

お客様各位

---

## カタログ等資料中の旧社名の扱いについて

---

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願ひ申し上げます。

ルネサスエレクトロニクス ホームページ (<http://www.renesas.com>)

2010年4月1日  
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社 (<http://www.renesas.com>)

【問い合わせ先】 <http://japan.renesas.com/inquiry>

本ドキュメントに記載されているURLは、以下のとおり読み替えをお願いいたします。  
<http://www.necel.com/>  
<http://www2.renesas.com/>

開発環境トップページ <http://japan.renesas.com/tools>  
ダウンロードポータル [http://japan.renesas.com/tool\\_download](http://japan.renesas.com/tool_download)

技術問合せについては、以下のページをご覧ください。  
[http://japan.renesas.com/tech\\_inquiry](http://japan.renesas.com/tech_inquiry)

ツールユーザ登録については、以下のページをご覧ください。  
<http://japan.renesas.com/myrenesas>

## ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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。
5. 輸出に際しては、「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。本資料に記載されている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の目的で使用しないでください。また、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に使用することができません。
6. 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のですが、誤りが無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。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されてい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生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。  
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  
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、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）  
特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社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。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、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、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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、当社は、一切その責任を負いません。
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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社と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。

# ユーザズ・マニュアル

## CA850 Ver.3.20

Cコンパイラ・パッケージ

リンク・ディレクティブ編

---

対象デバイス  
V850シリーズ

資料番号 U18515JJ1V0UM00 (第1版)  
発行年月 May 2007 CP(K)

(メモ)

## 目次要約

第1章 概 説 ...	14
第2章 インストレーション ...	17
第3章 起動と終了 ...	18
第4章 生成方法 ...	19
第5章 ウィンドウ・レファレンス ...	23
第6章 メッセージ ...	71
付録A リンク・ディレクティブ ...	77
総合索引 ...	123

Windowsは米国Microsoft Corporationの米国およびその他の国における登録商標または商標です。

- 本資料に記載されている内容は2007年5月現在のもので、今後、予告なく変更することがあります。量産設計の際には最新の個別データ・シート等をご参照ください。
- 文書による当社の事前の承諾なしに本資料の転載複製を禁じます。当社は、本資料の誤りに関し、一切その責を負いません。
- 当社は、本資料に記載された当社製品の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、一切その責を負いません。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。
- 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。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関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、一切その責を負いません。
- 当社は、当社製品の品質、信頼性の向上に努めておりますが、当社製品の不具合が完全に発生しないことを保証するものではありません。当社製品の不具合により生じた生命、身体および財産に対する損害の危険を最小限度にするために、冗長設計、延焼対策設計、誤動作防止設計等安全設計を行ってください。
- 当社は、当社製品の品質水準を「標準水準」、「特別水準」およびお客様に品質保証プログラムを指定していただく「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。

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

特別水準：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、生命維持を目的として設計されていない医療機器

特定水準：航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器、生命維持のための装置またはシステム等

当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。意図されていない用途で当社製品の使用をお客様が希望する場合には、事前に当社販売窓口までお問い合わせください。

(注)

- (1) 本事項において使用されている「当社」とは、NECエレクトロニクス株式会社およびNEC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う。
- (2) 本事項において使用されている「当社製品」とは、(1)において定義された当社の開発、製造製品をいう。

# はじめに

**対象デバイス** V850シリーズCコンパイラ・パッケージは、NECエレクトロニクス製RISCマイクロプロセッサV850シリーズ用のオブジェクト・コードを生成するためのCコンパイラ・パッケージです。

**対象者** このマニュアルは、V850シリーズCコンパイラ・パッケージを使用して、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** このマニュアルでは、パッケージに含まれるリンカ (ld850) がサポートするメモリ配置情報を記述するリンク・ディレクティブ・ファイルをGUI操作するリンク・ディレクティブ・ジェネレータとリンク・ディレクティブ仕様について説明しています。

**構成** このマニュアルは、次の内容で構成されています。

- ・概説
- ・インストレーション
- ・起動と終了
- ・生成方法
- ・ウィンドウ・レファレンス
- ・メッセージ



**関連資料** このマニュアルを使用する場合は、次の資料もあわせてご覧ください。

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。

**開発ツールに関する資料 (ユーザーズ・マニュアル)**

資料名		資料番号	
		和文	英文
CA850 Ver.3.20 Cコンパイラ・パッケージ	操作編	U18512J	U18512E
	C言語編	U18513J	U18513E
	アセンブリ言語編	U18514J	U18514E
	リンク・ディレクティブ編	このマニュアル	U18515E
PM+ Ver.6.30 プロジェクト・マネージャ		U18416J	U18416E
ID850 Ver.3.00 統合デバッグ	操作編	U17358J	U17358E
ID850NW Ver.3.10 統合デバッグ	操作編	U17369J	U17369E
ID850QB Ver.3.20 統合デバッグ	操作編	U17964J	U17964E
SM+ システム・シミュレータ	操作編	U17246J	U17246E
	ユーザ・オープン・インタフェース編	U18212J	U18212E
SM850 Ver.2.50 システム・シミュレータ	操作編	U16218J	U16218E
SM850 Ver.2.00以上 システム・シミュレータ	外部部品ユーザ・オープン・インタフェース仕様編	U14873J	U14873E
RX850 Ver.3.20以上 リアルタイムOS	基礎編	U13430J	U13430E
	インストレーション編	U17419J	U17419E
	テクニカル編	U13431J	U13431E
	タスク・デバッグ編	U17420J	U17420E
RX850 Pro Ver.3.21 リアルタイムOS	基礎編	U18165J	U18165E
	内部構造編	U18164J	U18164E
	タスク・デバッグ編	U17422J	U17422E
RX850V4 Ver.4.22 リアルタイムOS	機能編	U16643J	U16643E
	内部構造編	U16644J	U16644E
	タスク・デバッグ編	U16811J	U16811E
RX-NET ネットワーク・ライブラリ (TCP/IP) Ver.1.30		U15083J	-
RX-NET ネットワーク・ライブラリ (PPP) Ver.1.30		U15303J	-
RX-NET ネットワーク・ライブラリ (DNS) Ver.1.30		U15304J	-
RX-NET ネットワーク・ライブラリ (DHCP) Ver.1.30		U15382J	-
RX-NET ネットワーク・ライブラリ (SMTP)		U15505J	-
RX-NET ネットワーク・ライブラリ (POP)		U15539J	-
RX-NET Ver.1.10 ネットワーク・ライブラリ (telnet)		U16085J	-
RX-NET Ver.1.00 ネットワーク・ライブラリ (FTP)		U15946J	-
RX-NET Ver.1.00 ネットワーク・ライブラリ (WebServer)		U16294J	-
AZ850 Ver.3.30 システム・パフォーマンス・アナライザ		U17423J	U17423E
AZ850V4 Ver.4.10 システム・パフォーマンス・アナライザ		U17093J	U17093E
TW850 Ver.2.00 性能解析チューニング・ツール		U17241J	U17241E

[メモ]

# 目次

第 1 章	概説	...	14
1.1	概要	...	14
1.2	システム構成	...	15
1.3	動作環境	...	16
第 2 章	インストレーション	...	17
2.1	LDG のインストール	...	17
2.2	フォルダ構成	...	17
2.3	LDG のアンインストール	...	17
第 3 章	起動と終了	...	18
3.1	起動方法	...	18
3.2	終了方法	...	18
第 4 章	生成方法	...	19
4.1	生成手順	...	19
4.2	開発環境の設定	...	20
4.2.1	リンク・ディレクティブ・ファイルの新規作成	...	20
4.2.2	既存のリンク・ディレクティブ・ファイルの編集	...	20
4.3	編集	...	21
4.3.1	メモリの追加	...	21
4.3.2	セクションの追加	...	21
4.3.3	オブジェクト・ファイルの追加	...	21
4.4	保存	...	22
4.4.1	リンク・ディレクティブ・ファイルの書式	...	22
第 5 章	ウインドウ・リファレンス	...	23
5.1	LDG のウインドウ / ダイアログの概要	...	23
5.2	各ウインドウ / ダイアログの説明	...	24
	メイン・ウインドウ	...	25
	[新しいリンク・ディレクティブ] ダイアログ	...	45
	[環境選択] ダイアログ	...	47
	[開く] ダイアログ	...	49
	[名前を付けて保存] ダイアログ	...	51
	[検索] ダイアログ	...	53

[オブジェクト・ファイルを選択]ダイアログ ...	55
[メモリを追加]ダイアログ ...	57
[セクションを追加]ダイアログ ...	60
[シンボルを追加]ダイアログ ...	64
[オプション]ダイアログ ...	67

## 第6章 メッセージ ... 71

6.1 表示形式 ...	71
6.2 エラー・メッセージ ...	72
6.3 警告メッセージ ...	72
6.4 質問メッセージ ...	75
6.5 情報メッセージ ...	76

## 付録A リンク・ディレクティブ ... 77

A.1 概要 ...	77
A.1.1 指定項目 ...	77
A.2 セクションとセグメント ...	79
A.2.1 セクション ...	79
A.2.2 セグメント ...	79
A.2.3 セグメントとセクションの関係 ...	81
A.2.4 セクションの種類 ...	81
A.2.5 セクションのタイプと属性 ...	86
A.3 シンボル ...	88
A.3.1 テキスト・ポインタ (tp) ...	88
A.3.2 グローバル・ポインタ (gp) ...	89
A.3.3 エlement・ポインタ (ep) ...	92
A.4 書式 ...	94
A.4.1 使用できる文字 ...	94
A.4.2 ファイル名 ...	95
A.4.3 セグメント・ディレクティブ ...	95
A.4.4 マッピング・ディレクティブ ...	101
A.4.5 シンボル・ディレクティブ ...	109
A.5 デフォルト ...	114
A.6 リンク・ディレクティブ・ファイルの記述例 ...	115

## 総合索引 ... 123

# 図の目次

図番号	タイトル, ページ
1-1	システム構成例 ... 15
2-1	フォルダ構成 ... 17
3-1	起動時のメイン・ウインドウ ... 18
4-1	リンク・ディレクティブ・ファイルの生成手順 ... 19
5-1	メイン・ウインドウ ... 25
5-2	メモリ・マッピング・ビュー・エリアの例 ... 26
5-3	メモリの表示内容例 ... 27
5-4	ミラー・イメージ領域表示内容例 (各メモリが非表示の場合) ... 28
5-5	セクションの表示内容例 ... 29
5-6	オブジェクト・ファイル表示内容例 ... 30
5-7	シンボル表示内容例 ... 31
5-8	プロパティ・ビュー・エリア (セクションを選択している場合) ... 37
5-9	メッセージ・ビュー・エリアの例 ... 40
5-10	[新しいリンク・ディレクティブ]ダイアログ ... 45
5-11	[環境選択]ダイアログ ... 47
5-12	[開く]ダイアログ ... 49
5-13	[名前を付けて保存]ダイアログ ... 51
5-14	[検索]ダイアログ ... 53
5-15	[オブジェクト・ファイルを選択]ダイアログ ... 55
5-16	[メモリを追加]ダイアログ ... 57
5-17	[セクションを追加]ダイアログ ... 60
5-18	[シンボルを追加]ダイアログ ... 64
5-19	[オプション]ダイアログ ([フォント]を選択している場合) ... 67
5-20	[オプション]ダイアログでの[フォント]の設定 ... 68
5-21	[オプション]ダイアログでの[色]の設定 ... 69
5-22	[オプション]ダイアログでの[全般]の設定 ... 69
6-1	メッセージ・ダイアログの例 ... 71
A-1	セグメント・ディレクティブとマッピング・ディレクティブ ... 77
A-2	シンボル・ディレクティブ ... 78
A-3	セグメントとセクションの関係 ... 81
A-4	CA850 による各セクションのメモリ配置イメージの例 (内蔵 ROM あり) ... 85
A-5	tp の設定例 ... 88
A-6	gp の設定例 (セグメントを指定する場合) ... 89
A-7	gp の設定例 (tp からのオフセット指定の場合) ... 90
A-8	グローバル・ポインタ値の決定規則 ... 91
A-9	ep の設定例 ... 92
A-10	エレメント・ポインタ値の決定規則 ... 93

# 表の目次

表番号	タイトル, ページ
5-1	LDG のウインドウ / ダイアログ一覧 ... 23
5-2	メモリの表示内容 ... 27
5-3	ミラー・イメージ領域の表示内容 ... 28
5-4	メモリの属性と背景色の関係 (デフォルト) ... 28
5-5	セクションの表示内容 ... 29
5-6	セクションの属性と背景色の関係 (デフォルト) ... 30
5-7	セクションの表示内容 ... 31
5-8	マッピング・ビュー・エリアで直接編集できる項目とその留意事項 ... 32
5-9	ドラッグ・アンド・ドロップ操作による編集 ... 36
5-10	プロパティ・ビュー・エリアの表示内容と編集の可否 (メモリ選択時) ... 37
5-11	プロパティ・ビュー・エリアの表示内容と編集の可否 (セクション選択時) ... 38
5-12	プロパティ・ビュー・エリアの表示内容と編集の可否 (グループ選択時) ... 38
5-13	プロパティ・ビュー・エリアの表示内容と編集の可否 (オブジェクト・ファイル選択時) ... 39
5-14	プロパティ・ビュー・エリアの表示内容と編集の可否 (シンボル選択時) ... 39
5-15	メイン・ウインドウのツールバー ... 44
5-16	[新しいリンク・ディレクティブ] ダイアログの機能ボタン ... 46
5-17	[環境選択] ダイアログの機能ボタン ... 48
5-18	[開く] ダイアログの機能ボタン ... 50
5-19	[名前を付けて保存ダイアログ] の機能ボタン ... 52
5-20	[検索] ダイアログの機能ボタン ... 54
5-21	[オブジェクト・ファイルを選択] ダイアログの機能ボタン ... 56
5-22	[メモリを追加] ダイアログの機能ボタン ... 59
5-23	[セクションを追加] ダイアログの機能ボタン ... 63
5-24	[シンボルを追加] ダイアログの機能ボタン ... 66
5-25	[オプション] ダイアログのカテゴリ ... 68
5-26	[オプション] ダイアログの機能ボタン ... 70
6-1	メッセージ種別一覧 ... 71
6-2	エラー・メッセージ一覧 ... 72
6-3	警告メッセージ一覧 ... 72
6-4	質問メッセージ一覧 ... 75
6-5	情報メッセージ一覧 ... 76
A-1	CA850 の配置セクション種別 ... 82
A-2	セクションのタイプ ... 86
A-3	セクションの属性 ... 86
A-4	セクションの分類 ... 87
A-5	セグメント・ディレクティブで指定できる項目 ... 95
A-6	省略可能項目のデフォルト値 (セグメント・ディレクティブ) ... 96
A-7	セグメント名が固定されている予約セクション名 ... 96
A-8	セグメント属性とその意味 ... 97
A-9	セグメント例 ... 99
A-10	マッピング・ディレクティブで指定できる項目 ... 101
A-11	マッピング・ディレクティブの指定項目で省略可能な値のデフォルト値 / 規則 ... 102

A-12	出力セクション名が固定されている入力セクション	...	102
A-13	セクション・タイプとその意味	...	103
A-14	セクション属性とその意味	...	103
A-15	セクションの種類とその整列条件のデフォルト値	...	104
A-16	入力セクションとオブジェクト・ファイルの組み合わせ別の出力	...	105
A-17	入力セクションとオブジェクト・ファイルの組み合わせの具体例	...	105
A-18	マッピング・ディレクティブの指定例	...	108
A-19	tp シンボル作成で指定できる項目	...	109
A-20	tp シンボルのデフォルト値	...	109
A-21	gp シンボル作成で指定できる項目	...	110
A-22	gp シンボルのデフォルト値	...	110
A-23	ep シンボル作成で指定できる項目	...	111
A-24	ep シンボルのデフォルト値	...	111
A-25	tp シンボル, gp シンボルのアドレス指定	...	112
A-26	tp シンボル, gp シンボルの参照対象セグメント名	...	113
A-27	シンボル・ディレクティブの指定例	...	113

# 第1章 概説

## 1.1 概要

組み込み系のアプリケーション・システムでは、プログラム・コードやデータを特定のアドレスから配置したり分割して配置するなど、ターゲット・デバイスのアドレス空間に応じたメモリ配置に煩雑な気配りが必要となります。

リンク・ディレクティブ・ジェネレータ (Link Directive Generator : 以降 LDG と略します) は、これらのメモリ配置情報 (セクション情報など) を記述するリンク・ディレクティブ・ファイルを、GUI 操作により自動生成 / 編集するツールです。

ターゲット・デバイスのアドレス空間と、リンカが出力した実行可能オブジェクト・ファイルのアドレス配置イメージを視覚的に表示する LDG の機能を利用することにより、円滑かつ効率的にリンク・ディレクティブ・ファイルを生成 / 編集することができます。

LDG の主な機能と特徴は次のとおりです。

### **グラフィカルなマッピング表示**

使用するターゲット・デバイスの物理メモリ・マッピング、セクション・マッピング、および各セクションに含まれるオブジェクト・ファイル名や関連付けたシンボル名などを、1つのビューとしてグラフィカルに表示します。

### **GUI 操作によるリンク・ディレクティブ・ファイルの自動生成 / 編集**

セクション名、オブジェクト・ファイル名、またはシンボル名などをマウスによりドラッグすることにより、それらの配置を視覚的に編集することができます。

### **既存のリンク・ディレクティブ・ファイルの再編集**

LDG 以外で作成された既存のリンク・ディレクティブ・ファイルも読み込み可能です。

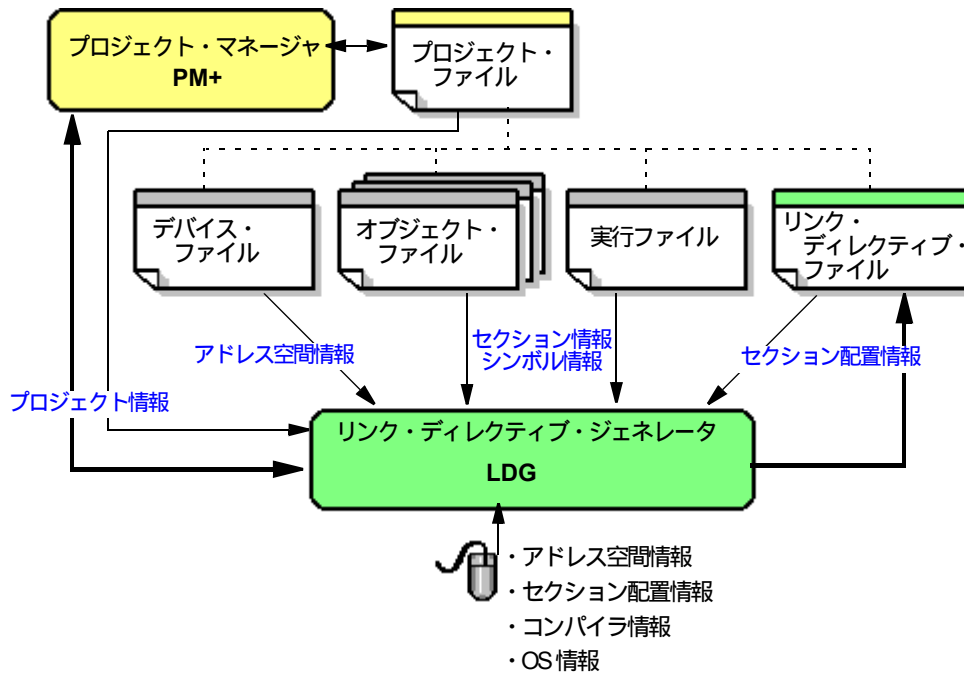


## 1.2 システム構成

LDG は、デバイス・ファイルからアドレス空間情報を、オブジェクト・ファイル/実行ファイル (ELF フォーマット) からセクション情報/シンボル情報を取得します。

LDG のシステム構成を次に示します。

図 1-1 システム構成例



【備考】PM+ と連携せず、LDG を単独で使用することも可能です。

## 1.3 動作環境

LDG を使用するには、次の環境が必要となります。

### (1) ホスト・マシン

次の OS が動作するもの

CPU : Pentium™ II 400MHz 以上

メモリ : 128M バイト以上

OS : Windows® 2000 , Windows XP Professional , Windows XP Home Edition

**【注意】** いずれの OS の場合も、最新の Service Pack がインストールされている必要があります。

### (2) ソフトウェア

- コンパイラ

CA850 ( Ver.3.00 以降 )

- デバイス・ファイル

使用するターゲット・デバイスのデバイス・ファイル

- 開発ツール ( 必要な場合のみ )

PM+ ( Ver.6.00 以降 )

## 第2章 インストール

### 2.1 LDG のインストール

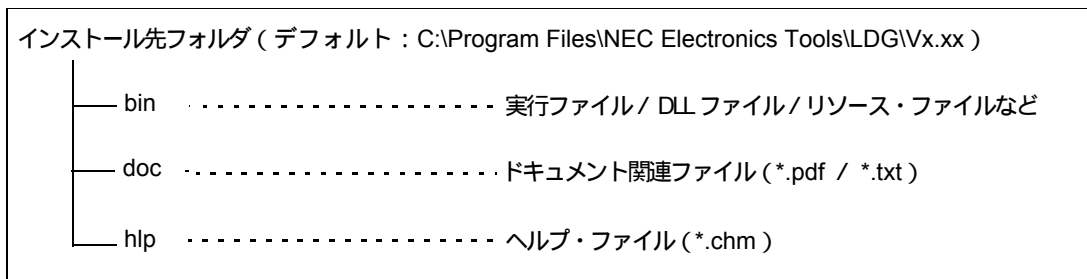
LDG は、コンパイラ・パッケージ (CA850) に含まれています。そのため、CA850 をインストールする際に、LDG をインストールすることになります。

CA850 のインストール方法については、「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル 操作編」を参照してください。

### 2.2 フォルダ構成

LDG のインストールによって構築されるフォルダ構成は、次のとおりです。

図 2-1 フォルダ構成



なお、Windows スタート・メニュー内には、LDG のショート・カット (デフォルト : [プログラム] [NEC Electronics Tools] [LDG] [Vx.xx] [LDG Vx.xx]) が自動的に追加されます。

### 2.3 LDG のアンインストール

LDG のアンインストールは、Windows のコントロール・パネルの “プログラムの追加と削除” (WindowsXP 以外の場合は、“プログラムの追加と削除”) を起動し、次の項目を選択することにより行います。

- NEC EL LDG Vx.xx
- NEC EL LDG Vx.xx ドキュメント一式

## 第3章 起動と終了

### 3.1 起動方法

LDG の起動は、次のいずれかの方法で行います。

#### (1) Windows [スタート]メニュー内ショート・カットからの起動

Windows の [スタート]メニュー [プログラム] [NEC Electronics Tools] [LDG] [Vx.xx] [LDG Vx.xx] (デフォルト) を選択します。

#### (2) PM+ からの起動

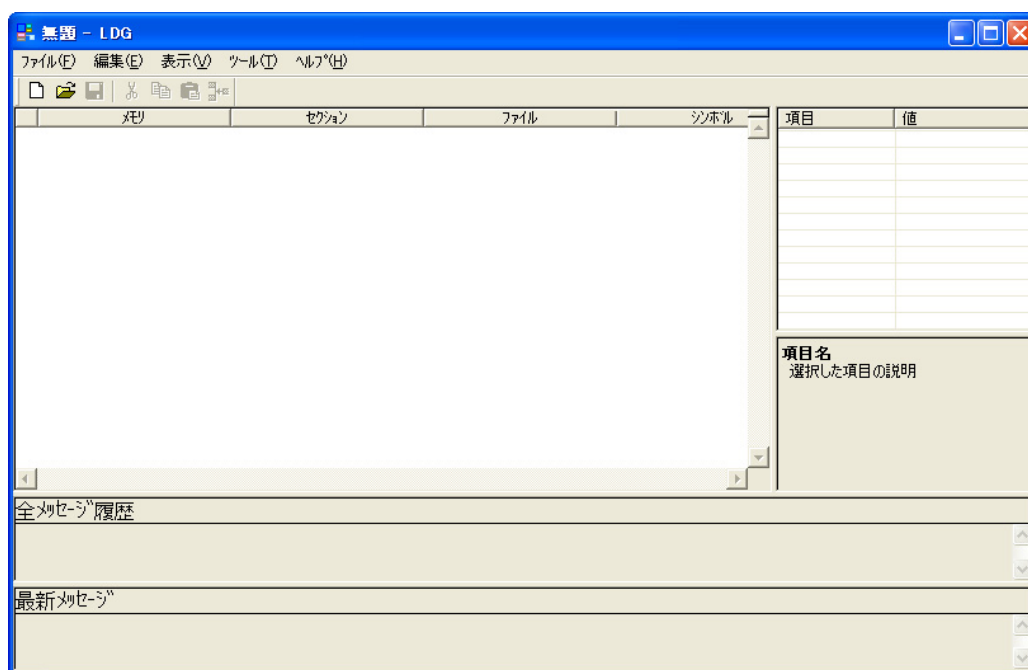
PM+ のメイン・ウインドウにおいて、[ツール]メニュー [LDG の起動] を選択します。

#### (3) LDG の実行ファイルのダブル・クリック

LDG の実行ファイル (デフォルト : C:\Program Files\NEC Electronics Tools\LDG\Vx.xx\bin\LDG.exe) を直接ダブル・クリックします。

LDG を起動すると、次のメイン・ウインドウがオープンします。

図 3-1 起動時のメイン・ウインドウ



**【注意】** LDG を PM+ から起動しない場合、PM+ 上で設定した “プロジェクト情報” (プロジェクトを構成するオブジェクト・ファイルの情報など) を LDG で利用することはできません。

### 3.2 終了方法

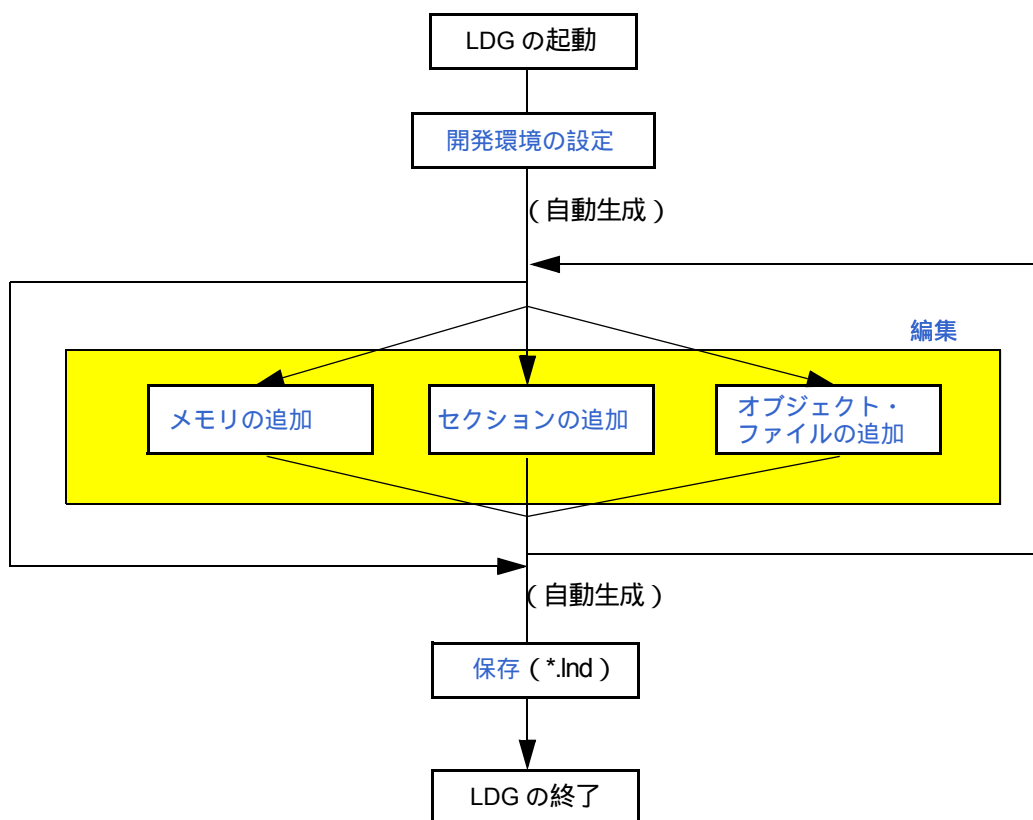
LDG の終了は、メイン・ウインドウ上の [ファイル]メニュー [終了] を選択することにより行います。

