブルクォーテーションで囲んで指定します。ただし，引数が void の場合は，" 関数名 ()" で指定します。同一ファイル名が複数存在する場合は，先に入力した方を優先します。
- 本オプションで，シンボル名削除を指定した場合，オブジェクトは削除されず，属性が内部シンボルに変更されます。

［例］

```
delete=(_sym1)           ; 全ファイル中のシンボル名 _sym1 を削除します。
delete=file1.obj(_sym2)  ; file1.obj 内のシンボル名 _sym2 を削除します。
```

［備考］

- extract または strip 指定時，本オプションは無効です。
- form=library のときに，モジュールを削除できます。
- form={absolute|relocate|hexadecimal|stype|binary} のときに，外部シンボルを削除できます。
- コンパイル・オプション -merge_files と本オプションを組み合わせて使用した場合，動作は保証しません。

-replace

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-replace = ＜サブオプション＞[,…]
           ＜サブオプション＞ : ＜ファイル名＞[(＜モジュール名＞[,…])]
```

- 省略時解釈
  なし

［詳細説明］

- ライブラリモジュールを置換します。
- 指定したファイルまたはライブラリモジュールと library オプションで指定したライブラリ内同名モジュールを置換します。

［例］

```
replace=file1.obj       ; モジュール file1 と file1.obj を置換します。
replace=lib1.lib(mdl1) ; モジュール mdl1 とライブラリファイル lib1.lib 内モジュール mdl1 を置換します。
```

［備考］

- form={object | relocate | absolute | hexadecimal | stype | binary} および extract，strip 指定時，本オプションは無効です。
- コンパイル・オプション -merge_files と本オプションを組み合わせて使用した場合，動作は保証しません。

-extract

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-extract = < モジュール名 >[,…]
```

- 省略時解釈
  なし

［詳細説明］

- ライブラリモジュールを抽出します。
- 指定したライブラリモジュールを library オプションで指定したライブラリファイルから抽出します。

［例］

```
extract=file1                ; モジュール file1 を抽出します。
```

［備考］

- form={absolute | hexadecimal | stype | binary} および strip 指定時，本オプションは無効です。
- form=library 指定時，モジュールを削除できます。
- form={absolute|relocate|hexadecimal|stype|binary} 指定時，外部シンボルを削除できます。

-strip

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-strip
```

- 省略時解釈
  なし

［詳細説明］

- アブソリュートファイル，ライブラリファイルのデバッグ情報を削除します。
- strip オプション指定時は，入力ファイルと出力ファイルは 1 対 1 対応になります。

［例］

```
input=file1.abs file2.abs file3.abs
strip
```

- file1.abs, file2.abs, file3.abs のデバッグ情報を削除し，それぞれ file1.abs, file2.abs, file3.abs に出力します。デバッグ情報削除前のファイルは，file1.abk, file2.abk, file3.abk にバックアップします。

［備考］

- form={object | relocate | hexadecimal | stype | binary} 指定時，本オプションは無効です。

-change_message

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-change_message = ＜サブオプション＞[,…]
                  ＜サブオプション＞ : ＜エラーレベル＞[=＜エラー番号＞[-＜エラー番号＞][,…]]
                  ＜エラーレベル＞   : {Information | Warning | Error}
```

- 省略時解釈
  なし

［詳細説明］

- インフォメーション，ウォーニング，エラーレベルのメッセージレベルを変更します。
- メッセージ出力時の実行継続 / 中断を変更できます。
- エラー番号を指定すると，指定した番号のメッセージの種別を変更します。
- また，ハイフン（-）を使用して，メッセージ番号の範囲を指定することもできます。
- エラー番号には，コンポーネント番号（05），発生フェーズ（6）に続けて出力される 4 桁の数値（E0562310 の場合は 2310）を指定します。
- メッセージ番号の指定を省略した場合は，すべてのメッセージの種別を指定したものに変更します。
- 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［ウォーニング・メッセージをインフォメーションレベルに変更する］，［ウォーニングレベルのエラー番号］，［インフォメーション・メッセージをウォーニングレベルに変更する］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニング・メッセージをエラーレベルに変更する］，［インフォメーション、ウォーニングレベルのエラー番号］
- ［ライブラリアン・オプション］タブの［その他］カテゴリの［ウォーニング・メッセージをインフォメーションレベルに変更する］，［ウォーニングレベルのエラー番号］，［インフォメーション・メッセージをウォーニングレベルに変更する］，［インフォメーションレベルのエラー番号］，［インフォメーション、ウォーニング・メッセージをエラーレベルに変更する］，［インフォメーション、ウォーニングレベルのエラー番号］

［例］

```
change_message=warning=2310
```

- E0562310 をウォーニングレベルに変更し、E0562310 出力時も処理を継続します。

```
change_message=error
```

- すべてのインフォメーション，ウォーニングメッセージをエラーレベルに変更します。
- メッセージを一つでも出力すると，処理を中断します。

［備考］

-

-hide

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-hide
```

- 省略時解釈
  なし

［詳細説明］

- 本オプションを指定した場合，出力ファイル内のローカルシンボル名情報を消去します。
- ローカルシンボルに関する名前の情報が消去されますので，バイナリエディタなどでファイルを開いてもローカルシンボル名は確認できなくなります。生成されるファイルの動作への影響は一切ありません。
- ローカルシンボル名を機密扱いにしたい場合などに本オプションを指定してください。
- 秘匿対象となるシンボルの種類を以下に挙げます。
- ソースファイル：static 型修飾子を指定した変数名，関数名など
- ソースファイル：goto 文のラベル名
- アセンブリソース：外部定義（参照）シンボル宣言していないシンボル名
- * エントリ関数名は秘匿対象になりません。
- 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［ローカルシンボル名情報を消去する］
- ［ライブラリアン・オプション］タブの［その他］カテゴリの［ローカルシンボル名情報を消去する］

［例］

- ソースファイルで本オプションの機能が有効となる記述の例を以下に示します。

```
int g1;
int g2=1;
const int g3=3;
static int s1;           //<--- static 変数名は秘匿対象
static int s2=1;         //<--- static 変数名は秘匿対象
static const int s3=2;   //<--- static 変数名は秘匿対象

static int sub1()        //<--- static 関数名は秘匿対象
{
    static int s1;       //<--- static 変数名は秘匿対象
    int l1;

    s1 = l1; l1 = s1;
    return(l1);
}

int main()
{
    sub1();
    if (g1==1)
        goto L1;
    g2=2;
L1:                      //<--- goto 文のラベル名は秘匿対象
    return(0);
}
```

［備考］

- 本オプションは出力ファイル形式が absolute,relocate,library の場合のみ有効です。
- コンパイル，アセンブル時に goptimize オプションを指定したファイルを入力する場合，出力ファイル形式が relocate,library の場合は本オプションを指定できません。
- 外部変数アクセス最適化を行う状況で本オプションを指定する場合は，一度目のリンク時には指定せず，二度目のリンク時にのみ本オプションを指定してください。
- デバッグ情報内のシンボル名は，本オプションを指定しても削除されません。

-total_size

＜最適化リンケージエディタ（rlink）・オプション / その他オプション＞

［指定形式］

```
-total_size
```

- 省略時解釈
  なし

［詳細説明］

- リンク後のセクションの合計サイズを，標準出力に表示するためのオプションです。
- 下記の 3 種類のセクションに分けて，合計サイズを表示します。
- 実行可能なプログラムセクション
- プログラムセクション以外の ROM 領域配置セクション
- RAM 領域配置セクション
- 本オプションを使用することにより，ROM,RAM に配置する合計のセクションサイズを容易に認識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［合計セクション・サイズを表示する］
- ［ライブラリアン・オプション］タブの［その他］カテゴリの［合計セクション・サイズを表示する］

［例］

-

［備考］

- リンケージリストへ合計サイズを表示するには，別途 show=total_size オプションを使用する必要があります。
- ROM 化支援機能（rom オプション）対象のセクションの場合，転送元（ROM）と転送先（RAM）の両方で領域を使用するため，双方の合計サイズに対してセクションサイズを加算します。

サブコマンドファイルオプション

＜最適化リンケージエディタ（rlink）・オプション / サブコマンドファイルオプション＞

サブコマンドファイルオプションには，次のものがあります。

- -subcommand

-subcommand

＜最適化リンケージエディタ（rlink）・オプション / サブコマンドファイルオプション＞

［指定形式］