## 第4章 生成方法

### 4.1 生成手順

LDG を利用して、リンク・ディレクティブ・ファイルを生成する手順は次のとおりです。

図 4-1 リンク・ディレクティブ・ファイルの生成手順



## 4.2 開発環境の設定

作成 / 編集するリンク・ディレクティブ・ファイルの開発環境の設定を行います。

### 4.2.1 リンク・ディレクティブ・ファイルの新規作成

新規にリンク・ディレクティブ・ファイルを作成する場合、次の手順により行います。

- (1) [ファイル]メニュー [新規作成 ...] の選択

メイン・ウィンドウ上の [ファイル]メニュー [新規作成 ...] の選択により、[\[新しいリンク・ディレクティブ\]ダイアログ](#)をオープンします。

- (2) 開発環境の詳細設定

使用するデバイス名とコンパイラ名、および必要に応じてリアルタイム OS 名を指定します。

- (3) [OK] ボタンのクリック

ダイアログ内の設定が終了したのち、[OK] ボタンをクリックします。

指定したデバイスに対応した内蔵 ROM / RAM 領域と、最低限必要なセクションやリアルタイム OS に必要なセクションを割り付けた状態が[メイン・ウィンドウ](#)に反映されます。

**【備考】** PM+ から LDG を起動した場合は、PM+ の開発環境（プロジェクト情報）を反映した状態で、[\[新しいリンク・ディレクティブ\]ダイアログ](#)をオープンします。

### 4.2.2 既存のリンク・ディレクティブ・ファイルの編集

LDG 以外で作成された既存のリンク・ディレクティブ・ファイルを編集する場合、次の手順により行います。

- (1) [ファイル]メニュー [開く ...] の選択

メイン・ウィンドウ上の [ファイル]メニュー [開く ...] の選択により、[\[開く\]ダイアログ](#)をオープンし、編集を行いたいリンク・ディレクティブ・ファイルを選択します。

- (2) 開発環境の詳細設定

次に自動的にオープンする [\[環境選択\]ダイアログ](#)上で、使用するデバイス名とコンパイラ名を指定します。

- (3) [OK] ボタンのクリック

ダイアログ内の設定が終了したのち、[OK] ボタンをクリックします。

指定したデバイスに対応した内蔵 ROM / RAM 領域と、最低限必要なセクションやリアルタイム OS に必要なセクションを割り付けた状態が[メイン・ウィンドウ](#)に反映されます。

**【備考】** PM+ から LDG を起動した場合、PM+ の開発環境（プロジェクト情報）を自動的に反映して、リンク・ディレクティブ・ファイルを開きます（[\[環境選択\]ダイアログ](#)はオープンしません）。

## 4.3 編集

### 4.3.1 メモリの追加

デバイスの内蔵メモリ以外のメモリがある場合は、次の手順により新規にメモリを追加します。

- (1) [編集]メニュー [追加] [メモリ]の選択

メイン・ウィンドウ上の[編集]メニュー [追加] [メモリ]の選択により、[メモリを追加]ダイアログをオープンします。

- (2) メモリの詳細情報の設定

新規に追加するメモリの種類 (ROM / RAM) / 開始番地 / サイズ / アラインなどを設定したのち、[OK] ボタンをクリックします。

### 4.3.2 セクションの追加

新たにセクションを割り当てる必要がある場合は、次の手順により行います。

- (1) [編集]メニュー [追加] [セクション]の選択

メイン・ウィンドウ上のメモリ・マッピング・ビュー・エリアにおいて、新たにセクションを割り当てたいメモリを選択したのち、[編集]メニュー [追加] [セクション]の選択により、[セクションを追加]ダイアログをオープンします。

- (2) セクションの詳細情報の設定

新規に追加するアクセス属性 (リードオンリー / リードライト可 / 命令コード) / セクション名 / 配置方法 (アドレス指定 / 前セクションに続けて配置)などを設定したのち、[OK] ボタンをクリックします。

**【備考】**セクション同士を重ねた領域に配置する場合、追加するセクションをミラー領域に配置してください。ミラー領域の表示は、[表示]メニュー [ミラー・イメージを表示]をチェックすることにより行われず (デフォルトではチェックされています)。

### 4.3.3 オブジェクト・ファイルの追加

リンクするオブジェクト・ファイル (\*.o) / 実行ファイル (\*.out) を追加する場合は、次の手順により行います。

なお、LDG では、指定したオブジェクト・ファイルからセクション情報を読み出しメモリに割り当てます。また、セクションのサイズのチェックも行います。

- (1) [ファイル]メニュー [オブジェクト・ファイルを選択...]の選択

メイン・ウィンドウ上の[ファイル]メニュー [オブジェクト・ファイルを選択...]の選択により、[開く]ダイアログをオープンし、リンクしたいオブジェクト・ファイル (\*.o) / 実行ファイル (\*.out) を選択したのち [OK] ボタンをクリックします。

なお、この際、オブジェクト・ファイルは複数選択することができます。

## 4.4 保存

編集作業がすべて終わったら、リンク・ディレクティブ・ファイルを次の手順で保存します。

なお、LDG では、追加したメモリやセクションの情報を付加してリンク・ディレクティブ・ファイルを保存します。

(1) [ファイル]メニュー [名前を付けて保存 ...]の選択

メイン・ウインドウ上の[ファイル]メニュー [名前を付けて保存 ...]の選択により、[名前を付けて保存]ダイアログをオープンし、保存するファイル名(\*.lnd)を指定したのち[OK]ボタンをクリックします。

### 4.4.1 リンク・ディレクティブ・ファイルの書式

LDG が生成するリンク・ディレクティブ・ファイルには、デバイス情報/メモリ情報/コメントがコメント形式で追加されます。

なお、文字コードは“シフト JIS コード”，改行コードは“CR+LF”で保存されます。

**【注意】** LDG がリンク・ディレクティブ・ファイルに出力したデバイス情報/メモリ情報を編集した場合、正しく読み込める保証はありません。

**【備考】** リンク・ディレクティブ・ファイルを生成する際に、[名前を付けて保存]ダイアログ上の[LDG 情報を出力する]チェック・ボックスのチェックを外すことにより、LDG 固有の情報をファイルに追加しません。



## 第5章 ウィンドウ・リファレンス

### 5.1 LDG のウィンドウ / ダイアログの概要

LDG には、次のウィンドウ / ダイアログが用意されています。

表 5-1 LDG のウィンドウ / ダイアログ一覧

ウィンドウ / ダイアログ名	機能概要
メイン・ウィンドウ	LDG の基本動作を行うウィンドウです。 このウィンドウ上で、ターゲット・デバイスの物理メモリ・マッピング、セクション・マッピング、および各セクションに含まれるオブジェクト・ファイル名や関連付けているシンボル名を表示 / 編集します。
[新しいリンク・ディレクティブ] ダイアログ	新規にリンク・ディレクティブ・ファイルを生成します。
[環境選択] ダイアログ	LDG 以外で作成された既存のリンク・ディレクティブ・ファイルを開く場合、あらたに環境の設定を行います。
[開く] ダイアログ	LDG が読み込むファイルを指定します。
[名前を付けて保存] ダイアログ	編集後のファイルを名前を付けて保存します。
[検索] ダイアログ	メモリ名 / セクション名 / オブジェクト・ファイル名の検索、または LDG が出力したメッセージ内の文字列の検索を行います。
[オブジェクト・ファイルを選択] ダイアログ	新規にオブジェクト・ファイルを追加します。
[メモリを追加] ダイアログ	新規にメモリを追加します。
[セクションを追加] ダイアログ	新規にセクションを追加します。
[シンボルを追加] ダイアログ	新規にシンボルを追加します。
[オプション] ダイアログ	LDG の動作や表示に関する基本設定を行います。

## 5.2 各ウィンドウ/ダイアログの説明

LDGのウィンドウ/ダイアログについて、次の形式で説明します。

### ウィンドウ/ダイアログ名

枠内にウィンドウ/ダイアログ名を示します。

ここでは、ウィンドウ/ダイアログの表示イメージ、機能概要、およびオープン方法を示します。

#### 各エリアの説明

ウィンドウ/ダイアログの機能を各エリアごとに説明します。

#### メニューバー

メニューバーの対象となる項目からプルダウンされるメニュー項目を列挙し、各機能を説明します。

#### ツールバー

ツールバー上のボタン群の各機能を説明します。

#### 機能ボタン

ダイアログ内のボタンによる動作の説明をします。

#### その他

ウィンドウ/ダイアログが持つ特別な機能として、操作方法など特記すべき内容を記述します。

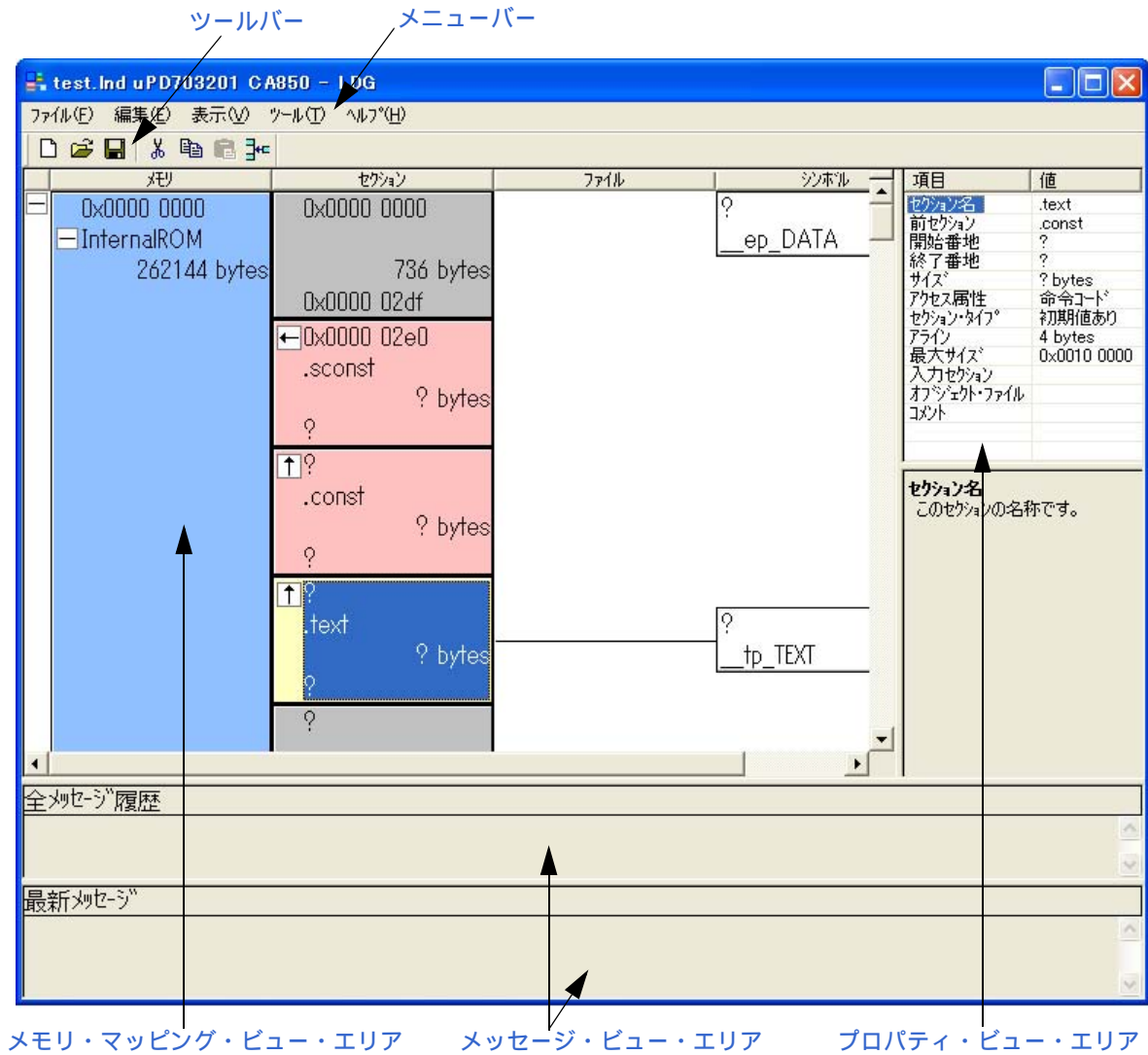
#### 操作上の注意事項

ウィンドウ/ダイアログを操作する際の注意事項を列挙します。

# メイン・ウィンドウ

LDG 起動後、自動的にオープンするウィンドウです。LDG を使用するためには、まずこのウィンドウより操作を開始します。

図 5-1 メイン・ウィンドウ



ここでは、次の項目について説明します。

メモリ・マッピング・ビュー・エリア

- (1) 各項目の表示内容について
- (2) メモリ・マッピング・ビュー・エリアでの編集方法

プロパティ・ビュー・エリア

メッセージ・ビュー・エリア

メニューバー

- (1) [ファイル(F)] メニュー
- (2) [編集(E)] メニュー
- (3) [表示(V)] メニュー
- (4) [ツール(T)] メニュー
- (5) [ヘルプ(H)] メニュー

ツールバー

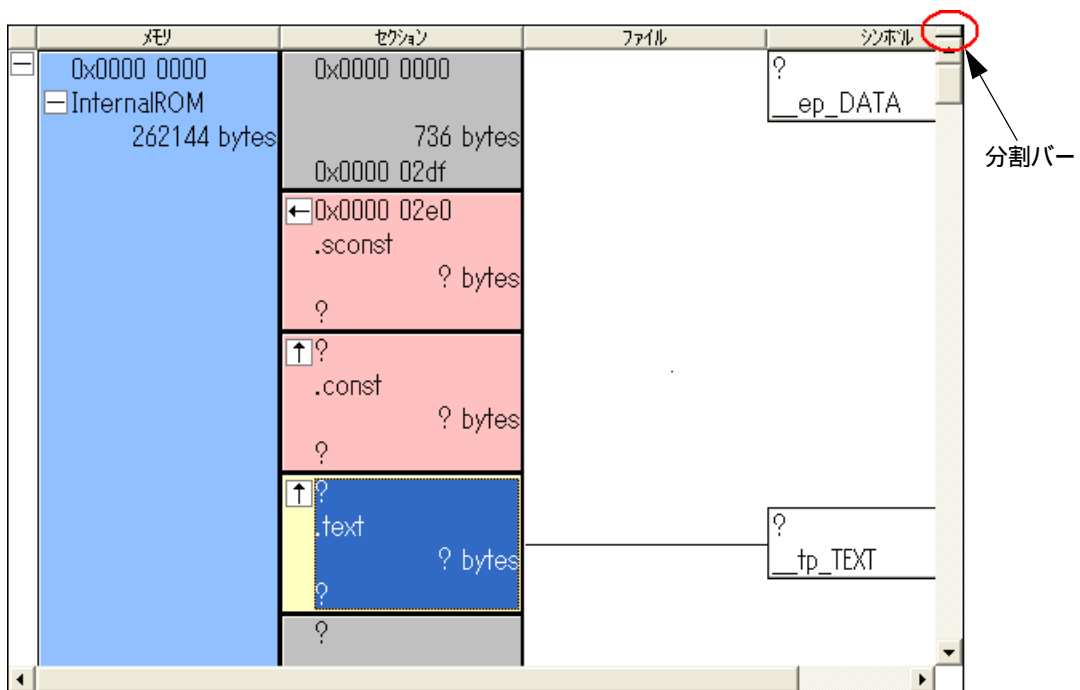
## メモリ・マッピング・ビュー・エリア

ターゲット・デバイスの物理メモリ・マッピング、セクション・マッピング、および各セクションに含まれるオブジェクト・ファイル名や関連付けているシンボル名を表示 / 編集するエリアです。

垂直スクロール・バーの上部にある分割バーをドラッグすると、表示領域を上下に2分割することができます。

なお、このエリアにおけるアドレス表示は、すべて16進数のアドレス桁数分で行います。足りない桁は0でパディングし、4桁ごとに1文字分の空白を挿入し表示します。

図 5-2 メモリ・マッピング・ビュー・エリアの例



(1) 各項目の表示内容について

(a) メモリ表示

各メモリを属性ごとにボックス・イメージで表示し、それぞれのボックス内には、メモリ名 / サイズ / 開始番地 / 終了番地を表示します。

ボックス内をクリックするとそのメモリが選択状態となり、選択しているメモリの詳細情報をプロパティ・ビュー・エリアに表示します。

また、マウス・カーソルを各メモリ上に移動させると、そのメモリの詳細情報をポップ・アップ表示します (表示内容は、プロパティ・ビュー・エリアに表示する内容と同じです)。ポップ・アップ表示時間は、[オプション]ダイアログ上で変更することができます。

図 5-3 メモリの表示内容例

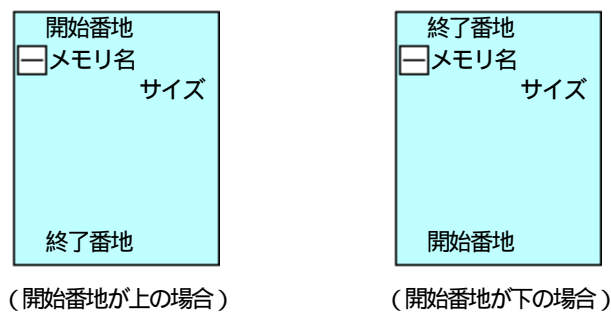


表 5-2 メモリの表示内容

項目	内容
メモリ名	メモリを識別するための名称を表示します。
	メモリにセクションが割り当てられている場合、メモリ名の左横に表示します。 “+” マークは、割り当てられているセクションが非表示である状態を示し、“-” マークは、割り当てられているセクションが表示されている状態を示します。
	このマークをクリックすることで、割り当てられているセクション表示の有無を切り替えることができます。
サイズ	メモリのサイズ (byte) を表示します。 サイズは、デフォルトでは 10 進数で表示しますが、次のいずれかの方法により、“0x” を接頭辞とした 16 進数で表示することができます。 ・ボックス内を右クリックすることで表示されるコンテキスト・メニューにおいて、[16 進数表示] を選択 ・ボックスを選択状態にしたのち、[表示]メニュー [16 進数表示] を選択
開始番地	メモリ領域の開始番地を “0x” を接頭辞とした 16 進数で表示します。
終了番地	メモリ領域の終了番地を “0x” を接頭辞とした 16 進数で表示します。

また、ターゲット・デバイスのアドレス空間がミラー・イメージを持っている場合は、各ミラー・イメージ領域を次のようにまとめて表示します。

図 5-4 ミラー・イメージ領域表示内容例（各メモリが非表示の場合）

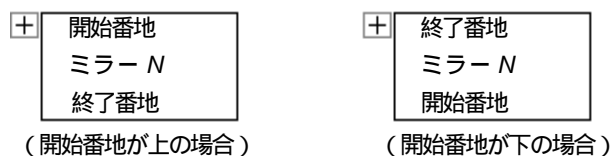




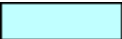





表 5-3 ミラー・イメージ領域の表示内容

項目	内容
ミラー <i>N</i>	“ <i>N</i> ” は、ミラー・イメージの番号（0 ~ ）を示します。
	ミラー・イメージ内の各メモリの表示の有無を示します。 “+” マークは、ミラー・イメージ内のメモリが非表示である状態を示し、“-” マークは、ミラー・イメージ内のメモリが表示されている状態を示します。
	このマークをクリックすることで、ミラー・イメージ内の各メモリの表示の有無を切り替えることができます。
開始番地	ミラー・イメージ領域の開始番地を“0x”を接頭辞とした 16 進数で表示します。
終了番地	ミラー・イメージ領域の終了番地を“0x”を接頭辞とした 16 進数で表示します。

【備考】 [表示] メニュー [ミラー・イメージを表示] の選択、または [オプション] ダイアログの [全般] により、ミラー・イメージ領域の表示の有無を指定することができます。

なお、各ボックス内の背景色はメモリの属性を示します。  
メモリの属性と背景色の関係は次のとおりです（デフォルト）。

表 5-4 メモリの属性と背景色の関係（デフォルト）

背景色	属性
	外部 ROM を示します。
	外部 RAM を示します。
	空き領域を示します。
	メモリの配置が不可能な領域を示します。
	内蔵 ROM を示します。
	内蔵 RAM を示します。

【備考】 [オプション] ダイアログの [色] により、ボックス内の背景色、および文字色を任意に指定することができます。

(b) セクション表示

セクションを属性ごとにボックス・イメージで表示し、それぞれのボックス内には、セクション名/サイズ/開始番地/終了番地を表示します。

ボックス内をクリックするとそのセクションが選択状態となり、選択しているセクションの詳細情報をプロパティ・ビュー・エリアに表示します。

また、マウス・カーソルを各セクション上に移動させると、そのセクションの詳細情報をポップ・アップ表示します（表示内容は、プロパティ・ビュー・エリアに表示する内容と同じです）。ポップ・アップ表示時間は、[オプション]ダイアログ上で変更することができます。

図 5-5 セクションの表示内容例

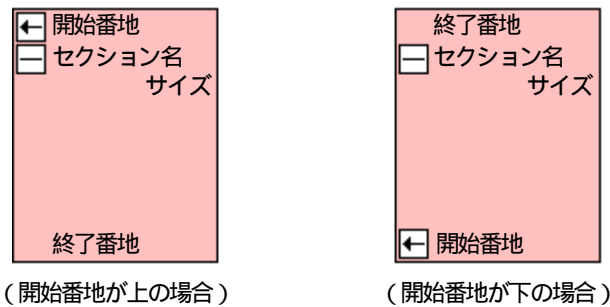


表 5-5 セクションの表示内容

項目	内容
セクション名	セクション名を表示します。
☐	オブジェクト・ファイルの読み込みを行っている場合、セクション名の左横に表示します。“+”マークは、対象オブジェクト・ファイルが非表示である状態を示し、“-”マークは、対象オブジェクト・ファイルが表示されている状態を示します。
⊕	このマークをクリックすることで、対象オブジェクト・ファイルの表示の有無を切り替えることができます。
サイズ	セクションのサイズ (bytes) を表示します。 サイズは、デフォルトでは 10 進数で表示しますが、次のいずれかの方法により、“0x” を接頭辞とした 16 進数で表示することができます。 ・ボックス内を右クリックすることで表示されるコンテキスト・メニューにおいて、[16 進数表示] を選択 ・ボックスを選択状態にしたのち、[表示]メニュー [16 進数表示] を選択
開始番地	セクションの開始番地を “0x” を接頭辞とした 16 進数で表示します。 前セクションに続いて配置している場合は、() で囲んで表示します。
←	セクションが開始番地から始まる場合、開始番地の左横に表示します。
↑	セクションが前セクションに続いて配置している場合、開始番地の左横に表示します。
終了番地	セクションの終了番地を “0x” を接頭辞とした 16 進数で表示します。 前セクションに続いて配置している場合は、() で囲んで表示します。

【注意】 次の状態の場合、該当する項目には “?” が表示されます。

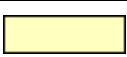
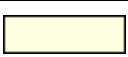




- ・ セクションのサイズが取得できない場合の [サイズ]

- セクションのサイズが取得できない場合で、他のセクションに続けて配置している場合の [ 開始番地 ] と [ 終了番地 ]
- 開始番地方向に隣接している連続したセクションのサイズ、または開始番地が確定していない場合の 空き領域の [ サイズ ] と [ 開始番地 ]

なお、各ボックス内の背景色はセクションの属性を示します。

セクションの属性と背景色の関係は次のとおりです ( デフォルト )。

表 5-6 セクションの属性と背景色の関係 ( デフォルト )

背景色		属 性
	ミラー・イメージ内	
		命令コードであることを示します。
		空き領域を示します。
		上記以外のセクションを示します。

【備考】 [ オプション ] ダイアログの [ 色 ] により、セクション・タイプ ( 初期値あり / 初期値なし )、およびアクセス属性に依存してボックス内の背景色 / 文字色を指定することができます。

### (c) オブジェクト・ファイル表示

各セクションに含まれるオブジェクト・ファイル名をボックス内に表示します。

各オブジェクト・ファイル名をクリックするとそのオブジェクト・ファイルが選択状態となり、選択しているオブジェクト・ファイルの詳細情報を **プロパティ・ビュー・エリア** に表示します。

また、マウス・カーソルを各オブジェクト・ファイル名上に移動させると、そのオブジェクト・ファイルの詳細情報をポップ・アップ表示します ( 表示内容は、プロパティ・ビュー・エリアに表示する内容と同じです )。ポップ・アップ表示時間は、 [ オプション ] ダイアログ上で変更することができます。

図 5-6 オブジェクト・ファイル表示内容例

オブジェクト・ファイル名1 オブジェクト・ファイル名2 オブジェクト・ファイル名3 オブジェクト・ファイル名4
--

【注意】ここでのオブジェクト・ファイルの表示順序は、リンクの解決順に影響します。



(d) シンボル表示

セクションに関連付けたシンボル名をボックス内に表示します。

シンボル表示は、リンクがセクションに関連付けたシンボルを生成できる場合にのみ行います。

各シンボルは、関連のあるメモリやセクションと線で繋いで表示されます。複数のメモリやセクションに関連付けたシンボルの場合は、それらを複数の線で繋いで表示します。

また、メモリやセクションの切れ目を基点とするシンボルの場合は、該当するメモリやセクションを示すボックスの上辺/下辺と線で繋いで表示します。

図 5-7 シンボル表示内容例

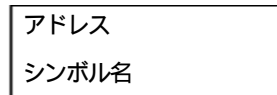


表 5-7 セクションの表示内容

項目	内容
アドレス	シンボルのアドレスを表示します。ただし、取得できない場合は、“?” を表示します。 なお、シンボルのアドレスが取得できる場合は、そのアドレスに即した位置にこのボックスを表示し、アドレスが取得できない場合は、関連付けたメモリ/セクションのボックスの中央右に表示します。
シンボル名	シンボルの名前を表示します。

## (2) メモリ・マッピング・ビュー・エリアでの編集方法

## (a) キーボード入力操作による編集

次の操作を行うことにより、メモリとセクション内の各項目の内容をキーボード入力による直接編集することができます。

## 【操作方法】

メモリ、またはセクションを示すボックスをマウスによりクリックすることで選択状態となります。この状態で、ボックス内の項目をもう一度クリックすると直接編集が可能となります。

## 【変更値の決定】

[Enter] キーの押下、または選択の解除により行われます。

ただし、不正な値を指定した場合は、選択を解除することはできません。この場合、メッセージ・ビュー・エリアにメッセージを表示します。

次に、上記方法により変更できる項目と、その際の留意事項を示します。

表 5-8 マッピング・ビュー・エリアで直接編集できる項目とその留意事項

項目	留意事項
<b>メモリ</b>	
メモリ名	1文字目は英字，2文字目以降は英数字を使用します。
サイズ	<ul style="list-style-type: none"> <li>- 変更に応じて [ 終了番地 ] を変更します。</li> <li>- 小さくした場合，隣接していたメモリとの間に空き領域を表示します。</li> <li>- 変更後，アドレスの大きい方に領域が重なるメモリが存在しない場合，空き領域の [ 開始番地 ] を変更します。</li> <li>- 変更後，アドレスの大きい方に領域が重なるメモリが存在する場合，変更は適用せず，メッセージ・エリアにメッセージを表示します。</li> </ul>
開始番地	- 変更後，別のメモリと領域が重なる場合，変更は適用せず，メッセージ・エリアにメッセージを表示します。
終了番地	<ul style="list-style-type: none"> <li>- 変更に応じて [ サイズ ] を変更します。</li> <li>- 小さくした場合，隣接していたメモリとの間に空き領域を表示します。</li> <li>- 変更後，アドレスの大きい方に領域が重なるメモリが存在しない場合，空き領域の [ 開始番地 ] を変更します。</li> <li>- 変更後，アドレスの大きい方に領域が重なるメモリが存在する場合，変更は適用せず，メッセージ・エリアにメッセージを表示します。</li> </ul>
<b>セクション</b>	
セクション名	特になし。
開始番地	- 変更後，別の [ セクション ] と領域が重なる場合，変更は適用せず，メッセージ・エリアにメッセージを表示します。