```
-subcommand = < ファイル名 >
```

- 省略時解釈
  なし

［詳細説明］

- オプションをサブコマンドファイルで指定します。
- サブコマンドファイルの書式は以下の通りです。
- < オプション > {= | △ } ［< サブオプション > ［,…］][△ &] [;< コメント >]
- オプションとサブオプションの区切りは，= の代わりに空白も指定できます。
- input オプションの場合は，サブオプション区切りに空白を指定できます。
- サブコマンドファイル内では 1 オプション / 行で指定します。
- サブオプションを 1 行に記述できない場合は，& を用いて継続指定できます。
- サブコマンドファイル中に subcommand オプションは指定できません。

［例］

- コマンドライン指定

```
rlink file1.obj -sub=test.sub file4.obj
```

- サブコマンド指定

```
input    file2.obj file3.obj      ; ここはコメントです。
library  lib1.lib, &              ; 継続行を指定します。
lib2.lib
```

- サブコマンドファイルで指定したオプション内容を，コマンドライン上のサブコマンド指定位置に展開し，実行します。
- ファイルの入力順序は，file1.obj, file2.obj, file3.obj, file4.obj になります。

残りのオプション

＜最適化リンケージエディタ（rlink）・オプション / 残りのオプション＞
残りのオプションには，次のものがあります。

- -logo
- -nologo
- -end
- -exit

-logo

＜最適化リンケージエディタ（rlink）・オプション / 残りのオプション＞

［指定形式］

```
-logo
```

- 省略時解釈
  コピーライト表示を出力します。

［詳細説明］

コピーライト表示を出力します。
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［その他］カテゴリの［コピーライト情報を表示する］
［ライブラリアン・オプション］タブの［その他］カテゴリの［コピーライト情報を表示する］

［例］

-

［備考］

-

-nologo

＜最適化リンケージエディタ（rlink）・オプション / 残りのオプション＞

［指定形式］

```
-nologo
```

- 省略時解釈
  コピーライト表示を出力します。

［詳細説明］

- コピーライト表示出力を抑止します。
- 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［コピーライト情報を表示する］
- ［ライブラリアン・オプション］タブの［その他］カテゴリの［コピーライト情報を表示する］

［例］

-

［備考］

-

-end

＜最適化リンケージエディタ（rlink）・オプション / 残りのオプション＞

［指定形式］

```
-end
```

- 省略時解釈
  なし

［詳細説明］

- END より前に指定したオプション列を実行します。リンケージ処理終了後，END 以降に指定したオプション列の入力，リンケージ処理を継続します。
- 本オプションは，コマンドライン上では指定できません。

［例］

```
input=a.obj,b.obj                    ; 処理 (1)
start=P,C,D/100,B/8000               ; 処理 (2)
output=a.abs                         ; 処理 (3)
end
input=a.abs                          ; 処理 (4)
form=stype                           ; 処理 (5)
output=a.mot                         ; 処理 (6)
```

- (1) 〜 (3) の処理を実行し，a.abs を出力します。
- その後，(4) 〜 (6) の処理を実行し，a.mot を出力します。

［備考］

-

-exit

＜最適化リンケージエディタ（rlink）・オプション / 残りのオプション＞

［指定形式］

```
-exit
```

- 省略時解釈
  なし

［詳細説明］

- オプション指定の終了を指定します。
- 本オプションは，コマンドライン上では指定できません。

［例］

- コマンドライン指定