【注意】内蔵メモリ，空き領域，およびメモリ配置不能領域（SFR 領域など）は，編集操作によって値を変更することはできません（内蔵メモリの情報はデバイス・ファイルから取得します）。

**(b) マウス操作による編集**

マウス操作による次の機能を利用することより、メモリ、セクション、オブジェクト・ファイル、およびシンボルの配置や割付を視覚的に編集することができます。

- [コンテキスト・メニューの利用](#)
- [ドラッグ・アンド・ドロップ機能の利用](#)

**コンテキスト・メニューの利用**

マウスにより、メモリ/セクションの各ボックス、またはオブジェクト・ファイル名を右クリックすると、クリックした箇所のメモリ/セクション/オブジェクト・ファイルが選択状態となり、次のコンテキスト・メニューを表示します。

**【メモリ】**

[16進数表示 (H)]	チェックあり:[サイズ]を16進数で表示します。 チェックなし:[サイズ]を10進数で表示します(デフォルト)
[切り取り (I)]	選択状態のメモリを切り取ります。 この際、切り取ったメモリの[開始番地]/[終了番地]の情報はなくなります。 ただし、空きメモリ領域を選択している場合は、この項目は選択できません。
[コピー (C)]	選択状態のメモリをコピー・バッファにコピーします。 この際、コピーしたメモリの[開始番地]/[終了番地]の情報はなくなります。 ただし、空きメモリ領域を選択している場合は、この項目は選択できません。
[貼り付け (P)]	メモリを貼り付ける場合、選択状態のメモリのアドレスの小さい方に隣接させてメモリを貼り付けます(空きメモリ領域を選択している場合、その領域の[開始番地]にメモリを貼り付けます)。 セクションを貼り付ける場合、選択状態のメモリにセクションを配置します。 ただし、コピー・バッファにメモリ/セクションがコピーされていない場合は、この項目は選択できません。
[新規メモリ追加 (M)]	<a href="#">[メモリを追加]ダイアログ</a> をオープンします。
[新規セクション追加 (S)]	<a href="#">[セクションを追加]ダイアログ</a> をオープンします。
[新規シンボル追加 (B)]	<a href="#">[シンボルを追加]ダイアログ</a> をオープンします。
[削除 (D)]	選択状態のメモリを削除します。 ただし、空きメモリ領域を選択している場合は、この項目は選択できません。

## 【セクション】

[16進数表示 (H)]	<p>チェックあり:[サイズ]を16進数で表示します。</p> <p>チェックなし:[サイズ]を10進数で表示します(デフォルト)。</p>
[切り取り (I)]	<p>選択状態のセクションを切り取ります。</p> <p>この際、切り取ったセクションの[開始番地]/[終了番地]の情報はなくなります。</p> <p>ただし、空き領域を選択している場合は、この項目は選択できません。</p>
[コピー (Q)]	<p>選択状態のセクションをコピー・バッファにコピーします。</p> <p>この際、コピーしたセクションの[セクション名]/[開始番地]/[終了番地]の情報はなくなります。</p> <p>ただし、空き領域を選択している場合は、この項目は選択できません。</p>
[貼り付け (P)]	<p>セクションを貼り付ける場合、選択状態のセクションに続いて配置するように貼り付けます(空き領域を選択している場合、その領域の[開始番地]にセクションを貼り付けます)。</p> <p>オブジェクト・ファイルを貼り付ける場合、選択状態のセクションにオブジェクト・ファイルを貼り付けます。</p> <p>ただし、コピー・バッファにセクション/オブジェクト・ファイルがコピーされていない場合は、この項目は選択できません。</p>
[新規セクション追加 (S)]	[セクションを追加]ダイアログをオープンします。
[新規オブジェクトファイル選択 (E)]	[オブジェクト・ファイルを選択]ダイアログをオープンします。
[新規シンボル追加 (B)]	<p>[シンボルを追加]ダイアログをオープンします。</p> <p>gp / tp シンボルの生成が可能です( ep シンボルについては、1つしか生成できないため、新規に作成することはできません)。</p>
[削除 (D)]	<p>選択状態のセクションをメモリから削除します。</p> <p>ただし、空きメモリ領域を選択している場合は、この項目は選択できません。</p>
[グループ化 (G)]	<p>選択した複数のセクションをグループ化します(セグメントとして扱います)。</p> <p>ただし、複数のセクションを選択していない場合、この項目は選択できません。</p>
[グループ化解除 (U)]	<p>グループ化されているセクションのグループ化を解除します。</p> <p>ただし、グループ化されたセクションを選択していない場合は、この項目は選択できません。</p>

## 【オブジェクト・ファイル】

[16進数表示 (H)]	チェックあり:[サイズ]を16進数で表示します。 チェックなし:[サイズ]を10進数で表示します(デフォルト)。
[切り取り (I)]	選択状態のオブジェクト・ファイルを切り取ります。
[コピー (C)]	選択状態のオブジェクト・ファイルをコピー・バッファにコピーします。
[貼り付け (P)]	オブジェクト・ファイルを貼り付けます。 なお、コピー・バッファにオブジェクト・ファイルがコピーされていない場合は、この項目は選択できません。
[新規オブジェクト ファイル選択 (E)]	[オブジェクト・ファイルを選択]ダイアログをオープンします。
[削除 (D)]	選択状態のオブジェクト・ファイルをセクションから削除します。

**ドラッグ・アンド・ドロップ機能の利用**

マウスにより、メモリ/セクションの各ボックス，またはオブジェクト・ファイル名をドラッグ・アンド・ドロップすることができます。

表 5-9 ドラッグ・アンド・ドロップ操作による編集

ドラッグ元	ドロップ先	動作
メモリ	メモリ	- ドラッグ元のメモリを削除し、ドロップ先のメモリの前に配置します。 ただし、内蔵 ROM / RAM をドロップすることはできません。
セクション	メモリ	- ドラッグ元のメモリからドラッグしたセクションを削除し、ドロップ先のメモリに割り当てます。
	セクション	- ドラッグ元のセクションをドロップ先のセクションの前に配置します。
オブジェクト・ファイル名	セクション	- ドロップ先が同じ属性のセクションの場合、ドラッグしたオブジェクト・ファイルを削除し、ドロップ先のセクションに割り当てます。 - ドロップ先が異なる属性のセクションの場合、ドラッグしたオブジェクト・ファイルは削除せず、ドロップ先のセクションに割り当てます。
	オブジェクト・ファイル名	- ドロップ先が同じ属性のオブジェクト・ファイルの場合、ドラッグしたオブジェクト・ファイルを削除し、ドロップ先のセクションに割り当てます。 - ドロップ先が異なる属性のオブジェクト・ファイルの場合、ドラッグしたオブジェクト・ファイルは削除せず、ドロップ先のセクションに割り当てます。 - 同一セクション上の別ファイル名上にドロップすると、ドラッグ元の位置からドロップ先の位置に移動します。なお、ここでの表示順は、リンクの際の解決順に影響します。
シンボル	セクション	- ドラッグ元のシンボルが、複数のセクションと関連付けて生成される場合、ドロップ先のセクションをシンボルの関連付けに追加します。 - ドラッグ元のシンボルが、単一のセクションと関連付けて生成される場合、ドロップ先のセクションにシンボルの関連付けを切り替えます。 【対象】tp シンボル，gp シンボル

**【注意】** tp シンボルは text 属性のセクションにのみ，gp シンボルは data 属性のセクションにのみドロップすることができます。

また，ep シンボルは，SIDATA / 内蔵 RAM 以外のセクションやメモリに関連付けてアドレスを解決することはできないため，ドロップすることはできません。

**【備考】** メモリ，セクション，およびオブジェクト・ファイルは複数選択が可能です。

[Ctrl] キーを押下しながらクリックすると，クリックした項目が順次選択状態となります。

また，いずれかの項目が選択状態の際に，[Shift] キーを押下しながらクリックすると，選択状態だった項目からクリックした項目までが選択状態となります。

## プロパティ・ビュー・エリア

メモリ・マッピング・ビュー・エリア内で選択しているアイテム（メモリ/セクション/オブジェクト・ファイル）の詳細情報を表示/編集するエリアです。

メモリ・マッピング・ビュー・エリアで表示されない詳細な設定をこのエリアで直接編集することができます。

図 5-8 プロパティ・ビュー・エリア（セクションを選択している場合）

項目	値
セクション名	.text
前セクション	.const
開始番地	?
終了番地	?
サイズ	? bytes
アクセス属性	命令コード
セクション・タイプ	初期値あり
アライン	4 bytes
最大サイズ	0x0010 0000
入力セクション	
オブジェクト・ファイル	
コメント	

<p><b>セクション名</b> このセクションの名称です。</p>
--

選択している項目に対しての説明、および注意事項が表示されます。

メモリ・マッピング・ビュー・エリア内で選択している各アイテムに対しての表示内容とその編集の可否については、次のとおりです。

表 5-10 プロパティ・ビュー・エリアの表示内容と編集の可否（メモリ選択時）

選択アイテム	表示内容	編集の可否
メモリ	メモリ名（1文字目は英字，2文字目以降は英数字を使用）	
	開始番地	
	終了番地	
	サイズ	
	種類（ROM / RAM / 空き領域 / メモリ配置不能領域）	
	アライン（1 byte / 2 bytes / 4 bytes / 8 bytes）	
	コメント	

**【注意】** 内蔵メモリ，空き領域，およびメモリ配置不能領域（SFR領域など）は，編集操作によって値を変更することはできません（内蔵メモリの情報はデバイス・ファイルから取得します）。

表 5-11 プロパティ・ビュー・エリアの表示内容と編集の可否 (セクション選択時)

選択アイテム	表示内容	編集の可否
セクション (グループ化していないもの)	セクション名	
	前セクション (前セクション名 / なし)	
	開始番地 確定しない場合は “ ? ” を表示します。	
	終了番地 確定しない場合は “ ? ” を表示します。	×
	サイズ 確定しない場合は “ ? ” を表示します。	×
	アクセス属性 (命令コード / リード・オンリー / リード / ライト可)	
	セクション・タイプ (初期値あり / 初期値なし)	
	アライン (1 byte / 2 bytes / 4 bytes / 8 bytes)	
	最大サイズ	
	入力セクション	
	オブジェクト・ファイル	×
	コメント	

表 5-12 プロパティ・ビュー・エリアの表示内容と編集の可否 (グループ選択時)

選択アイテム	表示内容	編集の可否
グループ (セクションをグループ化することにより、セグメント扱いとしているもの)	グループ名 ただし、CA850 の規定セグメントのグループ名については編集できません。	
	前セクション (前セクション名 / なし)	
	開始番地 確定しない場合は “ ? ” を表示します。	
	終了番地 確定しない場合は “ ? ” を表示します。	×
	サイズ 確定しない場合は “ ? ” を表示します。	×
	アクセス属性 (命令コード / リード・オンリー / リード / ライト可)	
	アライン (1 byte / 2 bytes / 4 bytes / 8 bytes)	×
	最大サイズ	
	セクション	×
	コメント	



表 5-13 プロパティ・ビュー・エリアの表示内容と編集の可否（オブジェクト・ファイル選択時）

選択アイテム	表示内容	編集の可否
オブジェクト・ファイル	ファイル名	×
	開始番地 確定しない場合は“？”を表示します。	×
	終了番地 確定しない場合は“？”を表示します。	×
	サイズ 確定しない場合は“？”を表示します。	×
	セクション・タイプ（初期値あり / 初期値なし）	×
	パス	×
	ライブラリ ライブラリ・ファイルに含まれたオブジェクト・ファイルでない場合は表示されません。	×
	コメント	

表 5-14 プロパティ・ビュー・エリアの表示内容と編集の可否（シンボル選択時）

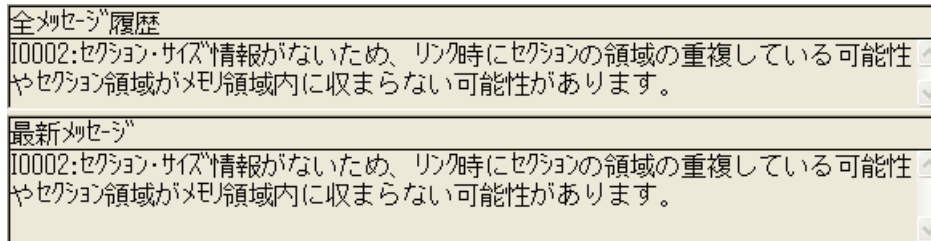
選択アイテム	表示内容	編集の可否
シンボル	シンボル名	
	シンボル種別（TP シンボル / GP シンボル / EP シンボル）	
	番地	
	アライン（1 byte / 2 bytes / 4 bytes / 8 bytes）	
	ベース・シンボル [シンボル種別]が“GP シンボル”以外の場合は表示されません。	
	参照セクション	×
	コメント	

## メッセージ・ビュー・エリア

ユーザに対するメッセージを表示するエリアです。

メッセージのすべてを履歴として表示する領域（全メッセージ履歴）と、最新のメッセージを表示する領域（最新メッセージ）とに分かれています（これらの領域は、マウスにより任意にサイズを変更することができます）。

図 5-9 メッセージ・ビュー・エリアの例



なお、このエリア上で右クリックすると、マウス・カーソル下の領域が選択状態となり、次のコンテキスト・メニューを表示します。

[クリア (C)]	表示しているメッセージを削除します。 なお、最新メッセージ領域を選択している場合は、この項目は選択できません。
[コピー (C)]	選択状態の文字列をコピー・バッファにコピーします。 なお、文字列を選択していない場合は、この項目は選択できません。
[検索 (E)]	メッセージ・ログから文字列を検索するための <a href="#">[検索] ダイアログ</a> をオープンします。

メニューバー

## (1) [ファイル(F)] メニュー

[新規作成 (N)...]	<p>リンク・ディレクティブを新規に作成するために <b>[新しいリンク・ディレクティブ]</b> ダイアログをオープンします。</p> <p>編集中の場合は、ファイルに保存するか否かを確認するメッセージを表示します。なお、この際に “保存する” を選択した場合は、<b>[上書き保存]</b> と同じ動作をします。</p> <p>【ショートカット・キー】: [Ctrl]+[N]</p>
[開く (O)...]	<p><b>[開く]</b> ダイアログをオープンします。</p> <p>なお、LDG が生成したリンク・ディレクティブ・ファイルではない場合、またはリンク・ディレクティブ・ファイルにコメントを記述できないコンパイラのリンク・ディレクティブ・ファイルの場合は、<b>[環境選択]</b> ダイアログをオープンします。</p> <p>ただし、LDG を PM+ から起動している場合には、PM+ 上で設定している環境でリンク・ディレクティブ・ファイルを開きます (LDG のファイルの読み込みに関するの詳細は、<b>[開く]</b> ダイアログを参照してください)。</p> <p>編集中の場合は、ファイルに保存するか否かを確認するメッセージを表示します。なお、この際に “保存する” を選択した場合は、<b>[上書き保存]</b> と同じ動作をします。</p> <p>【ショートカット・キー】: [Ctrl]+[O]</p>
[上書き保存 (S)]	<p>ファイルを上書き保存します。</p> <p>ただし、新規作成後に一度も保存していない場合には、<b>[名前を付けて保存]</b> と同じ動作をします。</p> <p>【ショートカット・キー】: [Ctrl]+[S]</p>
[名前を付けて保存 (A)...]	<p><b>[名前を付けて保存]</b> ダイアログをオープンします。</p> <p>保存するファイルの形式は、*.ind となります。</p>
[オブジェクト・ファイルを選択 (B)...]	<p>リンクするオブジェクト・ファイル (複数選択可) / 実行ファイル選択するために <b>[オブジェクト・ファイルを選択]</b> ダイアログをオープンします。</p> <p>読み込み可能なファイルの種類は次のとおりです。</p> <ul style="list-style-type: none"> <li>・実行ファイル (*.out)</li> <li>・オブジェクト・ファイル (*.o)</li> <li>・ライブラリ (*.a)</li> </ul> <p>編集中の場合は、ファイルに保存するか否かを確認するメッセージを表示します。なお、この際に “保存する” を選択した場合は、<b>[上書き保存]</b> と同じ動作をします。</p>
[履歴ファイル名]	<p>使用したファイル名の履歴を最大 4 個まで表示します。</p> <p>ファイル名を選択することにより、そのファイルをオープンします。</p>
[アプリケーションの終了 (X)]	LDG を終了します。

(2) [編集 (E)] メニュー

[元に戻す (U)]	直前の編集操作を戻します。 ただし、元に戻せない場合、この項目は選択できません。 【ショートカット・キー】: [Ctrl]+[Z]	
[切り取り (I)]	選択を切り取ります。 【ショートカット・キー】: [Ctrl]+[X]	
[コピー (C)]	選択をコピー・バッファにコピーします。 コピー元の名前情報は失われます。 【ショートカット・キー】: [Ctrl]+[C]	
[貼り付け (P)]	コピー・バッファの内容を選択箇所に貼り付けます。 貼り付けるメモリに名前がない場合は、名前を “NewMemory” として貼り付けます。 貼り付けるセクションに名前がない場合は、名前を “NewSection. セクションの種類” として貼り付けます。 すでに同じ名前が存在する場合は、名前の後ろに 10 進数の番号 (0 ~ 4294967295) を付与します。 【ショートカット・キー】: [Ctrl]+[V]	
[追加 (A)]	メモリ・マッピング・ビュー・エリアにアイテムを追加するために、次のカスケード・メニューを表示します。 なお、オブジェクト・ファイルの追加は、[ファイル]メニュー [オブジェクト・ファイルを選択] により行ってください。	
	[メモリ (M)]	メモリを追加するために、[メモリを追加] ダイアログをオープンします。
	[セクション (S)]	セクションを追加するために、[セクションを追加] ダイアログをオープンします。
	[シンボル (Y)]	シンボルを追加するために、[シンボルを追加] ダイアログをオープンします。
[削除 (D)]	選択を削除します。 【ショートカット・キー】: [Del]	
[グループ化 (G)]	[グループ化 (G)]	選択したセクションをグループ化します。
	[グループ化解除 (U)]	グループ化されているセクションのグループを解除します。
[結合 (J)]	選択したメモリを結合し、1つのメモリとして扱います。	
[検索 (E)...]	セクションやオブジェクト・ファイル、またはメッセージ・ログ内の文字列などを検索するために [検索] ダイアログをオープンします。 【ショートカット・キー】: [Ctrl]+[F]	

(3) [表示 (V)] メニュー

[ツール・バー (I)]	チェックあり：ツールバーを表示します（デフォルト） チェックなし：ツールバーを表示しません。
[サイズを 16 進数で表示 (X)]	チェックあり：メモリ・サイズを 16 進数で表示します。 チェックなし：メモリ・サイズを 10 進数で表示します（デフォルト）
[上下反転表示 (D)]	チェックあり：開始番地を下にしてメモリ・マップを表示します。 チェックなし：開始番地を上にしてメモリ・マップを表示します（デフォルト）
[ミラー・イメージを表示 (M)]	チェックあり：ミラー・イメージを表示します（デフォルト） チェックなし：ミラー・イメージを表示しません。
[全メモリ空間表示 (A)]	チェックあり：すべてのメモリ空間をリニアに表示します。セクションは、 配置しているアドレスのみに表示されます。 チェックなし：ミラー・イメージの展開/折りたたみが可能です（デフォルト）
[メッセージをクリア (C)]	メッセージ・ビュー・エリアに表示しているメッセージをクリアします。

【備考】チェックの状態は、LDG 終了時に保存され、次回起動時に反映されます。

(4) [ツール (I)] メニュー

[オプション (O)...]	LDG の各種設定（表示フォント/メモリ・マッピングの表示色など）を行うために、 <a href="#">[オプション]ダイアログ</a> をオープンします。
----------------	---








(5) [ヘルプ (H)] メニュー

[LDG のヘルプ (H)]	LDG のオンライン・ヘルプをオープンします。 【ショートカット・キー】: F1
[NEC Electronics マイコン Web(N)]	NEC エレクトロニクス製マイクロコンピュータ関連の Web サイトをオープンします。
[V850 ツール FAQ(V)]	V850 マイクロコントローラ用開発環境 FAQ の Web サイトをオープンします。
[バージョン情報 (A)]	LDG のバージョン情報を表示します。 “LDG バージョン番号 [日 月 年]” の順で表示します。

## ツールバー

比較的、使用頻度の高いメニュー項目をワン・アクションで実行可能にしたボタン群です。

表 5-15 メイン・ウィンドウのツールバー

アイコン	機能
	[ファイル]メニュー [新規作成 ...]の選択と同等の機能です。
	[ファイル]メニュー [開く ...]の選択と同等の機能です。
	[ファイル]メニュー [上書き保存]の選択と同等の機能です。
	[編集]メニュー [切り取り]の選択と同等の機能です。
	[編集]メニュー [コピー]の選択と同等の機能です。
	[編集]メニュー [貼り付け]の選択と同等の機能です。
	[編集]メニュー [追加]の選択と同等の機能です。

## [新しいリンク・ディレクティブ]ダイアログ

新しいリンク・ディレクティブを生成するために、ターゲット・デバイス、コンパイラ、およびリアルタイム OS の情報を設定するダイアログです。

このダイアログは、次のいずれかの操作でオープンします。


- [ファイル]メニュー [新規作成 ...] を選択
-  ボタンをクリック

図 5-10 [新しいリンク・ディレクティブ]ダイアログ



ここでは、次の項目について説明します。

[各エリアの説明](#)

[機能ボタン](#)

## 各エリアの説明

### (1) デバイス名

使用するデバイス名をプルダウン・メニューより選択します。

これにより、LDG はアドレス空間 / 内蔵メモリ領域などを決定します。

なお、この項目の指定を省略することはできません。

### (2) コンパイラ名

使用するコンパイラ名をプルダウン・メニューより選択します。

ただし、今回の版では、“CA850” 固定となります。

なお、この項目の指定を省略することはできません。

### (3) RTOS

使用するリアルタイム OS 名をプルダウン・メニューより選択します。

これにより、LDG は必要なセクションを追加します。

なお、リアルタイム OS を使用しない場合は“使用しない”を選択してください。

【備考】PM+ より起動した場合、各項目は、プロジェクトの設定を反映した状態で表示します。

## 機能ボタン

表 5-16 [新しいリンク・ディレクティブ] ダイアログの機能ボタン

ボタン	機能
OK	指定した内容で新しいリンク・ディレクティブ・ファイルを作成します。
キャンセル	設定を無視してダイアログをクローズします。
ヘルプ(H)	このダイアログのオンライン・ヘルプを表示します。



## [環境選択] ダイアログ

既存のリンク・ディレクティブ・ファイル (LDG が生成したものではないリンク・ディレクティブ・ファイル、または [名前を付けて保存] ダイアログ上の [LDG 情報出力する] 項目のチェックを外して保存したリンク・ディレクティブ・ファイル) を開いた場合に、ターゲット・デバイスとコンパイラ情報を設定するダイアログです。

このダイアログは、次のいずれかの操作でオープンします。


- [ファイル] メニュー [開く ...] の選択後、既存のリンク・ディレクティブ・ファイルを指定した場合
-  ボタンをクリック後、既存のリンク・ディレクティブ・ファイルを指定した場合

図 5-11 [環境選択] ダイアログ



ここでは、次の項目について説明します。

[各エリアの説明](#)

[機能ボタン](#)

## 各エリアの説明

### (1) デバイス名

使用するデバイス名をプルダウン・メニューより選択します。

これにより、LDG はアドレス空間 / 内蔵メモリ領域などを決定します。

なお、この項目の指定を省略することはできません。

### (2) コンパイラ名


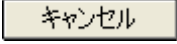
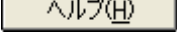
使用するコンパイラ名をプルダウン・メニューより選択します。

ただし、今回の版では、“CA850” 固定となります。

なお、この項目の指定を省略することはできません。

## 機能ボタン

表 5-17 [環境選択] ダイアログの機能ボタン

ボタン	機能
	指定した内容で新しいリンク・ディレクティブ・ファイルを作成します。
	設定を無視してダイアログをクローズします。
	このダイアログのオンライン・ヘルプを表示します。

## [開く] ダイアログ

新しく開くファイルを選択するダイアログです。

このダイアログは、次のいずれかの操作でオープンします。


- [ファイル]メニュー [開く...]を選択
-  ボタンをクリック

図 5-12 [開く] ダイアログ



ここでは、次の項目について説明します。

[各エリアの説明](#)

[機能ボタン](#)

[操作上の注意事項](#)

## 各エリアの説明

### (1) ファイルの場所

指定ファイルの存在するドライブ，またはフォルダをドロップダウン・リストから選択します。エリアの下欄には，指定場所にあるファイルが表示されます。

### (2) ファイル名

指定するファイル名をキーボードから入力します。  
上欄から選択した場合は，選択したファイル名が表示されます。

### (3) ファイルの種類

次のファイルの種類を選択できます。


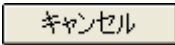
- ・リンク・ディレクティブ・ファイル (\*.lnd , \*dir)

*.lnd	LDG がコメントを含めて保存したファイル
*.dir	CA850 で標準としているリンク・ディレクティブ・ファイル RX850 Pro のサンプルで使用しているリンク・ディレクティブ・ファイル

**【注意】** 拡張子 “.dr” のファイルを選択した場合，ファイルの内容が CA850 のリンク・ディレクティブ・ファイルである場合は該当ファイルを開くことができますが，異なる場合はエラー・メッセージを表示し，該当ファイルを開くことはできません。

## 機能ボタン

表 5-18 [開く] ダイアログの機能ボタン

ボタン	機能
	指定したファイルを開き，このダイアログをクローズします。
	設定を無視してダイアログをクローズします。

## 操作上の注意事項

リンク・ディレクティブ・ファイルを開く際，LDG が生成したリンク・ディレクティブ・ファイルではないファイルを指定した場合，[\[環境選択\] ダイアログ](#)がオープンします（このダイアログ上で，あらたに開発環境の設定を行います）。

ただし，LDG を PM+ から起動している場合には，PM+ で設定している環境でリンク・ディレクティブ・ファイルを開きます。

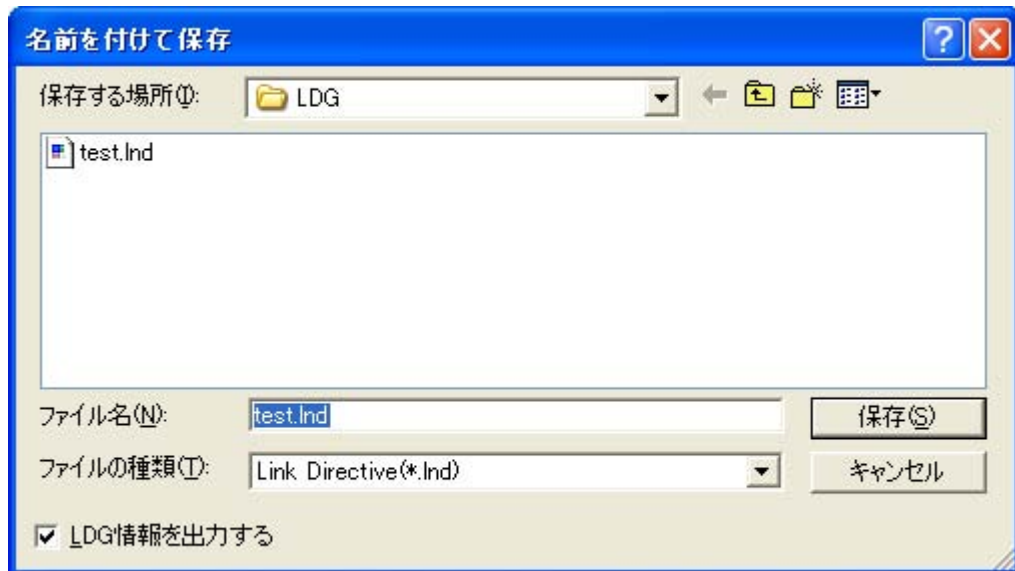
## [名前を付けて保存] ダイアログ

指定したファイルに名前を付けて保存するダイアログです。

このダイアログは、次の操作でオープンします。

- ・[ファイル]メニュー [名前を付けて保存...]を選択

図 5-13 [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。

[各エリアの説明](#)

[機能ボタン](#)

### 各エリアの説明

#### (1) 保存する場所

ファイルを保存するドライブ、またはフォルダをドロップダウン・リストから選択します。エリアの下欄には、指定場所にあるファイルが表示されます。

#### (2) ファイル名

保存するファイル名をキーボードから入力します。  
上欄から選択した場合は、選択したファイル名が表示されます。

#### (3) ファイルの種類

保存するファイルの種類をドロップダウン・リストから指定します。  
ただし、LDG で保存できるファイルの種類は、\*.lnd のみです。

(4) LDG 情報を出力する

必要に応じて、チェック・ボックスにより指定します。

<p>チェックあり</p>	<p>LDG 固有の情報を出力し、リンク・ディレクティブ・ファイルとして保存します（デフォルト） この方法で保存したファイルを次回 LDG で読み込んだ場合、保存前の情報がすべて復元されます。</p>
<p>チェックなし</p>	<p>リンク・ディレクティブ・ファイルとしての可読性を上げるため、LDG 固有の情報を出力せずに保存します。 ただし、この方法で保存したファイルを次回 LDG で読み込んだ場合、次の注意が必要となります。</p> <ul style="list-style-type: none"> <li>- メモリ情報が元の状態に復元しません。</li> <li>- デバイス情報が元の状態に復元しません（再選択が必要）</li> <li>- 保存前に追加したセクションで、グループ化（セグメント）していないものに対しては、保存の際に自動的にセグメントを補うため、そのセグメントが読み込まれ表示されます。</li> </ul> <p>【例】</p> <div style="display: flex; align-items: flex-start;"> <div style="border: 1px solid black; padding: 5px; margin-right: 20px;"> <p>↑ ?</p> <p>.text</p> <p style="text-align: right;">? bytes</p> <p>?</p> </div> <div style="border: 1px solid black; padding: 5px;"> <p>ファイル出力</p> <pre>SEGMENT.text : !LOAD ?RX H0x0 F0x0 A0x8 {     .text = \$PROGBITS ?AX A0x8; };</pre> </div> </div> <p style="text-align: center;">再読み込み時の表示</p> <div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 20px;"> <p>↑ ?</p> <p>+ SEGMENT.text</p> <p style="text-align: right;">? bytes</p> <p>?</p> <p style="text-align: center;">“+”をクリック</p> </div> <div style="margin-right: 20px;">→</div> <div style="border: 1px solid black; padding: 5px;"> <p>↑ ?</p> <p>- SEGMENT.text</p> <p style="text-align: right;">? bytes</p> <div style="border: 1px solid black; padding: 5px; margin-top: 5px;"> <p>↑ ?</p> <p>.text</p> <p style="text-align: right;">? bytes</p> <p>?</p> </div> <p>?</p> </div> </div>

機能ボタン

表 5-19 [名前を付けて保存ダイアログ]の機能ボタン

ボタン	機能
<p>保存(S)</p>	<p>指定したファイル名でファイルを保存し、このダイアログをクローズします。</p>
<p>キャンセル</p>	<p>設定を無視してダイアログをクローズします。</p>

## [ 検索 ] ダイアログ

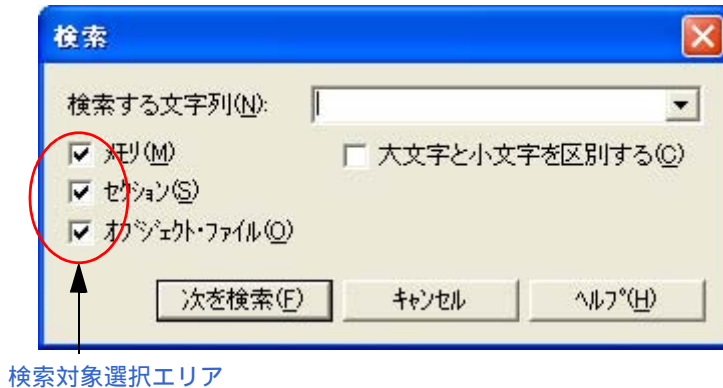
メモリ・マッピング・ビュー・エリアに表示しているメモリ名 / セクション名 / オブジェクト・ファイル名から文字列を検索したり、メッセージ・ビュー・エリア内のメッセージ内から文字列を検索するダイアログです。

指定文字列が存在した場合、検索箇所を選択状態にして、エリア内に表示します。

このダイアログは、次のいずれかの操作でオープンします。

- [編集]メニュー [検索]を選択
- メッセージ・ビュー・エリア上で右クリック  
[検索]コンテキスト・メニューを選択

図 5-14 [ 検索 ] ダイアログ



ここでは、次の項目について説明します。

[各エリアの説明](#)

[機能ボタン](#)

## 各エリアの説明

### (1) 検索する文字列

検索する文字列を指定します。

なお、起動後に検索した最新5個までの文字列をプルダウン・メニューにより選択することができます。

### (2) 検索対象選択エリア

メモリ・マッピング・ビュー・エリアを選択している場合にのみ表示される選択エリアです。

検索の対象とするアイテム（メモリ/セクション/オブジェクト・ファイル）を選択します。

チェックした項目のみが検索対象となります。

なお、起動後の初回オープン時はすべての項目がチェックされていますが、次回オープン時は、前回の状態が保持されます。

### (3) 大文字と小文字を区別する

チェックすると、[検索する文字列]で指定された文字列の大文字/小文字がすべて一致する場合のみ検索します。

## 機能ボタン

表 5-20 [検索] ダイアログの機能ボタン

ボタン	機能
次を検索(F)	指定した文字列を検索します。
キャンセル	設定を無視してダイアログをクローズします。
ヘルプ(H)	このダイアログのオンライン・ヘルプを表示します。



## [オブジェクト・ファイルを選択] ダイアログ

新しくオブジェクト・ファイルを追加するために、オブジェクト・ファイルを選択するダイアログです。  
このダイアログは、次のいずれかの操作でオープンします。

- [ファイル]メニュー [オブジェクト・ファイルを選択...] を選択
- [メモリ・マッピング・ビュー・エリア](#)のセクション上で右クリック  
[新規オブジェクトファイル追加]コンテキスト・メニューを選択

図 5-15 [オブジェクト・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。

[各エリアの説明](#)

[機能ボタン](#)

## 各エリアの説明

### (1) ファイルの場所

指定ファイルの存在するドライブ，またはフォルダをドロップダウン・リストから選択します。エリアの下欄には，指定場所にあるファイルが表示されます。

### (2) ファイル名

指定するファイル名をキーボードから入力します。

上欄から選択した場合は，選択したファイル名が表示されます（オブジェクト・ファイルについては，複数選択することができます）。


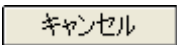
### (3) ファイルの種類

次のファイルの種類を選択できます。

- ・実行ファイル（\*.out）
- ・オブジェクト・ファイル（\*.o）
- ・ライブラリ（\*.a）

## 機能ボタン

表 5-21 [オブジェクト・ファイルを選択] ダイアログの機能ボタン

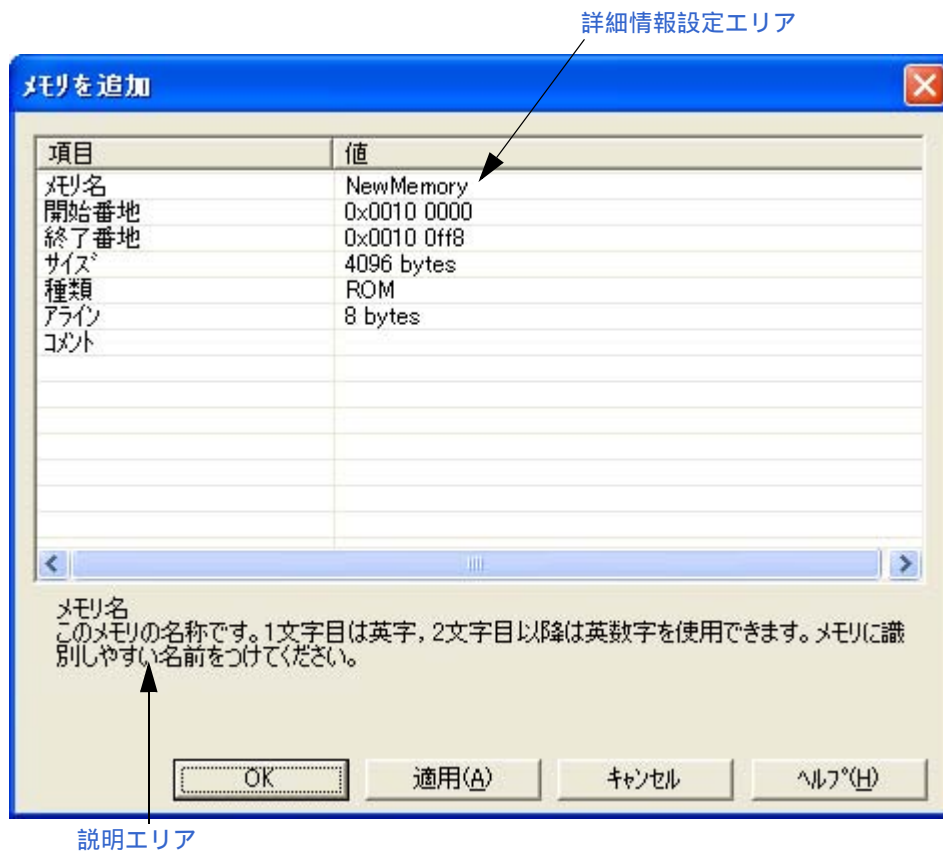
ボタン	機能
	指定したオブジェクト・ファイルを追加し，このダイアログをクローズします。
	設定を無視してダイアログをクローズします。

## [メモリを追加] ダイアログ

メイン・ウィンドウ上のメモリ・マッピング・ビュー・エリアに新規にメモリを追加するダイアログです。  
このダイアログは、次のいずれかの操作でオープンします。

- [編集]メニュー [追加] [メモリ]を選択
- メモリ・マッピング・ビュー・エリアのメモリ上で右クリック  
[新規メモリ追加]コンテキスト・メニューを選択

図 5-16 [メモリを追加] ダイアログ



ここでは、次の項目について説明します。

[各エリアの説明](#)

[機能ボタン](#)

## 各エリアの説明

### (1) 詳細情報設定エリア

次に示す項目の値をそれぞれ指定します。

#### (a) メモリ名

追加するメモリの名前を指定します。

1文字目は英字, 2文字目以降から英数字を使用することができます。

デフォルトでは, “NewMemory” が指定されます。すでに同名のメモリが存在する場合には, 名前の後ろに10進数の番号(0 ~ 4294967295)を付与します。

#### (b) 開始番地

追加するメモリの開始番地を “0x” を接頭辞とした16進数, または10進数で指定します。

**メモリ・マッピング・ビュー・エリア**で, メモリを選択している場合は, 選択しているメモリの直前に追加するアドレスがデフォルトで指定されます。

また, **メモリ・マッピング・ビュー・エリア**でメモリを選択していない場合では, 空き領域の先頭に追加するアドレスがデフォルトで指定されます。

なお, 値の編集の際, 次の値が指定された場合は, 不正値入力としてメッセージを表示します。

- ・アドレス桁を越えた場合
- ・[開始番地] が [終了番地] より大きかった場合
- ・指定したアドレスが他のメモリに重なった場合

#### (c) 終了番地

追加するメモリの終了番地を “0x” を接頭辞とした16進数, または10進数で指定します。

**メモリ・マッピング・ビュー・エリア**で, メモリを選択している場合は, 選択しているメモリの直前に追加するアドレスがデフォルトで指定されます。

また, **メモリ・マッピング・ビュー・エリア**でメモリを選択していない場合では, 空き領域の先頭に追加するアドレスがデフォルトで指定されます。

サイズの値が変更された場合, その値に応じて終了番地は自動的に変更されます。

なお, 値の編集の際, 次の値が指定された場合は, 不正値入力としてメッセージを表示します。

- ・アドレス桁を越えた場合
- ・[終了番地] が [開始番地] より小さかった場合
- ・指定したアドレスが他のメモリに重なった場合

#### (d) サイズ

追加するメモリのサイズを “0x” を接頭辞とした16進数, または10進数で指定します。

デフォルトでは, “0x1000 bytes” が指定されます。ただし, 空き領域が0x1000バイトに満たない場合では, 空き領域のサイズが指定されます。

[開始番地] / [終了番地] の値が変更された場合, その値に応じてサイズは自動的に変更されます。

#### (e) 種類

追加するメモリの種類として, ROM / RAM のいずれかを指定します。

デフォルトでは, “ROM” が指定されます。

**(f) アライン**

アラインとして、1 byte / 2 bytes / 4 bytes / 8 bytes のいずれかを指定します。

デフォルトでは、“8 bytes” が指定されます。

**(g) コメント**

コメントを指定します。

デフォルトでは、空欄です。

**(2) 説明エリア**

[詳細情報設定エリア](#)で選択している項目の説明を表示します。

**機能ボタン**

表 5-22 [メモリを追加] ダイアログの機能ボタン

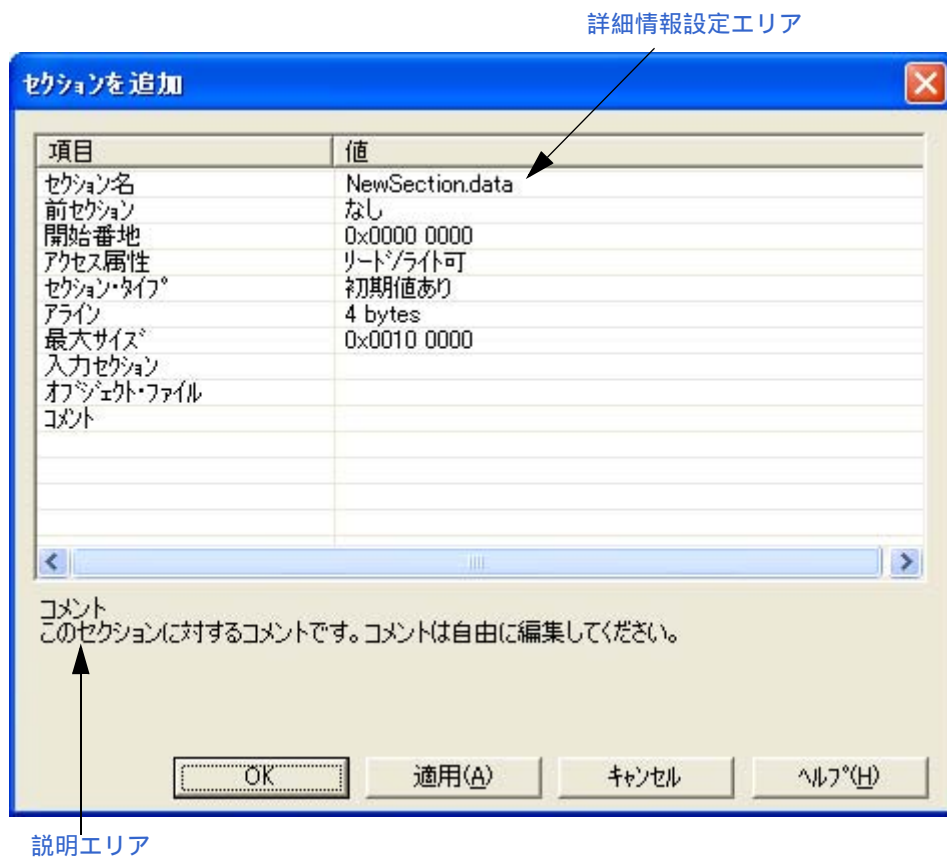
ボタン	機能
OK	指定した内容で新規にメモリを追加し、このダイアログをクローズします。
適用(A)	指定した内容で新規にメモリを追加し、このダイアログはクローズしません。
キャンセル	指定した内容を無効にします。
ヘルプ(H)	このダイアログのオンライン・ヘルプを表示します。

## [セクションを追加] ダイアログ

メイン・ウィンドウ上のメモリ・マッピング・ビュー・エリアに新規にセクションを追加するダイアログです。このダイアログは、次のいずれかの操作でオープンします。

- [編集]メニュー [追加] [セクション]を選択
- メモリ・マッピング・ビュー・エリアのメモリ/セクション上で右クリック  
[新規セクション追加]コンテキスト・メニューを選択

図 5-17 [セクションを追加] ダイアログ



ここでは、次の項目について説明します。

各エリアの説明

機能ボタン

## 各エリアの説明

### (1) 詳細情報設定エリア

次に示す項目の値をそれぞれ指定します。

#### (a) セクション名

追加するセクションの名前を指定します。

メモリ・マッピング・ビュー・エリアでの選択アイテムにより、デフォルトで次の名前が指定されます。

選択アイテム	名前 (デフォルト)
メモリ (RAM)	NewSection.data
メモリ (ROM)	NewSection.text
セクション	NewSection. 選択している一番先頭のセクションの種類
なし	NewSection.data

なお、すでに同名のセクションが存在する場合には、名前の後ろに 10 進数の番号 (0 ~ 4294967295) を付与します。

#### (b) 前セクション

追加するセクションの前のセクションを指定します。

メモリ・マッピング・ビュー・エリアで、連続したセクションの先頭セクション以外を選択している場合は、前のセクションがデフォルトで指定されます。

それ以外の場合は、デフォルトで空欄です。

また、[開始番地]を変更した場合、この項目の値は“なし”になります。

#### (c) 開始番地

追加するセクションの開始番地を“0x”を接頭辞とした 16 進数、または 10 進数で指定します。

メモリ・マッピング・ビュー・エリアで、メモリを選択している場合は、選択しているメモリ内の空き領域の先頭に追加するアドレスがデフォルトで指定されます。

また、メモリ・マッピング・ビュー・エリアで連続したセクションの先頭セクションを選択している場合で、そのセクションの前に空き領域がある場合は、選択しているセクションの直前に追加するアドレスがデフォルトで指定されます。

それ以外の場合では、デフォルトで空欄となります。

また、[前セクション]を変更した場合、この項目の値は“?”になります。

なお、値の編集の際、次の値が指定された場合は、不正値入力としてメッセージを表示します。

- アドレス桁を越えた場合
- 指定したアドレスが他のセクションに重なった場合
- メモリのないアドレスの場合

#### (d) アクセス属性

アクセス属性として、命令コード/リード・オンリー/リード/ライト可のいずれかを指定します。

メモリ・マッピング・ビュー・エリアでの選択アイテムにより、デフォルトで次のアクセス属性が指定されません。

選択アイテム	アクセス属性 (デフォルト)
メモリ (RAM)	リード/ライト可
メモリ (ROM)	命令コード
セクション	選択している一番先頭のアクセス属性

**(e) セクション・タイプ**

セクションの種類として、初期値あり / 初期値なしのいずれかを指定します。

[メモリ・マッピング・ビュー・エリア](#)での選択アイテムにより、デフォルトで次のセクション・タイプが指定されます。

選択アイテム	セクション・タイプ (デフォルト)
メモリ	初期値あり
セクション	選択している一番先頭のセクション・タイプ

**(f) アライン**

アラインとして、1 byte / 2 bytes / 4 bytes / 8 bytes のいずれかを指定します。

[メモリ・マッピング・ビュー・エリア](#)でメモリを選択している場合は、メモリのアラインと同じ値がデフォルトで指定されます。

**(g) 最大サイズ**

追加するセクションの最大サイズを“0x”を接頭辞とした16進数、または10進数で指定します。

デフォルトでは、“0x100000 bytes”が指定されます。

なお、アドレス桁を越えた値が入力された場合は、不正値入力としてメッセージを表示します。

**(h) 入力セクション**

入力セクションを指定します。

デフォルトでは、空欄です。

**(i) オブジェクト・ファイル**

このセクションと同じ属性を持つセクションを指定されたオブジェクト・ファイルから出力します。

**(j) コメント**

コメントを指定します。

デフォルトでは、空欄です。

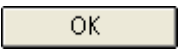


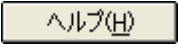
**(2) 説明エリア**

[詳細情報設定エリア](#)で選択している項目の説明を表示します。



## 機能ボタン

表 5-23 [ セクションを追加 ] ダイアログの機能ボタン

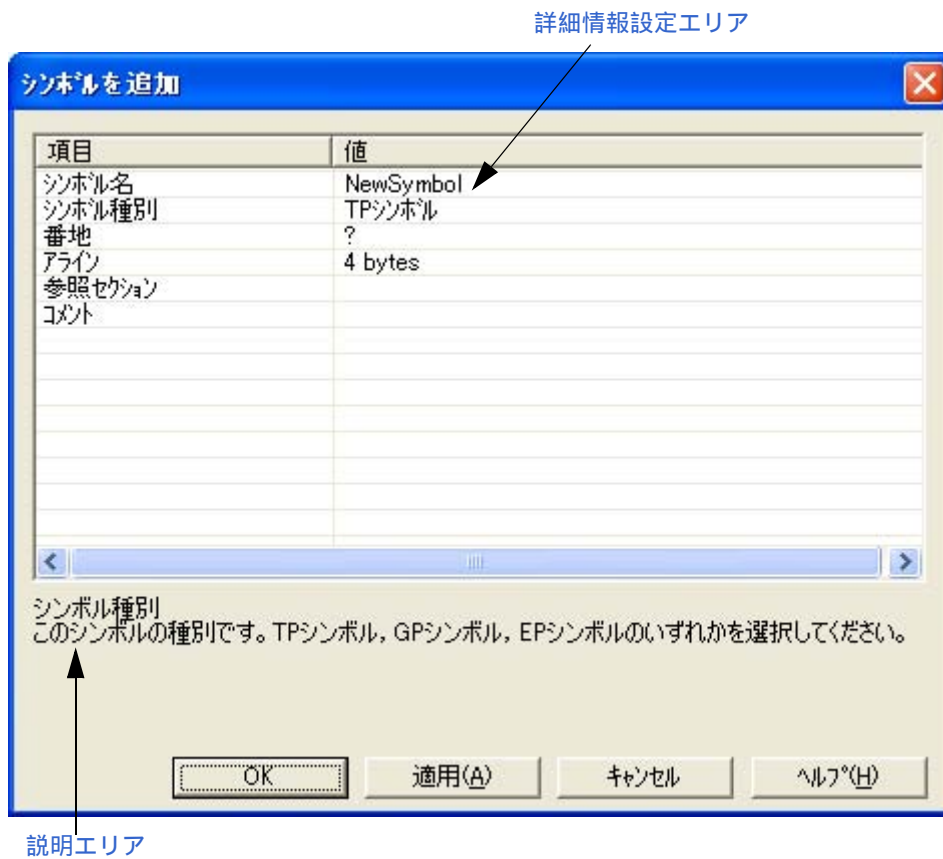
ボタン	機能
	指定した内容で新規にセクションを追加し、このダイアログをクローズします。
	指定した内容で新規にセクションを追加し、このダイアログはクローズしません。
	指定した内容を無効にします。
	このダイアログのオンライン・ヘルプを表示します。

## [シンボルを追加] ダイアログ

メイン・ウィンドウ上のメモリ・マッピング・ビュー・エリアに新規にシンボルを追加するダイアログです。このダイアログは、次のいずれかの操作でオープンします。

- [編集]メニュー [追加] [シンボル]を選択
- メモリ・マッピング・ビュー・エリアのメモリ/セクション上で右クリック  
[新規シンボル追加]コンテキスト・メニューを選択

図 5-18 [シンボルを追加] ダイアログ



ここでは、次の項目について説明します。

各エリアの説明

機能ボタン

## 各エリアの説明

### (1) 詳細情報設定エリア

次に示す項目の値をそれぞれ指定します。

#### (a) シンボル名

追加するシンボルの名前を指定します。

デフォルトでは、“NewSymbol”が指定されます。

すでに同名のシンボルが存在する場合には、名前の後ろに10進数の番号(0～4294967295)を付与します。

#### (b) シンボル種別

追加するシンボルとして、TPシンボル / EPシンボル / GPシンボルのいずれかを指定します。

デフォルトでは、“TPシンボル”が指定されます。

**【注意】** epシンボルは複数生成することはできません。

#### (c) 番地

シンボルの配置アドレスを指定します。

デフォルトでは、“?”が表示されます。

#### (d) アライン

アラインとして、1 byte / 2 bytes / 4 bytes / 8 bytes のいずれかを指定します。

デフォルトでは、“4 bytes”が指定されます。

#### (e) ベース・シンボル

ベース・シンボルを指定します。

ベース・シンボルの指定では、すべてのTPシンボルを選択することができます。

なお、この項目は[シンボル種別]で“GPシンボル”を指定した場合にのみ表示されます。

#### (f) 参照セクション

作成するシンボルの参照対象とするセクション名の一覧を表示します。

#### (g) コメント

コメントを指定します。


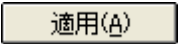
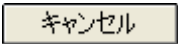
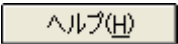
デフォルトでは、空欄です。

### (2) 説明エリア

[詳細情報設定エリア](#)で選択している項目の説明を表示します。

## 機能ボタン

表 5-24 [シンボルを追加] ダイアログの機能ボタン

ボタン	機能
	指定した内容で新規にシンボルを追加し、このダイアログをクローズします。
	指定した内容で新規にシンボルを追加し、このダイアログはクローズしません。
	指定した内容を無効にします。
	このダイアログのオンライン・ヘルプを表示します。

## [オプション]ダイアログ

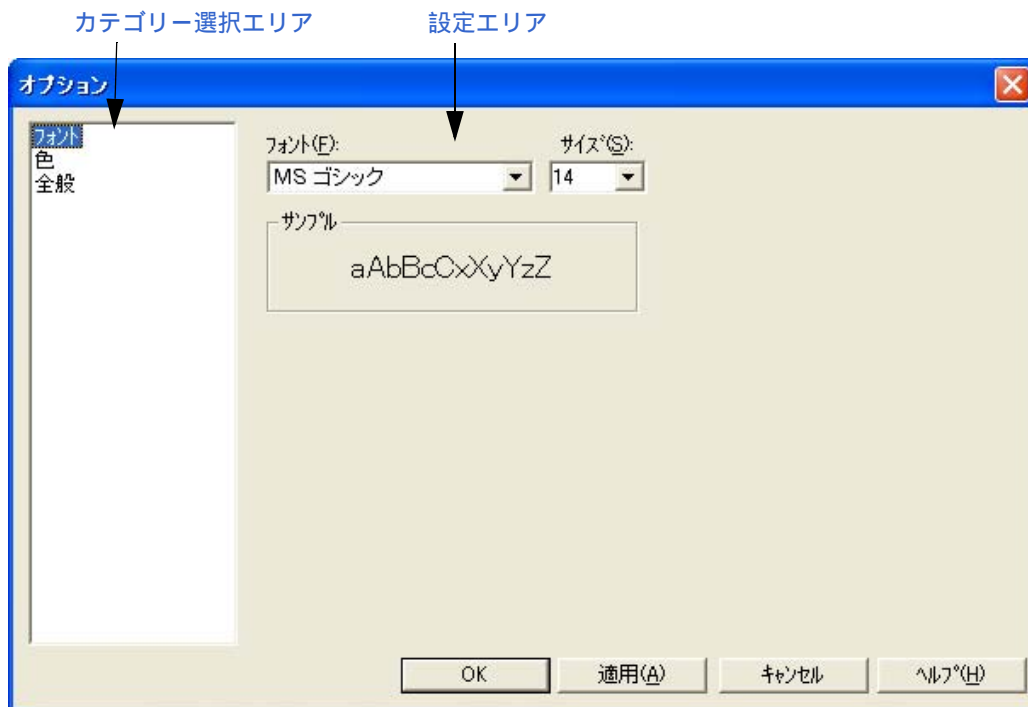
LDG の動作や表示に関する基本設定を行うダイアログです。

なお、次回 LDG の起動では、このダイアログで設定した状態（前回終了時の設定状態）で起動します。

このダイアログは、次のいずれかの操作でオープンします。

- [ツール]メニュー [オプション]を選択

図 5-19 [オプション]ダイアログ（[フォント]を選択している場合）



ここでは、次の項目について説明します。

[各エリアの説明](#)