```
rlink -sub=test.sub -nodebug
```

- test.sub

```
input=a.obj,b.obj                    ; 処理 (1)
start=P,C,D/100,B/8000               ; 処理 (2)
output=a.abs                         ; 処理 (3)
exit
```

- (1) 〜 (3) の処理を実行し，a.abs を出力します。
- Exit 実行後のコマンドライン指定の nodebug オプションは無効になります。

［備考］

-

B.1.3.4 ライブラリジェネレータ・オプション

分類 .	オプション	説明
ライブラリオプション	-head	構築対象のライブラリを指定
	-output	出力ライブラリファイル名を指定
	-nofloat	簡易入出力関数の生成
	-lang	使用可能な C 言語標準ライブラリ関数構成の選択
	-simple_stdio	機能縮小版入出力関数の生成
	-logo -nologo	コピーライトを出力 コピーライトの出力を抑止

ライブラリオプション

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

ライブラリオプションには，次のものがあります。

- -head
- -output
- -nofloat
- -lang
- -simple_stdio
- -logo
- -nologo

-head

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-head=<sub>[,…]
<sub>:{ all | runtime | ctype | math | mathf | stdarg | stdio | stdlib | string | ios |
        new | complex | cppstring | c99_complex  |  fenv | inttypes | wchar | wctype}
```

- 省略時解釈
  head=all です。

［詳細説明］

- 構築対象をヘッダファイル名で指定します。
- head=all を指定した場合，すべてのヘッダファイル名が構築対象として指定されます。
- ランタイムライブラリは常に構築対象になります。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［ライブラリ・ジェネレート・オプション］タブの［標準ライブラリ］カテゴリの［構築対象のライブラリ］，［ランタイム・ライブラリを有効にする］，［ctype.h(C89/C99) を有効にする］，［math.h(C89/C99) を有効にする］，［mathf.h(C89/C99) を有効にする］，［stdarg.h(C89/C99) を有効にする］，［stdio.h(C89/C99) を有効にする］，［stdio.h(C89/C99) を有効にする］，［stdlib.h(C89/C99) を有効にする］，［string.h(C89/C99) を有効にする］，［ios(EC++) を有効にする］，［new(EC++) を有効にする］，［complex(EC++) を有効にする］，［string(EC++) を有効にする］，［complex.h(C99) を有効にする］，［fenv.h(C99) を有効にする］，［inttypes.h(C99) を有効にする］，［wchar.h(C99) を有効にする］，［wctype.h(C99) を有効にする］

-output

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-output=< ファイル名 >
```

- 省略時解釈
  output=stdlib.lib です。

［詳細説明］

- 出力ファイル名を指定します。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［ライブラリ・ジェネレート・オプション］タブの［オブジェクト］カテゴリの［出力フォルダ］，［出力ファイル名］

-nofloat

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-nofloat
```

- 省略時解釈
  なし

［詳細説明］

- 浮動小数点変換（%f，%e，%E，%g，%G）をサポートしない，簡易入出力関数を生成します。
- 浮動小数点変換を必要としないファイル入出力を行う場合，ROM サイズを削減することができます。
  対象関数 fprintf，fscanf，printf，scanf，sprintf，sscanf，vfprintf，vprintf，vsprintf
- 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・ジェネレート・オプション］タブの［オブジェクト］カテゴリの［機能縮小版入出力関数を生成する］

［備考］

- 本オプションを指定して作成したライブラリでは，対象関数で浮動小数点数の入出力をした場合の動作は保証しません。

-lang

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-lang={c | c99}
```

- 省略時解釈
  lang=c です。

［詳細説明］

- 使用可能な C 言語標準ライブラリ関数の構成を選択します。
- lang=c を選択すると，C 言語の標準関数を，C89 規格準拠のものだけで構成し，C99 規格で拡張された関数を含めません。lang=c99 を選択すると，C 言語の標準関数を，C89 規格および C99 規格準拠の内容で構成します。
- 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・ジェネレート・オプション］タブの［標準ライブラリ］カテゴリの［ライブラリ構成］

［例］

-

［備考］

- C++,EC++ ライブラリの標準関数の構成は変化しません。
- lang=c99 を指定すると，C99 規格を含めたすべての関数が使用できますが，lang=c 指定時に比べて関数の数が多いため，ライブラリ生成に多くの時間が必要になる場合があります。

-simple_stdio

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-simple_stdio
```