[機能ボタン](#)

## 各エリアの説明

### (1) カテゴリー選択エリア

次のカテゴリーから、設定したいカテゴリーを選択します。

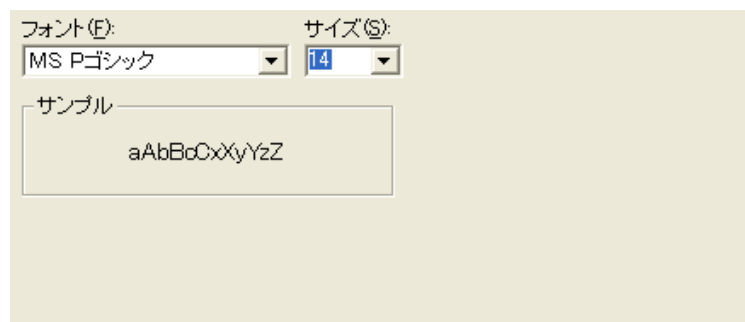
表 5-25 [オプション] ダイアログのカテゴリー

カテゴリー	設定内容
フォント	メモリ・マッピング・ビュー・エリアで表示するフォントやサイズを変更します。
色	メモリ・マッピング・ビュー・エリアで表示するアイテム（メモリ/セクションなど）の色設定を変更します。
全般	ポップ・アップ表示時間やミラー・イメージの表示/非表示などの設定を変更します。

### (2) 設定エリア

#### (a) [フォント]を選択した場合

図 5-20 [オプション] ダイアログでの [フォント] の設定



#### [フォント]

メモリ・マッピング・ビュー・エリアで表示するフォントをプルダウン・メニューにより選択します。

#### [サイズ]

メモリ・マッピング・ビュー・エリアで表示するフォント・サイズをプルダウン・メニューにより選択します。

#### [サンプル]

[フォント]と[サイズ]で指定した内容を元に、メモリ・マッピング・ビュー・エリアで表示する文字のサンプルを表示します。

## (b) [色] を選択した場合

図 5-21 [オプション] ダイアログでの [色] の設定

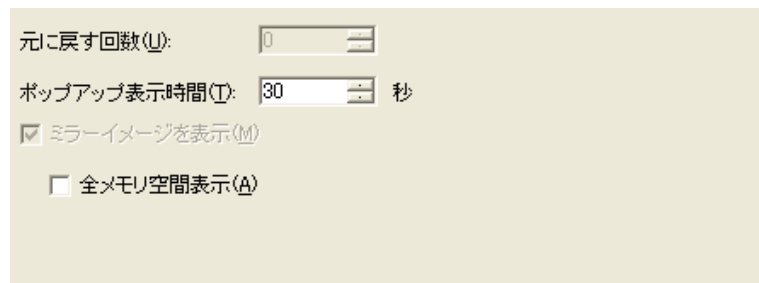


メモリ・マッピング・ビュー・エリアで表示するアイテム（メモリ / セクション）の背景色と文字色をプルダウン・メニューにより選択します。

各項目ごとのプルダウン・メニュー内の [ユーザ定義] を選択すると、Windows の [色の設定] ダイアログがオープンし、任意の色を選択 / 作成することができます。

## (c) [全般] を選択した場合

図 5-22 [オプション] ダイアログでの [全般] の設定



## [元に戻す回数]

今回の版ではサポートしていません。

## [ポップアップ表示時間]

メモリ・マッピング・ビュー・エリアで各アイテムの詳細表示を行うポップ・アップ表示の時間をプルダウン・メニューにより指定します。

デフォルトでは、“30” 秒が指定されています。

0 ~ 4294967295 までの値を指定することができます。

**[ミラーイメージを表示する]**

メモリ・マッピング・ビュー・エリアでミラー・イメージの表示 / 非表示を指定します。  
 ターゲット・デバイスがミラー・イメージ機能を持っている場合に有効となる項目です。

- チェックあり      ミラー・イメージの表示を行います (デフォルト)
- チェックなし     ミラー・イメージの表示を行いません。

**[全メモリ空間表示]**

[ミラーイメージを表示する] をチェックしている場合のみ有効となる項目です。

- チェックあり      すべてのメモリ空間をリニアに表示します。  
 セクションは、配置しているアドレスのみに表示され、ミラーリングしているアドレスには表示されません。
- チェックなし     ミラー・イメージの展開 / 折りたたみが可能です (デフォルト)

**機能ボタン**

表 5-26 [オプション] ダイアログの機能ボタン

ボタン	機能
OK	指定した内容で LDG の動作設定を行い、このダイアログをクローズします。
適用(A)	今回の版では選択できません。
キャンセル	指定した内容を無効にします。
ヘルプ(H)	このダイアログのオンライン・ヘルプを表示します。



## 第6章 メッセージ

### 6.1 表示形式

LDG が出力するメッセージは、[図 6-1](#) に示すメッセージ・ダイアログ内に表示されます。

なお、メッセージは種別ごとに4種類あり、メッセージを表示する際、先頭にその種別を示すアイコンと英字を付与します。各メッセージ種別の意味は、[表 6-1](#) に示すとおりです。

図 6-1 メッセージ・ダイアログの例



表 6-1 メッセージ種別一覧

アイコン	英字	種別
	E	エラー・メッセージ
	W	警告メッセージ
	Q	質問メッセージ
	I	情報メッセージ

## 6.2 エラー・メッセージ

エラー・メッセージとして表示される番号と内容は、次のとおりです。

なお、" "内の文字列は、" "内で記された内容の文字列を表示します。

表 6-2 エラー・メッセージ一覧

番号	メッセージ
E0001	メッセージ・ファイル"ファイル名"が見つかりません。再インストールしてください。
E0002	"ファイル名"を正しく読み込むことができませんでした。[環境選択ダイアログ]でリンク・ディレクティブの環境を選択してください。
E0900	デバイス・ファイルがインストールされていませでした。デバイス・ファイル・インストーラを使用して、使用するデバイスのデバイス・ファイルをインストールしてください。
E0901	"デバイス名"用のデバイス・ファイル読み込みに失敗したためアドレス空間の取得ができませんでした。デバイス・ファイル・インストーラを使用して再インストールしてください。

## 6.3 警告メッセージ

警告メッセージとして表示される番号と内容は、次のとおりです。

なお、" "内の文字列は、" "内で記された内容の文字列を表示します。

表 6-3 警告メッセージ一覧

番号	メッセージ
W0001	"ファイル名"は読み取り専用です。コピーを保存するには、[OK]をクリックし、[名前を付けて保存ダイアログ]で、別の名前を付けて保存してください。
W0002	"ドライブ"への書き込み中にディスクがいっぱいになりました。ディスクの空き容量を増やすか、他のディスクに保存してください。
W0003	実行ファイルが複数選択されています。"ファイル名["ファイル名"...]
W0004	"ファイル名"は変更されています。保存しますか？
W0005	"ファイル名"のフォーマットが正しくありませんでした。
W0006	"文字列"を数値に変換できませんでした。
W0007	内蔵メモリを編集することはできません。
W0008	空き領域を編集することはできません。
W0009	"メモリもしくはセクション名"を編集することはできません。
W000A	"メモリ/セクション名"の"プロパティ名"を編集することはできません。
W000B	領域が"メモリ/セクション名"と重なります。
W000C	メモリのない番地です。
W000D	名前が長すぎます。名前は"数字"文字以内で指定してください。
W000F	"メモリ名"の開始番地がアドレス空間の範囲外です。

番号	メッセージ
W0010	"メモリ名"の終了番地がアドレス空間の範囲外です。
W0011	"メモリ名"の開始番地が"メモリ名"の範囲外です。
W0012	"メモリ名"の終了番地が"メモリ名"の範囲外です。
W0013	"メモリ名"の開始番地と終了番地のミラー番号が異なります。
W0014	"メモリ名もしくはセクション名"のサイズが負になります。
W0015	"メモリ名"にドロップするにはサイズが大きすぎます。
W0017	ROMにデータセクションを配置できません。
W0018	前方のセクション"セクション名"の終了番地が確定していないため、このセクションの開始番地が確定できません。開始番地を指定するようにしてください。
W0019	名前を自動で割り当てることができませんでした。"メモリもしくはセクション名" ~ "メモリもしくはセクション名 4294967295"を適切な名称に変更してください。
W001A	"文字列"は見つかりませんでした。
W001B	"ファイル名"にアクセスできませんでした。ファイルが存在するか確認してください。ファイルやドライブのアクセス権を確認してください。
W0900	サイズより小さい値を指定しました。サイズより大きい値を指定してください。最大サイズに現在のサイズより小さい値を設定しようとした場合に表示します。
W0901	"リンク・ディレクティブ・ファイル名" "行番号":ディレクティブの終わりには";"が必要とされます。
W0902	"リンク・ディレクティブ・ファイル名" "行番号":リージョンの終わりには"}"が必要とされます。
W0903	"リンク・ディレクティブ・ファイル名" "行番号":ディレクティブは名前ではじめてください。
W0904	"リンク・ディレクティブ・ファイル名" "行番号":セクション・ディレクティブはセクション名ではじめてください。
W0905	"リンク・ディレクティブ・ファイル名" "行番号":ディレクティブの始まりの名前の後ろには":", "=", または"@"が必要とされます。
W0906	"リンク・ディレクティブ・ファイル名" "行番号":出力セクション名の後ろには"="が必要とされます。
W0907	"リンク・ディレクティブ・ファイル名" "行番号": "{" に対応する "}" の数が多すぎます。
W0908	"リンク・ディレクティブ・ファイル名" "行番号":リンク・ディレクティブに不正な文字("文字コード")が存在しています。
W090A	"リンク・ディレクティブ・ファイル名" "行番号":セグメント・ディレクティブにおいて"文字列"を指定することはできません。
W090B	"リンク・ディレクティブ・ファイル名" "行番号":セクション・ディレクティブにおいて"文字列"を指定することはできません。
W090C	"リンク・ディレクティブ・ファイル名" "行番号":シンボル・ディレクティブにおいて"文字列"を指定することはできません。
W090D	"リンク・ディレクティブ・ファイル名" "行番号":ファイル名を指定する部分に"文字列"を指定することはできません。

番号	メッセージ
W090E	"リンク・ディレクティブ・ファイル名" "行番号":セグメント名を指定する部分に"文字列"を指定することはできません。
W090F	"リンク・ディレクティブ・ファイル名" "行番号": "文字列" が、セグメント"セグメント名"に対し、同じセグメント・ディレクティブまたは別のセグメント・ディレクティブにおいて複数回指定されています。
W0910	"リンク・ディレクティブ・ファイル名" "行番号": "文字列" が、セクション"セクション名"に対し、同じセクション・ディレクティブまたは別のセクション・ディレクティブにおいて複数回指定されています。
W0911	"リンク・ディレクティブ・ファイル名" "行番号": "文字列" が、シンボル"シンボル名"に対し、同じシンボル・ディレクティブまたは別のシンボル・ディレクティブにおいて複数回指定されています。
W0912	"リンク・ディレクティブ・ファイル名" "行番号":セグメント"セグメント名"はすでに定義されています。
W0913	"リンク・ディレクティブ・ファイル名" "行番号":セクション"セクション名"は"行番号"行目においてすでに定義されています。
W0914	"リンク・ディレクティブ・ファイル名" "行番号":シンボル"シンボル名"は"行番号"行目においてすでに定義されています。
W0915	"リンク・ディレクティブ・ファイル名" "行番号":セグメント・タイプとして指定することのできない"文字列"が指定されています。
W0916	"リンク・ディレクティブ・ファイル名" "行番号":セクション・タイプとして指定することのできない"文字列"が指定されています。
W0917	"リンク・ディレクティブ・ファイル名" "行番号":セクション属性として指定することのできない"文字"が指定されています。
W0918	"リンク・ディレクティブ・ファイル名" "行番号":セグメント・タイプにLOADを指定していないセグメント・ディレクティブにおいて"文字列"を指定することはできません。
W0919	"リンク・ディレクティブ・ファイル名" "行番号":奇数の値"指定した数値"を偶数の値"数値"に整列しました。
W091A	"リンク・ディレクティブ・ファイル名" "行番号":セグメント"セグメント名"のセグメント・ディレクティブには"文字列"が必要とされます。
W091B	"リンク・ディレクティブ・ファイル名" "行番号":セクション・ディレクティブ"セクション名"には"文字列"が必要とされます。
W091C	"リンク・ディレクティブ・ファイル名" "行番号":シンボル"シンボル名"のシンボル・ディレクティブには"文字列"が必要とされます。
W091D	"リンク・ディレクティブ・ファイル名" "行番号":シンボル種別として指定することのできない"文字列"が指定されています。
W091E	"リンク・ディレクティブ・ファイル名" "行番号":シンボル種別"文字列"が、同じディレクティブまたは別のディレクティブにおいて複数回指定されています。

番号	メッセージ
W091F	"リンク・ディレクティブ・ファイル名" "行番号":セクション"セクション名"のセクション属性"属性を表す文字"とこのセクションの割り付けが指示されているセグメント"セグメント名"のセグメント属性が一致しません。 セクション属性 G は無視し、セクション属性 A, W および X がそれぞれセグメント属性 R, W および X に相当するものとして一致するようにしてください。
W0920	"リンク・ディレクティブ・ファイル名" "行番号":セクション"セクション名"の先頭アドレス"番地"が属するセグメント"セグメント名"の先頭アドレス"番地"よりも手前に割付られています。
W0921	"リンク・ディレクティブ・ファイル名" "行番号": "文字列"には有効なパラメータが必要です。
W0922	"リンク・ディレクティブ・ファイル名" "行番号": ep シンボル・ディレクティブにおいて"文字列"を指定することはできません。無視しました。
W0923	"リンク・ディレクティブ・ファイル名" "行番号": "文字列"がセクション"セクション名"に対し、同じセクション・ディレクティブまたは別のセクション・ディレクティブにおいて複数回指定されています。
W0924	"リンク・ディレクティブ・ファイル名" "行番号":セクション・タイプとして指定することのできない"文字列"が指定されました。

## 6.4 質問メッセージ

質問メッセージとして表示される番号と内容は、次のとおりです。

表 6-4 質問メッセージ一覧

番号	メッセージ
Q0001	全メモリ空間表示を行った後、全メモリ空間表示を解除できません。全メモリ空間表示は、同じ物理アドレスに重複してセクションを配置する場合に使用してください。全メモリ空間表示にしますか？

## 6.5 情報メッセージ

情報メッセージとして表示される番号と内容は、次のとおりです。

なお、" " 内の文字列は、" " 内で記された内容の文字列を表示します。

表 6-5 情報メッセージ一覧

番号	メッセージ
I0001	メモリ "メモリ名" ("開始番地" ~ "終了番地") を追加しました。 【備考】既存のリンク・ディレクティブ・ファイルを読み込んだ際に、セクションの配置が内蔵メモリ上にない場合、自動的にメモリを追加し、このメッセージを表示します。
I0002	セクション・サイズ情報がないため、リンク時にセクションの領域が重複している可能性やセクション領域がメモリ領域内に収まらない可能性があります。 【備考】リンク・ディレクティブを保存する際に、オブジェクト・ファイルを読み込んでいない場合、このメッセージを表示します。オブジェクト・ファイルを読み込んでいない場合、セクションのサイズ情報がないため、セクションの領域のチェックが行えないためです。
I0003	"メモリ名" に指定された開始番地 "指定した開始番地" にアラインを適用して "アライン適用後の開始番地" にしました。
I0004	前セクションがありません。"セクション名" の開始番地を "アドレス" にします。
I0005	"メモリ/セクション/オブジェクト・ファイル名" は既に存在します。
I0007	オブジェクト・ファイル "オブジェクト・ファイル名" が見つからなかったため、情報の取得は行えませんでした。
I0900	セグメント "セグメント名" のセグメント属性に R を付加しました。
I0901	セクション "セクション名" のセクション属性に A を付加しました。
I0902	セクション "セクション名" に指定したアドレスをセグメント "セグメント名" に指定したアドレスとして読み込みました。

# 付録 A リンク・ディレクティブ

## A.1 概要

ここでは、リンク・ディレクティブに必要となる項目や、リンク・ディレクティブ・ファイルの記述方法について説明します。

組み込み系のアプリケーションでは、プログラム・コードをある番地から配置したり、分割して配置するなど、メモリ配置に気を配る必要があります。

期待どおりのメモリ配置を実現するには、プログラム・コードやデータの配置情報を、リンクに指示する必要があります。この指示の情報を「リンク・ディレクティブ」と呼び、リンク・ディレクティブを記述するファイルを「リンク・ディレクティブ・ファイル」と呼びます。

リンクは、このリンク・ディレクティブ・ファイルに従ってメモリ配置を決定し、ロード・モジュールを作成します。

### A.1.1 指定項目

リンク・ディレクティブで指定する主な項目は、大きく分けて次の2つになります。

#### セグメント・ディレクティブとマッピング・ディレクティブ

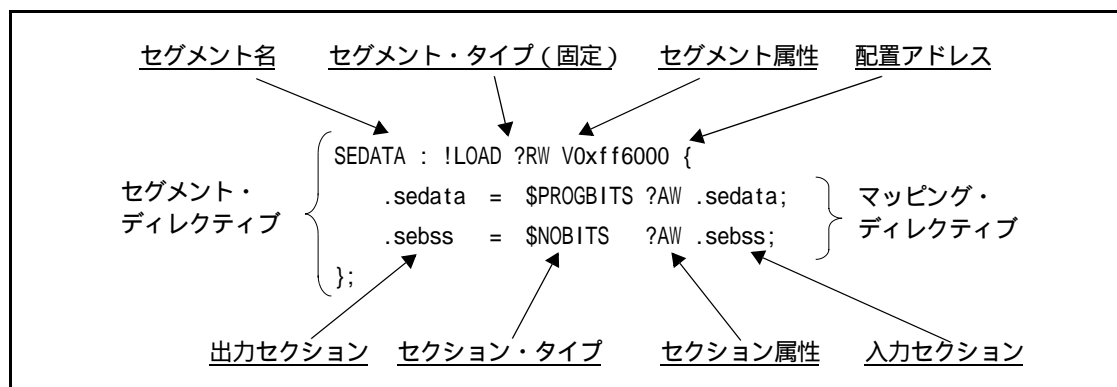
プログラムやデータが配置される領域である「セクション」の情報を、タイプと属性別に分けて「セグメント」の情報としてまとめ、その配置アドレスを決めます。

ここで、セクション情報の記述を「マッピング・ディレクティブ」、セグメント情報の記述を「セグメント・ディレクティブ」といいます。

次に、リンク・ディレクティブ・ファイルに記述する「セグメント・ディレクティブ」と「マッピング・ディレクティブ」の一例を示します。

なお、詳しい書式については、「[A.4 書式](#)」を参照してください。

図 A-1 セグメント・ディレクティブとマッピング・ディレクティブ



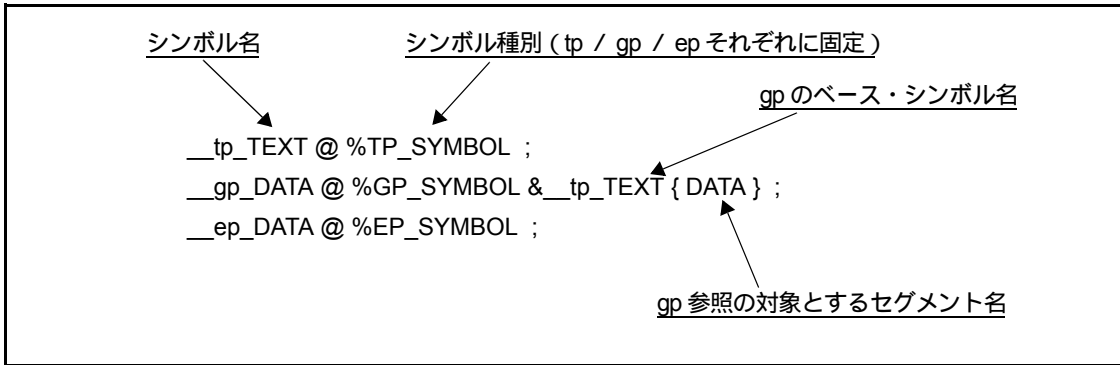
### シンボル・ディレクティブ

tp (テキスト・ポインタ), gp (グローバル・ポインタ), ep (エレメント・ポインタ) を生成するための「シンボル」を作成します。これらのシンボルの情報を「シンボル・ディレクティブ」と呼びます。

次に、リンク・ディレクティブ・ファイルに記述する「シンボル・ディレクティブ」の一例を示します。

なお、詳しい書式については「[A.4 書式](#)」を参照してください。

図 A-2 シンボル・ディレクティブ





## A.2 セクションとセグメント

ここでは、セクションとセグメントについて、それぞれ説明します。

### A.2.1 セクション

セクションとは、プログラムを構成する基本的な単位（プログラムやデータが配置される領域）です。たとえばプログラム・コードは text 属性セクションへ、初期値を持つ変数は data 属性セクションへ、というように、セクションごとに分けて配置することになります。

セクション名はアプリケーション内で指定できます。C 言語では “ # pragma section 指令 ” や “ # pragma text 指令 ”, アセンブリ言語では “ .section 疑似命令 ” によって指定できます。

ただし # pragma 指令でセクションの指定をしない場合でも、コンパイラはプログラム・コードやデータ（変数）にデフォルトとして決められたセクションを割り当てるようにしています。

これらの詳細については、「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル C 言語編, またはアセンブリ言語編」をそれぞれを参照してください。

### A.2.2 セグメント

セグメントとは、プログラムやデータをメモリにロードする際の単位です。属性が同じだったり、タイプが同じであるようなセクション群を 1 つに集め、1 セグメントとして扱います。つまり「似たようなセクションの集まり = セグメント」というイメージになります。

リンク・ディレクティブでは、セグメント名、属性、配置するアドレスなどを自由に設定できます。

**【注意】**セグメント名、属性として指定できない文字があります。詳細は、「[A.4.3 セグメント・ディレクティブ](#)」を参照してください。

読み出し可能 (R) で実行可能 (X) なセグメント「TEXT1」を 0x100000 番地へ配置する場合、リンク・ディレクティブ・ファイルでは次のように記述します。

```
TEXT1 : !LOAD ?RX V0x100000 {
      :
      (マッピング・ディレクティブ)
      :
};
```

セグメントはメモリにロードする際の単位であるため、プログラム・コードやデータを配置するときは、セグメント単位で行います。つまり、あるセクションを特定のメモリに配置したい場合、そのセクション情報をマッピング・ディレクティブに従って記述し、そのマッピング・ディレクティブを含むセグメントを作成します。そしてそのセグメントの配置アドレスを決定するという手順を踏みます。

**【注意】**マッピング・ディレクティブにおいて、セクションに直接アドレスを指定することもできますが、通常はセグメント単位でアドレスを指定します。

【例：変数 i を sdata 領域に配置し，関数 func1 を 0x120000 番地へ配置する場合】

**【test1.c】**

```
#pragma section sdata begin
i = 10 ;
#pragma section sdata end

#pragma text "f1.text" func1

void
func1( ){
    :
    return ;
}
```

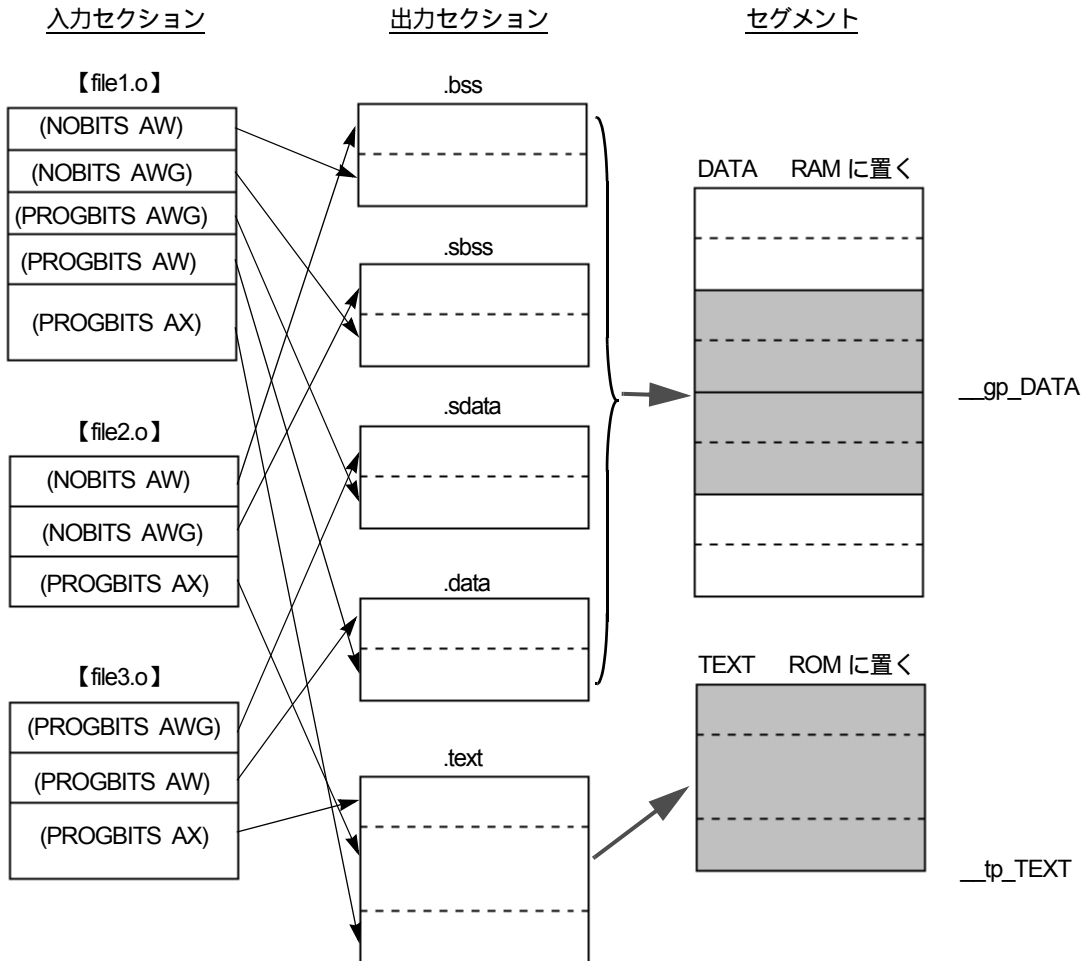
**【リンク・ディレクティブ(一部)】**

```
TEXT2 : !LOAD ?RX V0x120000{
    text1= $PROGBITS ?AX f1.text ;
};
DATA  : !LOAD ?R V0x200000{
    .data = $PROGBITS ?AW ;
    .sdata = $PROGBITS ?AWG ;
    .sbss = $NOBITS ?AWG ;
    .bss = $NOBITS ?AW ;
};
    :
    :
```

## A. 2.3 セグメントとセクションの関係

セグメントとセクションの関係を、マッピングのイメージで見ると次の例のようになります。

図 A-3 セグメントとセクションの関係



それぞれのオブジェクト (file1.o, file2.o, file3.o) に含まれているセクションを「入力セクション」と呼びます。これらは同種のセクションごとにまとめられます。まとめられて出力されたセクションを「出力セクション」と呼びます。出力セクション群は、リンク・ディレクティブにおいて、それぞれセグメント (DATA セグメント, TEXT セグメント) にまとめられているので、これに従って適切な場所へ (アドレスが指定されていれば、そのアドレスへ) マッピングされます。

テキスト・ポインタ (tp) のシンボル “\_\_tp\_TEXT” とグローバル・ポインタ (gp) のシンボル “\_\_gp\_DATA” も規則に従ってセットされます。

## A. 2.4 セクションの種類

ここでは、CA850 で扱うことのできるセクションとその特長について説明します。

割り当てを指定できるセクション種別とその特徴は、表 A-1 のとおりです。

なお、この形式やセクション・ファイルによりセクションへの割り当てを指定しないデータは、CA850 のオプションで指定されたサイズに従い、CA850 によって、.sdata セクション, .data セクション, .sbss セクション,

.bss セクションのいずれかに配置されます (注 1)。

型修飾子 const が指定されたデータ, および文字列定数は CA850 のオプションで指定されたサイズに従い, CA850 によって, .const セクション, .sconst セクションに配置されま す (注 2)。

また, セクションへの割り当ては, セクション・ファイルによっても指定できます (注 3)。

【注 1】デフォルトでは全データを .sdata セクション / .sbss セクションへ配置します。

「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル 操作編」の ca850 の -G オプションに関する説明を参照してください。

【注 2】「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル 操作編」の ca850 の -Xsconst オプションに関する説明を参照してください。

【注 3】「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル 操作編」の セクション・ファイル・ジェネレータ (sf850) を参照してください。

表 A-1 CA850 の配置セクション種別

種別	特長	指定文字列
.tidata.byte セクション .tidata.word セクション .tibss.byte セクション .tibss.word セクション ( tiny internal data / tiny internal bss )	ep ( エレメント・ポインタ ) から 1 命令で参照可能なセクションで, ep からプラス方向 へアクセスするセクションです。 sidata 属性 / sibss 属性セクションと違うところは, 同じ 1 命令アクセスでも, 使用するアセンブル命令が違うことです。sidata 属性 / sibss 属性セクションは, 格納 / 参照に 4 バイト長命令の "st / ld 命令" を使用しますが, tidata 属性 / tibss 属性セクションは, 2 バイト長命令の "sst / sld 命令" を使用して アクセスします。つまり sidata 属性 / sibss 属性セクションよりもコード効率がよくなります。ただし, sst / sld 命令が適用できる範囲は小さいので, 多くの変数を配置 することはできません。 初期値ありデータは, tidata ( tidata.byte, tidata.word ) 属性セクションへ, 初期値 なしデータは, tibss ( tibss.byte, tibss.word ) 属性セクションへ配置されます。 バイト・データを配置する場合は tidata.byte / tibss.byte 属性を, ワード・データを 配置する場合は tidata.word / tibss.word 属性を指定しますが, CA850 に自動判別 させたい場合は, tidata / tibss 属性を指定します。	tidata tidata_byte tidata_word
.data セクション .bss セクション ( data / bss )	gp ( グローバル・ポインタ ) から 2 命令で参照可能なセクションです。アドレス生成を行ってからアクセス (ld/st 命令) するため, その分コードが多くなり, 実行速度も落ちますが, 32 ビット 空間内すべてにアクセスが可能です。つまり RAM 上であれば, どこにでも配置が可能なセクションです。 初期値ありデータは data 属性セクションへ, 初期値なしデータは bss 属性セクションへ配置されます。	data

種別	特長	指定文字列
<p>.sdata セクション .sbss セクション ( sdata / sbss )</p>	<p>gp ( グローバル・ポインタ ) から 1 命令 ( ld/st 命令 ) で参照可能なセクションで, gp から ± 32K バイト内に配置される必要があります ( あわせて 64 K バイト )。</p> <p>初期値ありデータは sdata 属性 セクションへ, 初期値なしデータは sbss 属性セクションへ配置されます。</p> <p>CA850 では, まずこのセクションに配置するコードを生成しようとします。ただし, この属性のセクションに収まりきらないような場合は, data 属性 / bss 属性セクションへ配置するコードを生成します。</p> <p>なお, sdata 属性 / sbss 属性セクションへの配置データを少しでも多くする策として, CA850 のオプション "-G" で, 配置されるデータのサイズの上限を指定し, それ以上のサイズは sdata 属性 / sbss 属性セクションに配置しないという指定ができます。</p>	<p>sdata</p>
<p>.sedata セクション .sebss セクション ( small extended data / small extended bss )</p>	<p>ep ( エlement・ポインタ ) から 1 命令 ( ld/st 命令 ) で参照可能なセクションで, ep からマイナス方向へアクセスするセクションです。つまり ep からマイナス方向 32K バイト内に配置されるセクションです。</p> <p>初期値ありデータは sedata 属性 セクションへ, 初期値なしデータは sebss 属性セクションへ配置されます。</p> <p>gp から 1 命令でアクセスできる sdata 属性 / sbss 属性セクションに入りきらなくなったが, 1 命令アクセスしたい変数がまだ存在する場合, ep を使って 1 命令でアクセスできる範囲に置くことができます。</p> <p>sidata 属性 / sibss 属性セクションは ep からプラス方向にアクセスするためのセクションですが, sedata 属性 / sebss 属性セクションは ep からマイナス方向にアクセスするためのセクションです。</p>	<p>sedata</p>
<p>.sidata セクション .sibss セクション ( small internal data / small internal bss )</p>	<p>ep ( エlement・ポインタ ) から 1 命令 ( ld/st 命令 ) で参照可能なセクションで, ep からプラス方向へアクセスするセクションです。つまり ep からプラス方向 32K バイト内に配置されるセクションです。</p> <p>初期値ありデータは sidata 属性セクションへ, 初期値なしデータは sibss 属性セクションへ配置されます。</p> <p>gp から 1 命令でアクセスできる sdata 属性 / sbss 属性セクションに入りきらなくなったが, 1 命令アクセスしたい変数がまだ存在する場合, ep を使って 1 命令でアクセスできる範囲に置くことができます。</p> <p>sidata 属性 / sibss 属性セクションは ep からプラス方向にアクセスするためのセクションですが, sedata 属性 / sebss 属性セクションは ep からマイナス方向にアクセスするためのセクションです。</p>	<p>sidata</p>

種別	特長	指定文字列
.sconst セクション (small const data)	<p>r0, つまり 0 番地から 1 命令 (ld/st 命令) で参照可能なセクションで, 0 番地から ± 32K バイト内に配置される必要があります。基本的に "ROM に固定してもよいデータ" を配置するセクションです。</p> <p>V850 で内蔵 ROM を持つ製品の場合, 0 番地からプラス方向が内蔵 ROM である場合が多く, そこに 1 命令で参照したい, かつ ROM 固定してもよいデータを, sconst 属性セクションとして配置します。内蔵 ROM を持たないデバイス, および ROM レスモードを指定した場合には, 外部メモリに位置します。</p> <p>sconst 属性 / const 属性セクションに配置するデータは, const 修飾子をつけて宣言された変数 / データが対象となります。この属性のセクションに収まりきらないような場合は, const 属性セクションへ配置することになります。</p> <p>なお, sconst 属性セクションへの配置データを少しでも多くする策として, CA850 のオプション "-Xsconst" で, 配置されるデータのサイズの上限を指定し, それ以上のサイズは sconst 属性セクションに配置しないという指定ができます (オプションの詳細は「CA850C コンパイラ・パッケージ ユーザーズ・マニュアル 操作編」を参照してください)。</p>	sconst
.const セクション (const data)	<p>r0, つまり 0 番地から 2 命令で参照可能なセクションです。アドレス生成を行ってからアクセス (ld/st 命令) するため, その分コードが多くなり, 実行速度も落ちますが, 32 ビット空間内すべてにアクセスが可能です。sconst 属性セクションに入りきらなかった "ROM 固定してもよいデータ" や, V850 の ROM レス品で, 外部 ROM にデータを配置したい場合に, const 属性セクションに配置します。</p>	const

【注 1】“(2 命令)” は, アセンブラにより 2 命令に命令展開されるものです。

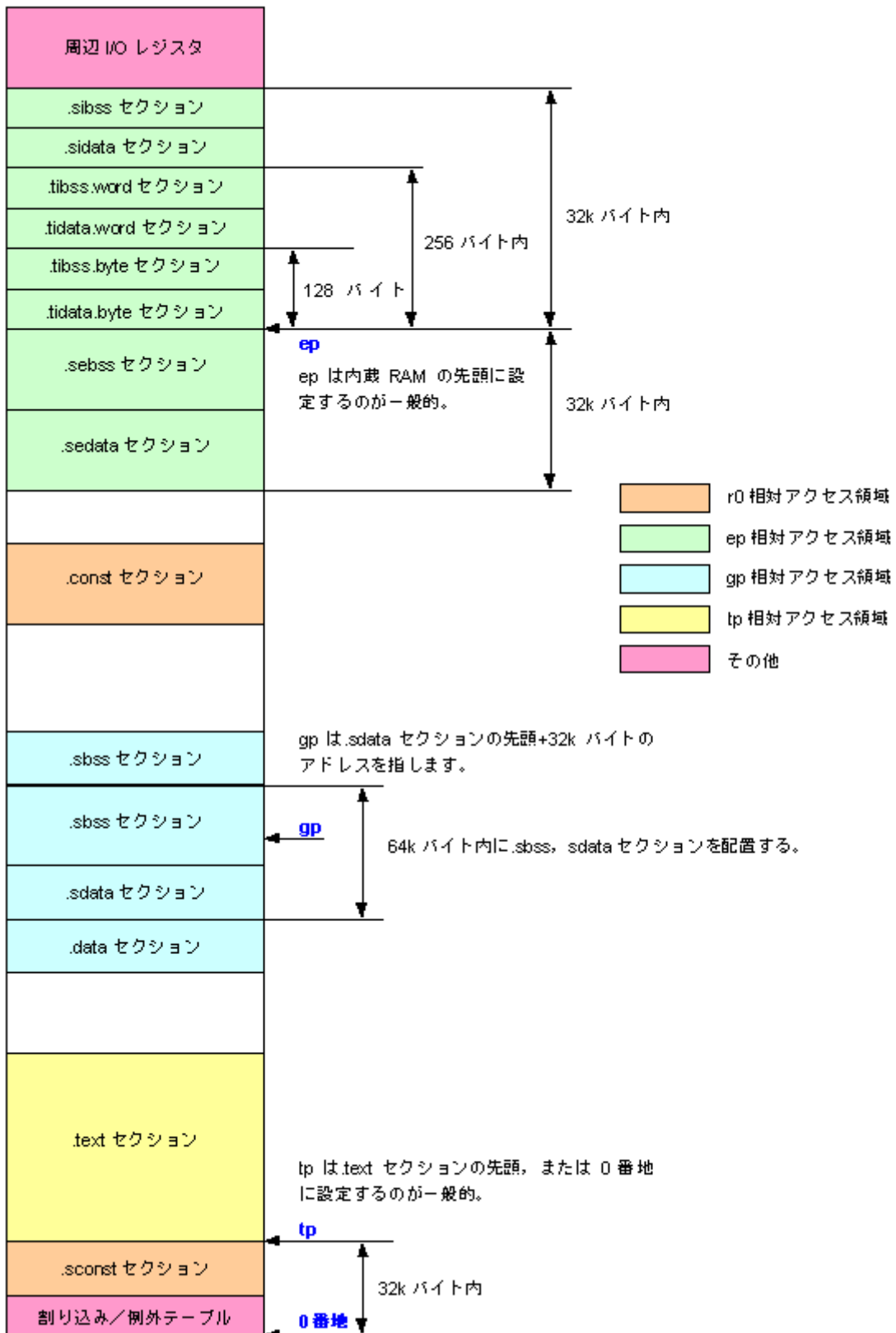
【注 2】“gp 相対”, “r0” 相対というのは, コンパイラが gp 相対や r0 相対のコードを出力することを示します。

【注 3】“外部メモリ” とあるセクション種別は, ターゲット・システムに外部メモリが実装されている場合に適用できます。

【備考】.tidata.byte, .tidata.word などの扱いについては, 「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル アセンブリ言語編」を参照してください。

次に, V852 における各セクションのメモリ配置イメージの例を示します。

図 A-4 CA850 による各セクションのメモリ配置イメージの例 (内蔵 ROM あり)



## A. 2.5 セクションのタイプと属性

セクションのタイプと属性について説明します。

これらは、マッピング・ディレクティブでセクションの情報を記述するときに必要となります。

セクションのタイプは、次のように分類できます。

表 A-2 セクションのタイプ

セクション・タイプ名	意味
PROGBITS	オブジェクト・ファイル内に実際の値を持っているセクション テキスト, 初期値ありデータ (変数)
NOBITS	オブジェクト・ファイル内に実際の値を持っていないセクション 初期値なしデータ (変数)

セクションの属性は、次のように分類できます。

表 A-3 セクションの属性

セクション属性	意味
A	メモリを占有するセクション (すべてのセクションが該当)
W	書き込み可能なセクション (RAM 上に配置するセクション)
X	実行可能なセクション (主にテキスト・セクション)
G	グローバル・ポインタ (gp) と 16 ビットのディスプレースメントを用いて参照すること のできるメモリ範囲内に割り付けるセクション (.sdata, .sbss セクション)



セクションを、タイプと属性別に分けると、次の 6 種類に分類されます。

表 A-4 セクションの分類

セクション属性	セクション・タイプ セクション属性		該当する予約セクション
bss 属性	セクション・タイプ	NOBITS	.bss .sebss .sibss .tibss.byte .tibss.word
	セクション属性	AW	
const 属性	セクション・タイプ	PROGBITS	.const .sconst
	セクション属性	A	
data 属性	セクション・タイプ	PROGBITS	.data .sedata .sidata .tidata.byte .tidata.word
	セクション属性	AW	
sbss 属性	セクション・タイプ	NOBITS	.sbss
	セクション属性	AWG	
sdata 属性	セクション・タイプ	PROGBITS	.sdata
	セクション属性	AWG	
text 属性	セクション・タイプ	PROGBITS	.pro_epi_runtime .text
	セクション属性	AX	

**【注意】** アプリケーション内でセクション名を独自に作成する場合、表 A-4 のようにそのセクションがどの属性にあたるのかを明確にし、ユーザ自身がマッピング・ディレクティブにおいて、セクション・タイプ、セクション属性とともに指定する必要があります。

なお、セクション名として、リンク・ディレクティブの書式により " V/H/A/+ 数字 " ではじまるセクション名は作成できません。

## A.3 シンボル

CA850 では、アプリケーションを実行する際に次のポインタを必要とします。

- テキスト・ポインタ (tp)
- グローバル・ポインタ (gp)
- エlement・ポインタ (ep)

各ポインタの値は「セグメントの位置」に関係するため、リンク・ディレクティブでポインタの値を決定する仕組みが必要となります。

リンク・ディレクティブでは、ポインタの値を決定するために「シンボル」を定義します。定義したシンボルの値はリンクによって決定され、その値をアプリケーション内でポインタにコピーすることにより、ポインタの値を決定できます。このように、リンク・ディレクティブで各ポインタ用のシンボルを定義することから、「シンボル・ディレクティブ」と呼ばれます。

ここでは、各ポインタの役割と、ポインタ値の決め方について説明します。

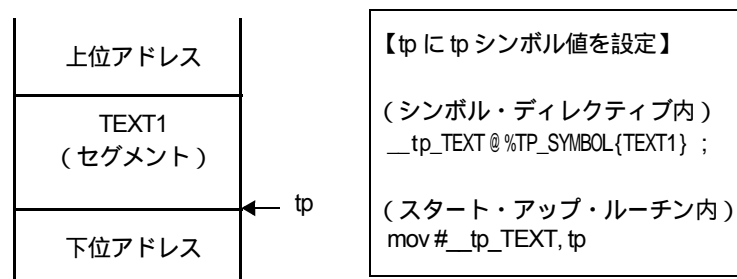
### A.3.1 テキスト・ポインタ (tp)

アプリケーションのテキスト領域を参照するとき、配置される位置に依存することのない参照 (Position Independent Code : PIC) を実現するために用意されるポインタが「テキスト・ポインタ (tp)」です。つまりテキストは tp 相対で参照されます。コンパイラは tp が正しく設定 (テキストの先頭) されていることを前提としたコードを出力するので、ポインタの値を正しく設定する必要があります。

tp はアプリケーションに 1 つだけではなく、セグメント単位で複数作成することもできます。

ただし、複数生成した場合、tp の切り替えはアプリケーション・プログラムで明示的に行う必要があります。

図 A-5 tp の設定例



この例では、リンク・ディレクティブで tp シンボル値が TEXT1 セグメントの先頭を指すように設定します。tp シンボル名は“\_\_tp\_TEXT”なので、リンク時に決定される TEXT1 セグメントの先頭アドレスが、シンボル“\_\_tp\_TEXT”にセットされます。

この値を tp に設定するために、スタート・アップ・ルーチンなどで、変数“tp”にこの“\_\_tp\_TEXT”の値を代入する式 (mov #\_\_tp\_TEXT, tp) を記述します。これにより tp に正しくテキスト・ポインタの値が設定されます。

### A. 3.2 グローバル・ポインタ (gp)

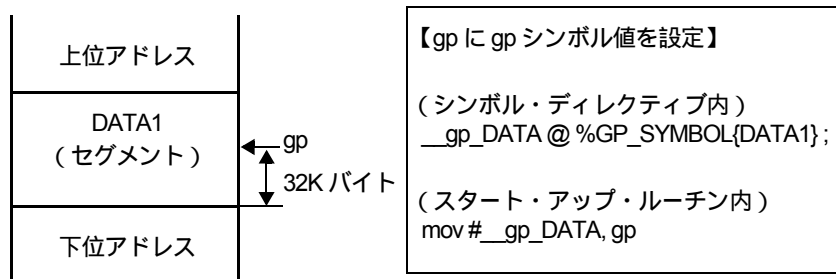
アプリケーション内でグローバル宣言したデータはメモリに配置されます。このメモリに配置されているデータを参照（ロード、ストア）する際、配置位置に依存することのない参照（Position Independent Data : PID）を実現するために用意されるポインタが「グローバル・ポインタ (gp)」です。

グローバル宣言したデータは gp 相対で参照されます。V850 コアでは「gp と 1 命令」か「gp と 2 命令」のどちらかでこれらのデータを参照できます。つまり「gp と 1 命令」でアクセスできる方が、アプリケーションの速度の向上、コード・サイズの縮小が見込めます。

gp と 1 命令 (ld / st 命令) で参照できるセクションは「sdata 属性, sbss 属性を持つセクション」、gp と 2 命令 (movhi + ld / st 命令) で参照できるセクションは「data 属性, bss 属性を持つセクション」です。つまり gp 相対で参照できるセクション (の属性) はこの 4 つとなります。このうち「sdata 属性, sbss 属性をもつセクション」は gp から正方向, 負方向にそれぞれ 32 K バイト内に配置されているため, この範囲にデータ (変数) を置くと 1 命令でアクセス可能で, 高速参照, コード・サイズ縮小につながります。

gp は 1 つだけではなく, セグメント単位で複数作成することもできます。ただし複数生成した場合, gp の切り替えはアプリケーション・プログラムで明示的に行う必要があります。

図 A-6 gp の設定例 (セグメントを指定する場合)



この例では, リンク・ディレクティブで gp シンボル値が DATA1 セグメント内を参照できるように設定します。gp シンボル名が “\_\_gp\_DATA” なので, リンク時に決定される DATA1 セグメントの先頭から 32 K バイト足したアドレスが, シンボル “\_\_gp\_DATA” にセットされます (図 A-4 参照)。

この値を gp に設定するために, スタート・アップ・ルーチンなどで, 変数 “gp” にこの “\_\_gp\_DATA” の値を代入する式 (mov #\_\_gp\_DATA, gp) を記述します。これにより gp に正しくグローバル・ポインタの値が設定されます。

なお, gp シンボルはアドレスだけではなく, tp シンボルからのオフセット値を指定することもできます。

次に, gp シンボルのオフセット指定について説明します。

**【gp シンボル値のオフセット指定】**

gp シンボル値の指定方法は、上記で説明したように「gp のアクセス対象とするセグメントを指定する方法」が一般的です。

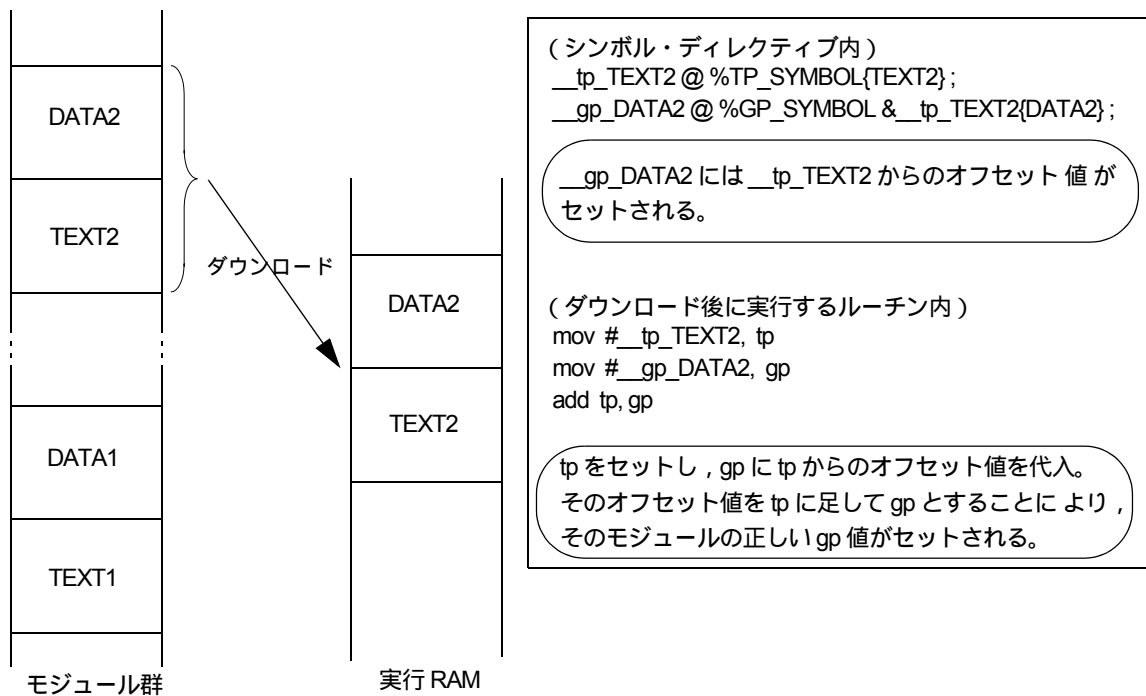
このほかに、「gp シンボルのアドレスを直接指定する方法」と「ベース・シンボルを決めて、そのシンボルからのオフセットを gp シンボル値とする方法」があります。ここでは後者について説明します（前者については、[【gp シンボル値の決定規則】](#)を参照してください）。

gp シンボルのベース・シンボルとして指定するものは「tp シンボル」です。

gp シンボルを作成する際に、tp シンボルをベース・シンボルとして指定すると、リンク・ディレクティブで gp のシンボル値として決定される値は「tp シンボル値からのオフセット値」となります。

こうすることにより tp シンボル値から gp シンボル値を「tp シンボル値 + tp シンボル値からのオフセット値」という計算によって容易に算出することができ、配置に依存しないアプリケーションの作成に有効となります。たとえば、複数の実行モジュールを持つアプリケーションで、その中の 1 つを RAM にコピーしてそこで実行したい場合に役立ちます。つまり、tp / gp の値を決定する際に、tp の値がわかれば、そのアドレスに gp のシンボル値（tp からのオフセット値）を足し、それを gp の値とすればよいことになります。

図 A-7 gp の設定例（tp からのオフセット指定の場合）



**【gp シンボル値の決定規則】**

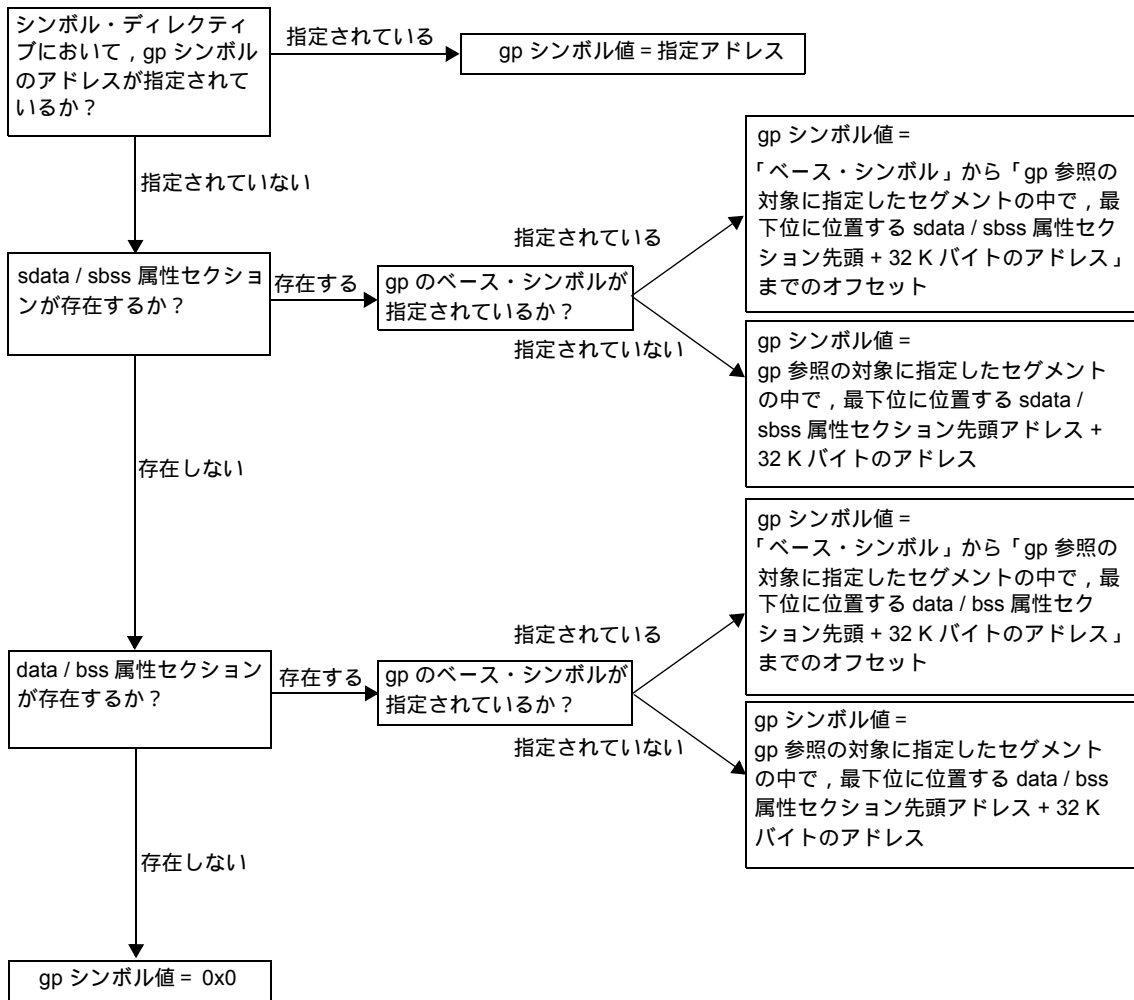
gp シンボル値の決定には、次のことが関係します。

- シンボル・ディレクティブにおけるアドレスの指定の有無
- sdata 属性 / sbss 属性 / data 属性 / bss 属性のセクションの有無
- ベース・シンボル指定の有無

リンカは、リンク・ディレクティブ・ファイルからこれらのことを調べ、gp シンボル値を決定します。

gp シンボル値の決定規則を図にまとめると、次のようになります。

図 A-8 グローバル・ポインタ値の決定規則



### A. 3.3 エlement・ポインタ (ep)

アプリケーション内でグローバル宣言したデータ (変数) を、V850 コアに内蔵されている RAM 領域へ配置し、より高速な参照 (ロード、ストア) を実現するために用意されているポインタが「Element・ポインタ (ep)」です。