- 省略時解釈
  なし

［詳細説明］

- 機能縮小版の入出力関数を生成します。
- 機能縮小版では，浮動小数点の変換（nofloat オプションでサポートされない機能と同じ），long long 型の変換，2 バイトコードの変換が含まれません。これらの機能を必要としないファイル入出力を行う場合，ROM サイズを削減することができます。
  対象関数 fprintf，fscanf，printf，scanf，sprintf，sscanf，vfprintf，vprintf，vsprintf
- 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・ジェネレート・オプション］タブの［オブジェクト］カテゴリの［機能縮小版入出力関数を生成する］

［備考］

- 対象関数で縮小された機能を使用した場合，本オプションを指定して作成したライブラリをリンクした時の動作は保証しません。
- 本機能は，C++ または EC++ コンパイル時は無効です。

-logo

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-logo
```

- 省略時解釈
  コピーライト表示が出力されます。

［詳細説明］

- コピーライト表示が出力されます。

-nologo

＜ライブラリジェネレータ・オプション / ライブラリオプション＞

［指定形式］

```
-nologo
```

- 省略時解釈
  コピーライト表示が出力されます。

［詳細説明］

- nologo オプション指定時は，コピーライトの表示の出力が抑止されます。
- 本オプションは，CubeSuite+ の以下のプロパティに相当します。
- ［ライブラリ・ジェネレート・オプション］タブの［その他］カテゴリの［コピーライトを出力する］

無効となるコンパイラオプション

ライブラリジェネレータでは、B.1.3.4 ライブラリジェネレータ・オプションの他に C/C++ コンパイラオプションを指定し、ライブラリをコンパイルするときに用いるオプションとして選択することができます。ただし、以下に示すオプションは無効となり、ライブラリのコンパイルでは選択されません。

表 B.18 無効オプション一覧

No.	無効とされるオプション	無効になる条件	無効になった場合にライブラリ構築時に選択されるオプション
1	lang	常に無効	なし
2	include	常に無効	なし
3	define	常に無効	なし
4	undefined	常に無効	なし
5	message nomessage	常に無効	nomessage
6	change_message	常に無効	なし
7	file_inline_path	常に無効	なし
8	comment	常に無効	なし
9	check	常に無効	なし
10	output	常に無効	output=obj
11	noline	常に無効	なし
12	debug nodebug	常に無効	nodebug
13	listfile nolistfile show	常に無効	nolistfile
14	file_inline	常に無効	なし
15	asmcmd	常に無効	なし
16	lnkcmd	常に無効	なし
17	asmopt	常に無効	なし
18	lnkopt	常に無効	なし
19	logo nologo	常に無効	nologo
20	euc sjis latin1 utf8	常に無効	なし
21	outcode	常に無効	なし
22	subcommand	常に無効	なし
23	alias	常に無効	alias=noansi
24	pic pid	lang=cpp または C++ ソースコンパイル時[注 1]	なし

No.	無効とされるオプション	無効になる条件	無効になった場合にライブラリ構築時に選択されるオプション
25	ip_optimize	常に無効	なし
26	merge_files	常に無効	なし
27	whole_program	常に無効	なし
28	big5 gb2312	常に無効注 2	なし
29	map nomap	常に無効	smap

注 1. 警告 W0511171 が表示されます。

注 2. エラー F0593305 になります (ライブラリ生成できません)。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
Rev.1.00	2014.05.20	―	初版発行

CubeSuite+ V2.02.00　ユーザーズマニュアル
RX ビルド編

発行年月日　2014.05.20　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 エレクトロニクス株式会社 〒100-0004 千代田区大手町2-6-2（日本ビル）

■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 2.0

CubeSuite+ V2.02.00

RENESAS

ルネサス エレクトロニクス株式会社

R20UT2998JJ0100