グローバル宣言し、かつ、内蔵 RAM に配置するデータ (変数) は ep 相対で参照されます。

「ep と 1 命令」で参照することになりますが、その 1 命令が「sld / sst 命令」か「ld / st 命令」かによってセクションの属性が分かれます。

#### 「ep と sld / sst 命令」で参照できるセクション

tidata.byte 属性、tibss.byte 属性、tidata.word 属性、tibss.word 属性

#### 「ep と ld / st 命令」で参照できるセクション

sidata 属性、sibss 属性、sedata 属性、sebss 属性

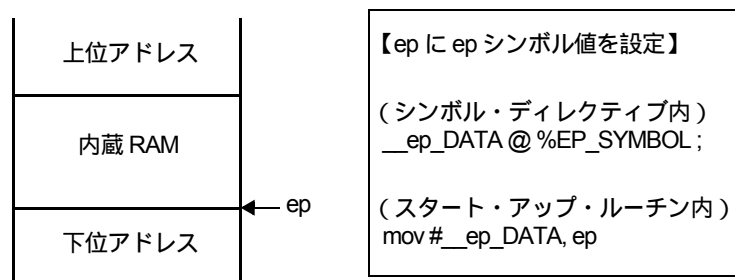
ただし、sedata 属性、sebss 属性は、内蔵 RAM ではなく、ep 相対でアクセス可能な外部 RAM のセクション属性になります。

一般的に内蔵 RAM の容量は少なく、多くのデータ (変数) を配置できませんが、特に参照を高速にしたいデータ (変数) を上記に示した領域に配置して「ep と 1 命令」でアクセスできるようにすると、アプリケーションの速度の向上やコード・サイズの縮小が見込めます。特に sld / sst 命令は、命令長が ld / st 命令の 4 バイトに対し、2 バイトとコード・サイズ縮小に役立ちます。

リンク・ディレクティブ・ファイルのシンボル・ディレクティブで ep シンボルの生成を指示した場合、リンカは使用するデバイスごとに用意されているデバイス・ファイルの情報を基に、自動的に内蔵 RAM 領域の先頭に設定されます。

なお ep はアプリケーション内で 1 つだけ生成できます。複数生成はできません。

図 A-9 ep の設定例



この例では、リンク・ディレクティブで ep シンボルを作成するように宣言します。ep シンボル名が “\_\_ep\_DATA” なので、リンカは内蔵 RAM の先頭アドレスを “\_\_ep\_DATA” にセットします。

この値を ep に設定するために、スタート・アップ・ルーチンなどで、変数 “ep” にこの “\_\_ep\_DATA” の値を代入する式 (mov #\_\_ep\_DATA, ep) を記述します。これにより ep に正しく Element・ポインタの値が設定されます。

【備考】アプリケーションが外部 RAM は一切使用せず、内蔵 RAM だけを使用するような場合は、gp シンボルを生成せず、ep シンボルだけを生成しても問題ありません。ただし、ランタイム・ライブラリを使用している場合、ランタイム関数が gp 相対でデータ (変数) 参照をしているので、gp シンボルを生成する必要があります。

【ep シンボル値の決定規則】

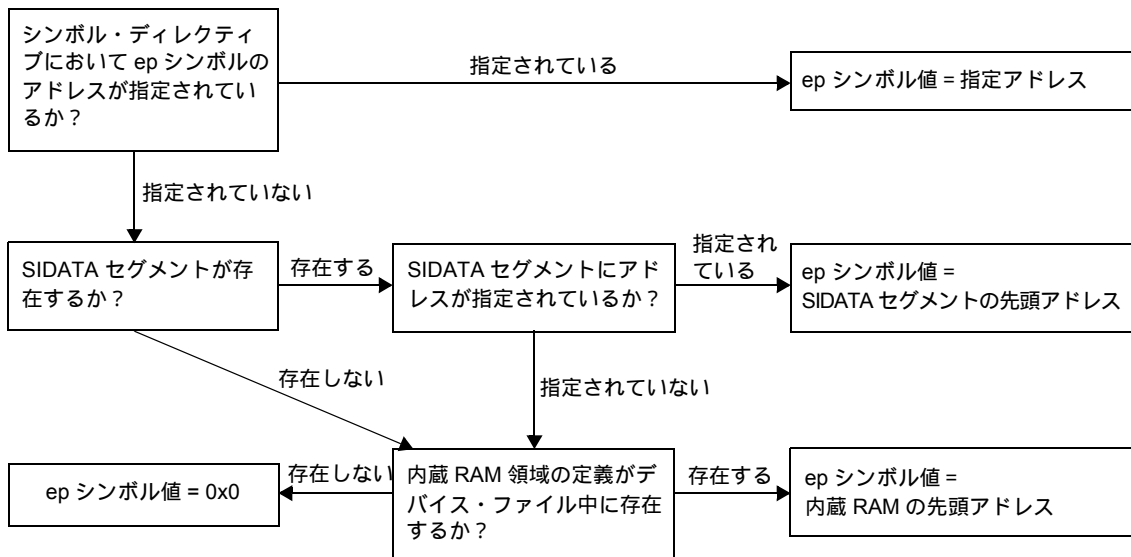
ep シンボル値の決定には次のことが関係します。

- シンボル・ディレクティブにおけるアドレス指定の有無
- SIDATA セグメントの有無
- デバイス・ファイル中の内蔵 RAM 領域の定義の有無

リンカは上記を調べ、ep シンボル値を決定します。

ep シンボル値の決定規則を図にまとめると、次のようになります。

図 A-10 エレメント・ポインタ値の決定規則



## A.4 書式

ここではリンク・ディレクティブ・ファイルの書式について、次の項目ごとに説明します。

- セグメント・ディレクティブ
- マッピング・ディレクティブ
- シンボル・ディレクティブ

リンク・ディレクティブの書式の概要は、次に示すとおりで、これらをエディタなどにより、テキスト形式で記述します。

```
セグメント・ディレクティブ 1{
    マッピング・ディレクティブ ;
};

セグメント・ディレクティブ 2{
    マッピング・ディレクティブ ;
};

セグメント・ディレクティブ 3{
    マッピング・ディレクティブ ;
};

セグメント・ディレクティブ 4{
    マッピング・ディレクティブ ;
};

tp シンボル・ディレクティブ ;
gp シンボル・ディレクティブ ;
ep シンボル・ディレクティブ ;
```

**【注意】** セグメント・ディレクティブは、アドレスの若い順に記述することを推奨します。

### A.4.1 使用できる文字

リンク・ディレクティブ・ファイルで使用できる文字は次のとおりです。

- 数字 (0 ~ 9)
- 英大文字 (A ~ Z)
- 英小文字 (a ~ z)
- アンダスコア (\_)
- ドット (.)
- スラッシュ (/)
- バック・スラッシュ, 円マーク (\, ¥)
- コロン (:) (ファイル名のみ使用できます)
- S-JIS 文字 (ファイル名のみ使用できます)
- 半角カナ文字 (ファイル名のみ使用できます)
- # (コメント用)



リンク・ディレクティブ・ファイル中の“#”は、コメントの始まりを示します。“#”が現れた位置からその行の行末まではコメントとして扱われます。

### A. 4.2 ファイル名

リンク・ディレクティブ・ファイルにつける「ファイル名」には、ファイル名として指定できる文字を使用していれば、特に制限はありません。ただし、拡張子が必要です。推奨は“dir”です。また、文字数があまり長くなると、リンク時に扱える全体の文字数（OS 依存）を越えてしまい、リンクできなくなってしまうことがあるので注意が必要です。

リンク時には、コマンド・ラインやメイク・ファイルを使用する際は、“-D” オプションを使用してリンク・ディレクティブ・ファイルを指定します。

PM+ を使用する際は、リンカ・オプションの設定ダイアログの [ 入力ファイル ] にある [ リンクディレクティブ [-D](D) ] の欄に、リンク・ディレクティブ・ファイル名を記述します。

### A. 4.3 セグメント・ディレクティブ

ここでは、次の項目でセグメント・ディレクティブの書式について説明します。

(1) 指定項目

(2) セグメント・ディレクティブの指定例

#### (1) 指定項目

セグメント・ディレクティブで指定する項目は、次のとおりです。

表 A-5 セグメント・ディレクティブで指定できる項目

項目	指定形式	意味	省略
セグメント名	セグメント名	作成するセグメントの名前	不可
セグメント・タイプ	!LOAD	メモリにロードされるタイプ（固定）	不可
セグメント属性	?[R][W][X]	作成するセグメントの属性が「読み出し可能（R）」、「書き込み可能（W）」、「実行可能（X）」であるかを指定（複数指定可能）	不可
アドレス	V アドレス	作成するセグメントの先頭アドレス	可
最大メモリ・サイズ	L 最大メモリ・サイズ	作成するセグメントが占有するメモリの上限	可
ホール・サイズ	H ホール・サイズ	セグメントの後ろに作成するホールのサイズ（次のセグメントとの間に入れる余白）	可
フィリング値	F フィリング値	ホール領域を埋めるのに用いる値	可
整列条件	A 整列条件	メモリ割り付けにおける整列条件（アラインメント）	可

セグメント・ディレクティブの具体的な書式は次とおりです。

```

セグメント名 : !セグメント・タイプ ?セグメント属性 Vアドレス
               L最大メモリ・サイズ Hホール・サイズ Fフィリング値
               A 整列条件 {
                   :
                   (マッピング・ディレクティブ)
                   :
               } ;
    
```

それぞれの項目は、空白で区切って記述します。また終わりには、必ず“ ; ”を付けます。なお、マッピング・ディレクティブについては、「A.4.4 マッピング・ディレクティブ」を参照してください。

「Vアドレス」「L最大メモリ・サイズ」「Hホール・サイズ」「Fフィリング値」「A 整列条件」は省略できます。省略した場合はデフォルトの値が使われます。デフォルトの値は次のとおりです。

表 A-6 省略可能項目のデフォルト値 (セグメント・ディレクティブ)

省略項目	デフォルト値
アドレス	先頭のセグメントである場合は 0x0 番地，以降のセグメントは，前のセグメントの終端に続いた値
最大メモリ・サイズ	0x100000 (バイト)
ホール・サイズ	0x0 (バイト)
フィリング値	0x0000
整列条件	0x8 (バイト)

【注意】セグメント・ディレクティブは、アドレスの若い順に記述することを推奨します。

(a) セグメント名

作成するセグメントの名前を指定します。

セグメント作成においては、このセグメント名の指定は省略できません。

セグメント名として指定できる文字は、「A.4.1 使用できる文字」で示したもので、任意の長さの文字列を指定できます。ただし、次の表 A-7 にある予約セクションを割り当てるセグメントは、セグメント名が固定されています。それ以外のセグメント名は使用できません。

表 A-7 セグメント名が固定されている予約セクション名

セクション名	セグメント名
.sidata .sibss .tidata .tibss .tidata.byte .tibss.byte .tidata.word .tibss.word	SIDATA
.sedata .sebss	SEDATA
.sconst	SCONST

【注意】.sconst は変更することはできませんが、一部エラー・チェックが行われません。

**(b) セグメント・タイプ**

作成するセグメントのタイプを指定します。

セグメント作成においては、このセグメント・タイプの指定を省略できません。

ここで指定できるタイプは、現在のところ「メモリにロードされるセグメント・タイプ」で「LOAD」のみです。これ以外の値を指定すると、リンカはエラーを出力します。なお「LOAD」は小文字でも可です。

セグメント・タイプの指定は“!”を先頭とし、“!”の直後に空白を置かないでください。

**(c) セグメント属性**

作成するセグメントの属性を指定します。

セグメント作成においては、このセグメント属性の指定は省略できません。

指定できるセグメント属性とその意味は次のとおりです。

なお、セグメント属性は、そのセグメントに属するマッピング・ディレクティブの属性に依存します。指定するときはマッピング・ディレクティブで指定するセクションの属性を考慮して設定する必要があります。

表 A-8 セグメント属性とその意味

セグメント属性	意味
R	読み出し可能なセグメント
W	書き込み可能なセグメント
X	実行可能なセグメント

セグメント属性は、複数個同時に指定することができ、R、W、Xの順番は任意で、空白を置かずに続けて記述します。また、セグメント属性の指定は、“?”を先頭とし、“?”の直後に空白を置かないでください。

【注意】1つのセグメント・ディレクティブにおいて、セグメント属性の指定が複数回行われた場合、リンカはエラーを出力しリンクを中止します。

[例]: SEG : !LOAD ?RX ?RW {};

**(d) アドレス**

作成するセグメントの開始アドレスを指定します。

セグメント作成においては、このアドレスの指定は省略できます。省略した場合、先頭のセグメントである場合は0x0番地、以降のセグメントは、前のセグメントの終端に続いた（アライメントを考慮した）値が開始アドレスとなります。

アドレス指定は、使用するマイコンのメモリ配置を考慮して指定する必要があります。

たとえば、V850 コアの場合、0x0番地はリセット割り込み処理（リセット割り込みハンドラ）となっています。リセット割り込み処理を行う場合、他のセグメントが0x0番地に割り付かないようにアドレスを設定する必要があります。

また、V850 コアも種類ごとに搭載しているメモリの大きさが違うので、内蔵 ROM / RAM の開始・終了アドレスも違ってきます。そのため使用するマイコンに応じて、各セグメントの配置アドレスを考慮する必要があります。各 CPU のメモリ情報などに関しては、その CPU のユーザーズ・マニュアル ハードウェア編や、デバイス・ファイルのユーザーズ・マニュアルを参照してください。

アドレスの値は偶数で指定してください。奇数を指定すると、リンカはメッセージを出力し、指定されたアド

レスに 1 を加えたアドレスが指定されたものとみなしてリンクを続行します。

アドレスの指定は “V” (大文字・小文字どちらでも可) を先頭とし、“V” の直後に空白を置かないでください。アドレスとして指定できる数値は 10 進数、16 進数のどちらかで、16 進数で指定する場合は数値の前に “0x” を記述してください。なお、アドレスの指定に式を用いることはできません。

#### (e) 最大メモリ・サイズ

作成するセグメントのメモリ・サイズの最大値を指定します。

これはセグメントが意図したサイズを越えないようにするための指定です。ですから、実際のサイズが最大メモリ・サイズとして指定した値よりも小さい場合は、次にくるセグメントは、そのセグメントの直後になります。

セグメント作成においては、この最大メモリ・サイズの指定は省略できます。省略した場合、0x100000 (バイト) がデフォルトの値として用いられます。

作成されたセグメントが最大メモリ・サイズで指定された値を越えた場合、リンクはエラーを出力し、リンクを中止します。

最大メモリ・サイズの指定は “L” (大文字・小文字どちらでも可) を先頭とし、“L” の直後に空白を置かないでください。なお、最大メモリ・サイズの指定に式を用いることはできません。

#### (f) ホール・サイズ

作成するセグメントのホールのサイズを指定します。

セグメントのホールとは、セグメントとセグメントの間の余白を意味します。ホール・サイズの指定を行った場合、そのセグメントの終わりにホールが作成されます。

セグメント作成においては、このホール・サイズの指定は省略できます。省略した場合、0x0 (バイト) がデフォルトの値として用いられます (ホールは作成しません)。

ホール・サイズの指定は “H” (大文字・小文字どちらでも可) を先頭とし、“H” の直後に空白を置かないでください。

なお、ホール・サイズの指定に式を用いることはできません。

#### (g) フィリング値

セグメントの割り付けにおいて生じるホール、および “H” 指定で明示的に作成したホールに対し、ホールの領域を埋めるのに用いる値 (フィリング値) を指定します。

フィリング値の指定を用いる場合は、“-B” オプションを指定し、2 パス・モードでリンクを行ってください。デフォルトの 1 パス・モードでフィリング値の指定が用いられた場合、リンクはメッセージを出力し、この指定を無視してリンクを続行します。

セグメント作成においては、このフィリング値の指定は省略できます。省略した場合、0x0000 がデフォルトの値として用いられます (0 で埋めます)。ただし、リンクのオプションで “-F” (フィリング値指定オプション) が指定された場合、リンクはメッセージを出力し、この指定を無視してリンクを続行します。

フィリング値の指定は “F” (大文字・小文字どちらでも可) を先頭とし、“F” の直後に空白を置かないでください。フィリング値は 2 バイト 4 桁の 16 進数で指定してください。4 桁に満たない場合は、満たない桁分の 0 が指定されたものとします。また、ホールのサイズが 2 バイトに満たない場合は必要な桁数分だけ、指定されたフィリング値の下位から取り出します。なお、フィリング値の指定に式を用いることはできません。

**(h) 整列条件**

作成するセグメントのメモリ割り付けにおいて、セグメントの整列条件（アラインメント値）を指定します。セグメント作成においては、この整列条件の指定は省略できます。省略した場合、0x8（バイト）がデフォルトの値として用いられます（8バイトでアラインされます）。

整列条件の指定は“A”（大文字・小文字どちらでも可）を先頭とし、“A”の直後に空白を置かないでください。整列条件の値は偶数で指定してください。奇数を指定すると、リンカはメッセージを出力し、指定された値に1を加えた値が指定されたものとみなしてリンクを続行します。なお、整列条件の指定に式を用いることはできません。

**(2) セグメント・ディレクティブの指定例**

次のようなセグメントを作成する場合を例とします。

表 A-9 セグメント例

項目	値
セグメント名	PROG1
セグメント属性	読み出し可能，実行可能
配置アドレス	0x1000 番地
最大メモリ・サイズ	0x200000（バイト）
ホール・サイズ	0x20（バイト）
フィリング値	0xffff
整列条件	0x16（バイト）

この場合、セグメント・ディレクティブの記述は次のようになります。

```
PROG1 : !LOAD ?RX V0x1000 L0x200000 H0x20 F0xffff A0x16{
      :
      (マッピング・ディレクティブ)
      :
      };
```

**【注意】** 基本的に、セグメント・ディレクティブは、配置するアドレス順に記述しなくても問題ありません。

ただし、.sedata セクション / .sebss セクションをもつセグメント（デフォルトでは“SEDATA セグメント”）に関しては、配置アドレスを省略したときにかぎり、この規則の例外となります。

CA850 では、SEDATA セグメントは内蔵 RAM 手前の領域を ep 相対の 1 命令参照するものとして定義しており、配置アドレスを省略すると、デバイス・ファイルに定義された内蔵 RAM の先頭アドレスから 0x8000 を引いたアドレスを指定したものと扱います。

たとえば、リンク・ディレクティブ・ファイル中に次の記述があったとします。

```

SIDATA : !LOAD ?RW V0xffb000 {
        .tidata.byte = $PROGBITS ?AW .tidata.byte ;
        .tibss.byte  = $NOBITS   ?AW .tibss.byte ;
        .tidata.word = $PROGBITS ?AW .tidata.word ;
        .tibss.word  = $NOBITS   ?AW .tibss.word ;
        .sidata      = $PROGBITS ?AW .sidata ;
        .sibss       = $NOBITS   ?AW .sibss ;
    } ;

SEDATA : !LOAD ?RW {
        .sedata      = $PROGBITS ?AW .sedata ;
        .sebss       = $NOBITS   ?AW .sebss ;
    } ;

DATA   : !LOAD ?RW {
        .data        = $PROGBITS ?AW .data ;
        .sdata       = $PROGBITS ?AWG .sdata ;
        .sbss        = $NOBITS   ?AWG .sbss ;
        .bss         = $NOBITS   ?AW .bss ;
    } ;

```

SEDATA のアドレスが省略されており、デバイス・ファイル情報から、この先頭アドレスは 0xff2000 (=0xffb000-0x8000) と判断します。SIDATA は 0xffb000 番地に配置定義されているため、CA850 は内部的に SEDATA を SIDATA の前に移動してリンクをします。

また、その後に定義されている DATA セグメントもアドレスが省略されているため、SEDATA セグメントの直後に配置されることとなります。

## A. 4. 4 マッピング・ディレクティブ

ここでは、次の項目でマッピング・ディレクティブの書式について説明します。

(1) 指定項目

(2) マッピング・ディレクティブの指定例

### (1) 指定項目

マッピング・ディレクティブで指定する項目は、次のとおりです。

表 A-10 マッピング・ディレクティブで指定できる項目

項目	指定形式	意味	省略
出力セクション名	出力セクション名	ロード・モジュールに出力されるセクションの名前	不可
セクション・タイプ	\$PROGBITS / \$NOBITS	作成するセクションのタイプ	不可
セクション属性	?[A][W][X][G]	作成するセクションの属性が「メモリを占有 (A)」、「書き込み可能 (W)」、「実行可能 (X)」、「gp と 16 ビット・ディスプレイメントで参照可 (G)」であるかを指定 (複数指定可能)	不可
アドレス	V アドレス	作成するセクションの先頭アドレス	可
ホール・サイズ	H ホール・サイズ	セクションの後ろに作成するホールのサイズ (次のセクションとの間に入れる余白)	可
整列条件	A 整列条件	メモリ割り付けにおける整列条件 (アラインメント)	可
入力セクション名	入力セクション名	出力セクションに割り付ける入力セクションの名前	可
オブジェクト・ファイル名	{ファイル名 ファイル名...}	入力セクションとして抽出したいセクションを含むオブジェクト・ファイル (複数指定可、スペースで区切る)	可

マッピング・ディレクティブの具体的な書式は次とおりです。

```
出力セクション名=$セクション・タイプ?セクション属性 Vアドレス
Hホール・サイズ A整列条件 入力セクション名
{オブジェクト・ファイル名 オブジェクト・ファイル名};
```

それぞれの項目は、空白で区切って記述します。また終わりには、必ず “ ; ” を付けます。

「V アドレス」「H ホール・サイズ」「A 整列条件」「入力セクション名」「オブジェクト・ファイル名」は省略できます。省略した場合はデフォルトの値が使われたり、決まった規則が用いられます。デフォルトの値、規則は次のとおりです。

表 A-11 マッピング・ディレクティブの指定項目で省略可能な値のデフォルト値 / 規則

省略項目	デフォルト値
アドレス	セグメント・ディレクティブで指定されたアドレスに従う 複数セクションがあり、先頭のセクションでなければ、前のセクションの終端に 続く値 先頭のセクションならば、セグメントの先頭に続く値
ホール・サイズ	0x0 (バイト)
整列条件	.tidata.byte / .tidata.word セクション : 0x1 (バイト) 上記セクション以外 : 0x4 (バイト)
入力セクション	作成する出力セクションと同じ属性を持つセクションを、すべてのオブジェクト から抽出 オブジェクト・ファイル名が指定されていれば、そのオブジェクトから抽出
オブジェクト・ファイル名	作成する出力セクションと同じ属性を持つセクションを、すべてのオブジェクト から抽出 入力セクションが指定されていれば、作成する出力セクションと同じ属性を持つ すべてのオブジェクトから抽出

次にそれぞれの指定項目について詳しく説明します。

**(a) 出力セクション名**

ロード・モジュールに出力されるセクションの名前を指定します。セクション作成においては、この出力セクション名の指定は省略できません。

出力セクション名として指定できる文字は、「A.4.1 使用できる文字」で示したもので、任意の長さの文字列を指定できます。

ただし、次の表 A-12 にあるセクションは、出力セクション名と入力セクション名の対応が固定されているので、別の名前のセクション名は指定できません。

表 A-12 出力セクション名が固定されている入力セクション

入力セクション名	出力セクション名
.tidata セクション	.tidata
.tibss セクション	.tibss
.tidata.byte セクション	.tidata.byte
.tibss.byte セクション	.tibss.byte
.tidata.word セクション	.tidata.word
.tibss.word セクション	.tibss.word
.sidata セクション	.sidata
.sibss セクション	.sibss
.sedata セクション	.sedata
.sebss セクション	.sebss
.pro_epi_runtime セクション	.pro_epi_runtime



**【注意】** .sconst は変更することはできませんが、一部エラー・チェックが行われません。

また、同一セグメント・ディレクティブ内で、複数のマッピング・ディレクティブを記述することができますが、異なるセグメント・ディレクティブであっても、同じ出力セクション名を複数個指定することはできません。同じ出力セクション名が複数個指定された場合、リンカはエラーを出力し、リンクを中止します。

**(b) セクション・タイプ**

出力セクションのタイプを指定します。

セクション作成においては、この出力セクションのタイプ指定は省略できません。

指定できるセクション・タイプとその意味は次のとおりです。

表 A-13 セクション・タイプとその意味

セクション・タイプ名	意味
PROGBITS	オブジェクト・ファイル内に実際の値を持っているセクション テキスト、初期値ありデータ（変数）
NOBITS	オブジェクト・ファイル内に実際の値を持っていないセクション 初期値なしデータ（変数）

セクション・タイプの指定は、“\$” を先頭とし、“\$” の直後に空白を置かないください。

また、“\$” のみが指定された場合、リンカはエラーを出力し、リンクを中止します。

**(c) セクション属性**

作成するセクションの属性を指定します。

セクション作成においては、このセクション属性の指定は省略できません。

指定できるセクション属性とその意味は次のとおりです。

表 A-14 セクション属性とその意味

セクション属性	意味
A	メモリを占有するセクション（すべてのセクションが該当）
W	書き込み可能なセクション（RAM 上に配置するセクション）
X	実行可能なセクション（主にテキスト・セクション）
G	グローバル・ポインタ（gp）と 16 ビットのディスプレイメントを用いて参照することのできるメモリ範囲内に割り付けるセクション（.sdata, .sbss セクション）

セクション属性は、複数個同時に指定でき、A、W、X、G の順番は任意で、空白を置かずに続けて記述します。

また、セクション属性の指定は“?” を先頭とし、“?” の直後に空白を置かないください。

**(d) アドレス**

作成するセクションの開始アドレスを指定します。

セクション作成においては、このアドレスの指定は省略できます。省略した場合、セグメント・ディレクティブで指定されたアドレスに従います。複数セクションがあり、先頭のセクションでなければ、前のセクションの終端に続く値になります。

通常セクションのアドレス指定は、セグメントごとにまとめて行いますが、特定のセクションを特定のアドレスに割り付けたい場合に用いることができます。

.tidata.byte, .tibss.byte 以外のアドレスの値は偶数で指定してください。奇数を指定すると、リンカはメッセージを出力し、指定されたアドレスに 1 を加えたアドレスが指定されたものとみなしてリンクを続行します。アドレスの指定は “V”(大文字・小文字どちらでも可) を先頭とし、“V” の直後に空白を置かないでください。アドレスとして指定できる数値は 10 進数、16 進数のどちらかで、16 進数で指定する場合は数値の前に “0x” を記述してください。なお、アドレスの指定に式を用いることはできません。

**(e) ホール・サイズ**

作成するセクションのホールのサイズを指定します。

セクションのホールとは、セクションとセクションの間の余白を意味します。ホール・サイズの指定を行った場合、そのセクションの終わりにホールが作成されます。

セクション作成においては、このホール・サイズの指定は省略できます。省略した場合、0x0 (バイト) がデフォルトの値として用いられます (ホールは作成しません)。

ホール・サイズの指定は “H”(大文字・小文字どちらでも可) を先頭とし、“H” の直後に空白を置かないでください。なお、ホール・サイズの指定に式を用いることはできません。

**(f) 整列条件**

作成するセクションのメモリ割り付けにおいて、セクションの整列条件 (アラインメント値) を指定します。

セクション作成においては、この整列条件の指定は省略できます。省略した場合はデフォルトの値が使用されますが、次に示すようにセクションの種類によってその値が異なります。

表 A-15 セクションの種類とその整列条件のデフォルト値

セクション名	整列条件
.tidata.byte / .tibss.byte セクション	0x1 (バイト)
上記以外のセクション	0x4 (バイト)

整列条件の指定は “A”(大文字・小文字どちらでも可) を先頭とし、“A” の直後に空白を置かないでください。

整列条件の値として指定できるものは、.tidata.byte, .tibss.byte セクションの場合は偶数が奇数、その他のセクションにおいては偶数のみです。.tidata.byte, .tibss.byte 以外のセクションで奇数を指定すると、リンカはメッセージを出力し、指定された値に 1 を加えた値が指定されたものとみなしてリンクを続行します。なお、整列条件の指定に式を用いることはできません。

**(g) 入力セクション名**

作成する出力セクションの基となる、入力セクション情報を指定します。

セクション作成においては、この入力セクション名の指定、オブジェクト・ファイル名の指定は省略できます。省略した場合、次のような記述の組み合わせにより、出力セクションに出力される情報が変化します。

表 A-16 入力セクションとオブジェクト・ファイルの組み合わせ別の出力

記述パターン	出力
1) 入力セクション名 + オブジェクト・ファイル名	指定した入力セクションを、指定したオブジェクトから抽出して出力
2) 入力セクション名のみ	指定した入力セクションを、すべてのオブジェクトから抽出して出力
3) オブジェクト・ファイル名のみ	作成する出力セクションと同じ属性を持つセクションを、指定されたオブジェクトから抽出して出力
4) 両方とも記述しなかった場合	作成する出力セクションと同じ属性を持つセクションを、すべてのオブジェクトから抽出して出力

具体的な例を示すと次のようになります。

表 A-17 入力セクションとオブジェクト・ファイルの組み合わせの具体例

記述例	出力
SEG1 : !LOAD ?RX { sec1 = \$PROGBITS ?AX usrsec1 ; }	すべてのオブジェクトから、usrsec1 セクションを抽出し、sec1 セクションとして出力
SEG1 : !LOAD ?RX { sec1 = \$PROGBITS ?AX {file1.o file2.o} ; }	file1.o と file2.o にある、\$PROGBITS タイプで、属性 A、X のセクションを抽出し、sec1 セクションとして出力
SEG1 : !LOAD ?RX { sec1 = \$PROGBITS ?AX usrsec1{file1.o} ; }	file1.o にある、usrsec1 セクションを抽出し、sec1 セクションとして出力
SEG1 : !LOAD ?RX { sec1 = \$PROGBITS ?AX ; }	すべてのオブジェクトから、\$PROGBITS タイプで、属性 A、X のセクションを抽出し、sec1 セクションとして出力

なお、セクションの配置時に候補が複数ある場合には、表 A-16 内の [記述パターン] 項目で示した番号を優先順位として（同順位の場合は低位アドレス優先）、セクションを配置します。

入力セクション名は、アプリケーションで設定したセクション名を指定してください。特にアプリケーション中で設定しなかった場合は、デフォルトのセクション名として定義されていますので、そのセクション名を指定してください。デフォルトのセクション名については、「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル C 言語編、またはアセンブリ言語編」を参照してください。

なお、「(a) 出力セクション名」で説明したように、出力セクション名と入力セクション名の対応が固定されているものがあります。これに該当するものは、別の名前のセクション名を指定することはできません。

**(h) オブジェクト・ファイル名**

オブジェクト・ファイル名の指定は、マッピング・ディレクティブの最後に記述し、ファイル名を“{”と“}”で囲んでください。また複数指定するときは「空白」で区切ります（ファイル名に「空白」が含まれている場合は、ファイル名を“ ”で括ります）。

オブジェクト・ファイルを複数指定した場合、指定した順で、下位アドレスから上位アドレスの方向に割り付けられます。ただし、リンカ起動時に指定する「リンクするオブジェクト」の記述順が、リンク・ディレクティブ・ファイルにおける順番と異なる場合、引数で指定したファイル名の順番が優先されます。

```

リンク・ディレクティブ
    sec = $PROGBITS ?AX {file1.o file2.o file3.o}

リンカ起動
    ld850 file3.o file1.o file2.o
           file3.o, file1.o, file2.o の順番で、下位から割り付く

```

マッピング・ディレクティブにおいて、オブジェクト・ファイル名を指定する場合、その属性のセクションを含んでいるファイル名はすべて記述してください。

たとえば、file1.o、file2.o、file3.o、file4.o の 4 つのオブジェクトが存在し、これらすべてに text 属性のセクションが含まれていたとします。このとき

```

TEXT1 : !LOAD ?RX {
        .text1 = $PROGBITS ?AX { file1.o file2.o } ;
} ;
TEXT2 : !LOAD ?RX{
        .text2 = $PROGBITS ?AX { file3.o } ;
} ;

```

というリンク・ディレクティブを記述し、file4.o の text 属性の配置場所を特定しなかった場合、file4.o 内の text 属性を、適当な text 属性のセクションを探して配置します。したがって、意図したとおりのマッピングにならない可能性がありますので注意が必要です（他のどのセクションにも配置されない場合、リンカはメッセージを出力します）。

また、異なるディレクトリに配置した同名ファイルを指定したい場合には、リンク・マップに表示されたパス付きファイル名で次のように指定することができます。

```

textsec1 = $PROGBITS ?AX { c:¥work¥dir1¥file1.o } ;
textsec2 = $PROGBITS ?AX { c:¥work¥dir2¥file1.o } ;
textsec3 = $PROGBITS ?AX { file1.o } ;

```

上記の場合、指定したディレクトリに存在する file1.o は、それぞれ textsec1 / textsec2 に配置され、それ以外は textsec3 に配置されます。なお、この際のパスの指定方法は、リンク・マップに表示された形式のみであるため、記述の際に注意が必要です。

さらに、ライブラリなど、アーカイブ・ファイル中のオブジェクトを、入力オブジェクト名に指定することもできます。たとえば libusr.a というアーカイブ・ファイル中にある lib1.o というオブジェクトを usrlib セクションに出力したい場合、次のように記述します。

```
usrlib = $PROGBITS ?AX { lib1.o(a:¥usrlib¥libusr.a) } ;
```

指定したライブラリ内のオブジェクトすべてを配置指定したい場合は、次のように記述します。

```
usrlib = $PROGBITS ?AX { libusr.a } ;
```

上記の場合、libusr.a 内のオブジェクトは usrlib セクションに配置されます。

#### (i) 同じ指定の場合

複数のセグメントに対して、同じセクション・タイプ/セクション属性/入力セクション名(省略可)/入力ファイル名(入力可)が指定された場合で、それに対応するセクションが存在した場合には、低位アドレスに配置されたセグメントに対し割り付けが行われます。

```
TEXT1 : !LOAD ?RX V0x1000 {
    .text1 = $PROGBITS ?AX.text { file1.o file2.o } ;
} ;
TEXT2 : !LOAD ?RX V0x2000 {
    .text2 = $PROGBITS ?AX.text { file1.o file2.o } ;
} ;
```

上記の場合、セクション・タイプ/セクション属性/入力セクション名/入力ファイル名が TEXT1 と TEXT2 で同じであるため、低位アドレスに配置されている TEXT1 に対して割り付けします。

**(2) マッピング・ディレクティブの指定例**

次のような出力セクションを作りたい場合を例とし、2種類のセクションを作るとします。

表 A-18 マッピング・ディレクティブの指定例

項目	値 - 1	値 - 2
出力セクション名	.text	textsec1
セクション・タイプ	テキスト	テキスト
セクション属性	読み出し可能, 実行可能	読み出し可能, 実行可能
ホール・サイズ	0x10 (バイト)	0x20 (バイト)
フィリング値	0xffff	0xffff
整列条件	0x10 (バイト)	0x10 (バイト)
入力セクション名	.text	usrsec1
オブジェクト・ファイル名	main.o	-

この場合、マッピング・ディレクティブの記述は次のようになります。

```
.text = $PROGBITS ?AX H0x10 F0xffff A0x10 .text {main.o} ;
textsec1 = $PROGBITS ?AX H0x20 F0xffff A0x20 usrsec1 ;
```

## A. 4.5 シンボル・ディレクティブ

ここでは、次の項目でシンボル・ディレクティブの書式について説明します。

(1) 指定項目

(2) シンボル・ディレクティブの指定例

### (1) 指定項目

シンボル・ディレクティブで指定する項目は、次のとおりです。

#### 【tp シンボル】

表 A-19 tp シンボル作成で指定できる項目

項目	指定形式	意味	省略
シンボル名	シンボル名	作成する tp シンボルの名前	不可
シンボル種別	%TP_SYMBOL	作成するシンボル種別（固定）	不可
アドレス	V アドレス	作成する tp シンボルのアドレス	可
整列条件	A 整列条件	シンボル値の整列条件（アラインメント）	可
セグメント名	{セグメント名 セグメント名...}	作成する tp の参照対象としたいセグメント名（複数指定可，空白で区切る）	可

具体的な書式は次のようになります。

```
シンボル名 @ %TP_SYMBOL V アドレス A 整列条件 {セグメント名 セグメント名} ;
```

それぞれの項目は、空白で区切って記述します。また終わりには、必ず ";" を付けます。

「V アドレス」「A 整列条件」「セグメント名」は省略できます。省略した場合はデフォルトの値が使われます。

デフォルトの値は次のとおりです。

表 A-20 tp シンボルのデフォルト値

省略項目	デフォルト値
アドレス	セグメント名が指定されていた場合，そのセグメント内で最下位に割り付けられた text 属性のセクションの先頭アドレス セグメント名が指定されていない場合，ロード・モジュールに存在する text 属性セグメント内で，最下位に割り付けられた text 属性のセクションの先頭アドレス
整列条件	0x4（バイト）
セグメント名	オブジェクトに存在するすべての text 属性セグメントを対象とします

## 【gp シンボル】

表 A-21 gp シンボル作成で指定できる項目

項目	指定形式	意味	省略
シンボル名	シンボル名	作成する gp シンボルの名前	不可
シンボル種別	%GP_SYMBOL	作成するシンボル種別（固定）	不可
ベース・シンボル名	& ベース・シンボル名	gp シンボルを tp シンボルからのオフセット値として指定する際の tp シンボル名	可
アドレス	V アドレス	作成する gp シンボルのアドレス	可
整列条件	A 整列条件	シンボル値の整列条件（アラインメント）	可
セグメント名	{セグメント名 セグメント名...}	作成する gp の参照対象としたいセグメント名 （複数指定可，空白で区切る）	可

具体的な書式は次のようになります。

```
シンボル名 @ %GP_SYMBOL & ベース・シンボル名 V アドレス A 整列条件 {セグメント名 セグメント名};
```

それぞれの項目は，空白で区切って記述します。また終わりには，必ず “;” を付けます。

「V アドレス」「A 整列条件」「セグメント名」は省略できます。省略した場合はデフォルトの値が使われます。

デフォルトの値は次のとおりです。

表 A-22 gp シンボルのデフォルト値

省略項目	デフォルト値
ベース・シンボル名	tp シンボルからのオフセットではなく，gp シンボル値として決定されるアドレス （決定方法は，【gp シンボル値のオフセット指定】を参照）
アドレス	【gp シンボル値の決定規則】に従います
整列条件	0x4（バイト）
セグメント名	オブジェクトに存在するすべての sdata / data / sbss / bss 属性セクションを含むセグメントを対象とします（決定方法は，【gp シンボル値のオフセット指定】を参照）



## 【ep シンボル】

表 A-23 ep シンボル作成で指定できる項目

項目	指定形式	意味	省略
シンボル名	シンボル名	作成する ep シンボルの名前	不可
シンボル種別	%EP_SYMBOL	作成するシンボル種別（固定）	不可
アドレス	V アドレス	作成する ep シンボルのアドレス	可
整列条件	A 整列条件	シンボル値の整列条件（アラインメント）	可

具体的な書式は次のようになります。

```
シンボル名 @ %EP_SYMBOL V アドレス A 整列条件 ;
```

それぞれの項目は、空白で区切って記述します。また終わりには、必ず“;”を付けます。

「V アドレス」「A 整列条件」は省略できます。省略した場合はデフォルトの値が使われます。デフォルトの値は次のとおりです。

表 A-24 ep シンボルのデフォルト値

省略項目	デフォルト値
アドレス	【ep シンボル値の決定規則】に従います
整列条件	0x4（バイト）

次にそれぞれの指定項目について詳しく説明します。

## (a) シンボル名

【該当シンボル：tp, gp, ep】

生成するシンボルの名前を指定します。シンボル作成においては、このシンボル名の指定は省略できません。シンボル名として指定できる文字は、「A.4.1 使用できる文字」で示したもので、任意の長さの文字列を指定できます。

## (b) シンボル種別

【該当シンボル：tp, gp, ep】

tp シンボルを生成するか、gp シンボルを生成するか、ep シンボルを生成するかを指定します。シンボル作成においては、このシンボル種別の指定は省略できません。

ここで指定できる種別は「tp シンボル」「gp シンボル」「ep シンボル」のどれかで、それぞれ「TP\_SYMBOL」「GP\_SYMBOL」「EP\_SYMBOL」です。これ以外の値を指定すると、リンクはエラーを出力します。シンボル種別の指定は“%”を先頭とし、“%”の直後に空白を置かないでください。

**(c) ベース・シンボル名**

【 該当シンボル : gp 】

gp シンボルの生成において、gp シンボル値を定める際に用いる tp シンボルを指定します。ベース・シンボル名を指定すると、tp シンボル値からのオフセット値が gp シンボル値となります。

gp シンボル作成においては、このベース・シンボル名の指定は省略できます。省略した場合は【gp シンボル値の決定規則】に従って決められるアドレスが gp シンボル値となります。

ベース・シンボルの指定は “&” を先頭とし、“&” の直後に空白を置かないでください。“&” の後にベースとしたい tp シンボル名を記述します。

**(d) アドレス**

【 該当シンボル : tp , gp , ep 】

tp シンボル値、gp シンボル値、つまり、アドレスを指定します。

シンボル作成においては、このアドレスの指定は省略できます。アドレスを省略した場合は、次のように決定されます。

表 A-25 tp シンボル、gp シンボルのアドレス指定

シンボル値	決定規則
tp シンボル	セグメント名が指定されていた場合 そのセグメント内で最下位に割り付けられた text 属性のセクションの先頭アドレス セグメント名が指定されていない場合 ロード・モジュールに存在する text 属性セグメント内で、最下位に割り付けられた text 属性のセクションの先頭アドレス
gp シンボル	【gp シンボル値の決定規則】に従います
ep シンボル	【ep シンボル値の決定規則】に従います

アドレスの指定は “V” (大文字・小文字どちらでも可) を先頭とし、“V” の直後に空白を置かないでください。

**(e) 整列条件**

【 該当シンボル : tp , gp , ep 】

作成する tp シンボル値、gp シンボル値、ep シンボル値の設定における整列条件 (アラインメント値) を指定します。

シンボルの作成においては、この整列条件の指定は省略できます。省略した場合はデフォルトの値が使用されます。デフォルト値は 0x4 (バイト) です。

整列条件の指定は “A” (大文字・小文字どちらでも可) を先頭とし、“A” の直後に空白を置かないでください。整列条件の値は偶数で指定してください。奇数を指定すると、リンクはメッセージを出力し、指定された値に 1 を加えた値が指定されたものとみなしてリンクを続行します。なお、整列条件の指定に式を用いることはできません。

**(f) セグメント名**

【 該当シンボル : tp , gp 】

作成する tp シンボル値, gp シンボル値の参照対象とするセグメント名を指定します。

つまり, 作成する tp シンボル, gp シンボルで参照したいセグメントを指定します。参照対象のセグメントは複数指定することができます。

シンボルの作成においては, このセグメント名の指定は省略できます。省略した場合はデフォルトとして, 次のように指定されたものとみなされます。

表 A-26 tp シンボル, gp シンボルの参照対象セグメント名

シンボル値	決定規則
tp シンボル	オブジェクトに存在するすべての text 属性セグメントを対象とします
gp シンボル	オブジェクトに存在するすべての sdata / data / sbss / bss 属性セクションを含むセグメントを対象とします (決定方法は【gp シンボル値のオフセット指定】を参照)

なお, gp シンボル参照の対象とするセグメント名には, gp 相対参照が前提となっているセグメント名を指定してください。

たとえば, ep 相対参照が前提となっている .sdata や .sebss セクションを含むセグメントは指定しないでください。これらの詳細は, 「A.3.2 グローバル・ポインタ (gp)」を参照してください。

セグメント名の指定は, シンボル・ディレクティブの最後に記述し, セグメント名 “{” と “}” で囲んでください。また複数指定するときは「空白」で区切ります。

## (2) シンボル・ディレクティブの指定例

次のようなシンボルを作成する場合とします。

表 A-27 シンボル・ディレクティブの指定例

シンボル	指定項目	指定値
tp シンボル	シンボル名	__tp_TEXT
	参照対象セグメント名	TEXT1
gp シンボル	シンボル名	__gp_DATA
	オフセット指定シンボル	__tp_TEXT
	参照対象セグメント名	DATA1, DATA2
ep シンボル	シンボル名	__ep_TEXT
	アドレス	0xFFFFD000

この場合, シンボル・ディレクティブの記述は次のようになります。

```
__tp_TEXT @ %TP_SYMBOL{TEXT1} ;
__gp_DATA @ %GP_SYMBOL &__tp_TEXT{DATA1 DATA2} ;
__ep_DATA @ %EP_SYMBOL 0xFFFFD000 ;
```

なお, シンボル・ディレクティブの指定がされていない場合, シンボルは作成されないため注意が必要です。

## A.5 デフォルト

ユーザがリンク・ディレクティブ・ファイルを作成していない、または参照するリンク・ディレクティブ・ファイルを指定しないでリンクを行った場合、CA850 は内部で持っているデフォルトのリンク・ディレクティブを用いてリンクを行います。

デフォルトのリンク・ディレクティブに記述されている各セグメントは、対応する入力セクションが存在する場合、セクションが現れた順番に下位アドレスから割り当てて生成されます。

また、セグメントの割り付けアドレスには、リンク・ディレクティブで記述されている順番に割り当てられるものと、デバイス・ファイルやリンカの情報に従った値が割り当てられるものがあります。

なお、デバイス・ファイルで定義されている割り込み要求名を使用して割り込みハンドラを定義した場合、デフォルトのディレクティブ、指定されたディレクティブに関わらず、決められたハンドラ・アドレスにその関数を割り付けるリンク・ディレクティブが、リンカ内部で自動的に生成されます。

割り込み要求名でマッピング・ディレクティブを記述した場合、セクション名の多重定義によりエラーとなりますので、注意が必要です。

### 【SIDATA セグメントが生成された場合】

- SIDATA セグメントは、内蔵 RAM の先頭アドレスへ割り付けられます。
- DATA セグメントは、デバイス・ファイルに従って、デバイスの種類に適したアドレスへ割り付けられます。たとえば、内蔵 ROM を持つ V851 では、外部メモリの先頭アドレスへ割り付けられます。
- SEDATA セグメントは、内蔵 RAM の先頭アドレスより下位のアドレスへ割り付けられます。

### 【SIDATA セグメントが生成されない場合】

- DATA セグメントは、内蔵 RAM の先頭アドレスへ割り付けられます。
- CONST セグメントは、デバイス・ファイルに従って、デバイスの種類に適したアドレスへ割り付けられません。たとえば、内蔵 ROM を持つ V851 では、外部メモリの先頭アドレスへ割り付けられます。
- SEDATA セグメントは、内蔵 RAM の先頭アドレスより下位のアドレスへ割り付けられます。

上記以外のセグメントのアドレスは、記述されている順番に割り付けられます。

なお、デフォルトのリンク・ディレクティブは、あくまでもサンプル的な扱いですので、一般的にはユーザ自身がリンク・ディレクティブ・ファイルを記述し、それをリンク・ディレクティブとしてください。デフォルトのリンク・ディレクティブの内容は、パッケージの中にサンプルとして提供しています。これらを参考にし、書き換えて使用してください。

## A.6 リンク・ディレクティブ・ファイルの記述例

ここでは、よく利用される記述を例にあげます。

なお、マッピング・ディレクティブにおける入力セクション名の記述パターンにより、セクション配置の優先度が異なることを考慮して記述を行ってください ([\(g\) 入力セクション名参照](#))。

- (1) 全オブジェクト内で「text 属性 (セクション・タイプ PROGBITS, セクション属性 AX) で、セクション名が .text (.text セクションのデフォルト名) であるセクション」を「セグメント TEXT」に割り付けたい場合

```
TEXT : !LOAD ?RX{
      .text = $PROGBITS ?AX .text ;
};
```

- (2) オブジェクト・ファイル file1.o , および file2.o 内の「text 属性セクション (セクション・タイプ PROGBITS, セクション属性 AX) を持つセクション」を「セグメント TEXT1」に割り付けたい場合

```
TEXT1 : !LOAD ?RX{
        sec = $PROGBITS ?AX{file1.o file2.o} ;
};
```

- (3) オブジェクト・ファイル file1.o , および file2.o 内で「text 属性 (セクション・タイプ PROGBITS, セクション属性 AX) で、セクション名を usrsec と名付けたセクション」を「セグメント USRTEXT」に割り付けたい場合

```
USRTEXT : !LOAD ?RX{
          usrsec = $PROGBITS ?AX usrsec{file1.o file2.o} ;
};
```

- (4) C ソース内で「#pragma text "funcsec1" func1」と記述し、「funcsec1」という独自に作成した text 属性のセクションに、関数「func1」を割り当てた場合 (セグメント名: FUNC1)

```
FUNC1 : !LOAD ?RX{
        funcsec1.text = $PROGBITS ?AX funcsec1.text ;
};
```

#pragma text 指令で、特定の関数を独自に指定した text 属性のセクションに配置する場合、実際に生成されるセクション名は「指定した文字列 + .text」となり、このセクション名をリンク・ディレクティブに記述する必要があります。この例であれば「funcsec1.text セクション」になります。詳しくは、「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル C 言語編」を参照してください。

- (5) オブジェクト・ファイル file1.o, file2.o, file3.o があり, file1.o, file2.o のテキストは 0x100000 番地に, file3.o のテキストは 0x120000 番地に, 別々に割り付けたい場合

```
TEXT1 : !LOAD ?RX V0x100000{
    .text1 = $PROGBITS ?AX{file1.o file2.o} ;
};
TEXT2 : !LOAD ?RX V0x120000{
    .text2 = $PROGBITS ?AX{file3.o} ;
};
```

**【注意】** 出力セクション名が同じにならないように注意してください。

- (6) 異なるディレクトリに配置した同名オブジェクトを指定したい場合

```
SEG : !LOAD ?RX {
    textsec1 = $PROGBITS ?AX { c:¥work¥dir1¥file1.o } ;
    textsec2 = $PROGBITS ?AX { c:¥work¥dir2¥file1.o } ;
    textsec3 = $PROGBITS ?AX { file1.o } ;
};
```

- (7) ライブラリ libusr.a 内のオブジェクトすべてを usrlib セクションに配置したい場合

```
SEG : !LOAD ?RX {
    usrlib = $PROGBITS ?AX { libusr.a } ;
};
```

- (8) ライブラリ libusr.a (c:\usrlib にある) 内のオブジェクト libobj1.o を usrlib セクションに配置したい場合

```
SEG : !LOAD ?RX {
    usrlib = $PROGBITS ?AX { libobj1.o(c:¥usrlib¥libusr.a) } ;
};
```

- (9) オブジェクト・ファイル file1.o, および file2.o 内の「セクション・タイプ PROGBITS, セクション属性 AW を持つセクション」と「セクション・タイプ NOBITS, セクション属性 AW を持つセクション」を「セグメント SEG」に割り付けたい場合

```
SEG : !LOAD ?RW{
    sec1 = $PROGBITS ?AW{file1.o file2.o} ;
    sec2 = $NOBITS ?AW{file1.o file2.o} ;
};
```

- (10) オブジェクト・ファイル file1.o , および file2.o 内の「data 属性セクション (セクション・タイプ PROGBITS とセクション属性 AW を持つセクション)」と「sdata 属性セクション (セクション・タイプ PROGBITS とセクション属性 AWG を持つセクション)」を「セグメント SEG」に割り付けたい場合

```
SEG : !LOAD ?RW{
    .data = $PROGBITS ?AW{file1.o file2.o} ;
    .sdata = $PROGBITS ?AWG{file1.o file2.o} ;
};
```

- (11) 全オブジェクトで gp 相対で参照する「data 属性セクション」「bss 属性セクション」「sdata 属性セクション」「sbss 属性セクション」を「セグメント DATA」としてまとめ、割り付けたい場合

```
DATA : !LOAD ?RW{
    .data = $PROGBITS ?AW ;
    .sdata = $PROGBITS ?AWG ;
    .sbss = $NOBITS ?AWG ;
    .bss = $NOBITS ?AW ;
};
```

- (12) C ソース内で “ # pragma section 指令 ” を用いて

```
#pragma section data "data1" begin
int a = 10 ;
int b ;
#pragma section data "data1" end
```

と記述して独自に作成した data 属性, bss 属性セクションに変数を割り当てた場合のリンク・ディレクティブの記述方法 (セグメント名: USRDATA)

```
USRDATA : !LOAD ?RW{
    data1.data = $PROGBITS ?AW data1.data ;
    data1.bss = $NOBITS ?AW data1.bss ;
};
```

この場合、初期値を持つ変数 “ a ” は「data1.data セクション」に、初期値を持たない変数 “ b ” は「data1.bss セクション」に割り当てられます。このように実際に生成されるセクション名は “ 指定した文字列 + .data ”, または “ 指定した文字列 + .bss ” となり、このセクション名をリンク・ディレクティブに記述する必要があります。詳しくは、「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル C 言語編」を参照してください。

## (13) C ソース内で “ # pragma section 指令 ” を用いて

```
# pragma section const "const1" begin
const int c = 10 ;
# pragma section const "const1" end
```

と記述して独自に作成した const 属性セクションに変数を割り当てた場合のリンク・ディレクティブの記述方法（セグメント名：USRCONST）

```
USRCONST : !LOAD ?R{
    const1.const = $PROGBITS ?A const1.const ;
};
```

この場合、変数 “ c ” は「const1.const セクション」に割り当てられます。このように実際に生成されるセクション名は“ 指定した文字列 + .const ” となり、このセクション名をリンク・ディレクティブに記述する必要があります。なおこの規則は sconst セクションにも適用されます。詳しくは、「CA850 C コンパイラ・パッケージ ユーザーズ・マニュアル C 言語編」を参照してください。

## (14) 全モジュールで「data 属性セクション / sdata 属性セクション」と「bss 属性セクション / sbss 属性セクション」を分けて配置し、gp シンボルを 1 つだけ生成する場合

```
TEXT : !LOAD ?RX V0x1000{
    .text = $PROGBITS ?AX .text ;
};
DATA1 : !LOAD ?RW V0x10000{
    .data = $PROGBITS ?AW .data ;
    .sdata = $PROGBITS ?AWG .sdata ;
};
DATA2 : !LOAD ?RW V0x12000{
    .sbss = $NOBITS ?AWG .sbss ;
    .bss = $NOBITS ?AW .bss ;
};
__tp_TEXT @ %TP_SYMBOL{TEXT} ;
__gp_DATA @ %GP_SYMBOL{DATA1 DATA2} ;
```



- (15) 全モジュールで「data 属性セクション / sdata 属性セクション」と「bss 属性セクション / sbss 属性セクション」を分けて配置し、gp シンボルを DATA1, DATA2 ごとに生成する場合

```

TEXT : !LOAD ?RX V0x1000{
    .text = $PROGBITS ?AX .text ;
};
DATA1 : !LOAD ?RW V0x100000{
    .data = $PROGBITS ?AW .data ;
    .sdata = $PROGBITS ?AWG .sdata ;
};
DATA2 : !LOAD ?RW V0x12000{
    .sbss = $NOBITS ?AWG .sbss ;
    .bss = $NOBITS ?AW .bss ;
};
__tp_TEXT @ %TP_SYMBOL{TEXT} ;
__gp_DATA1 @ %GP_SYMBOL{DATA1} ;
__gp_DATA2 @ %GP_SYMBOL{DATA2} ;

```

- (16) 全モジュールで「text 属性」が複数存在し、「sdata 属性セクション」、または「sbss 属性セクション」を持つセグメントも複数存在。「sdata 属性セクション / sbss 属性セクション」ごとに gp を持ち、ベース・シンボルもそれぞれ持つ場合

```

TEXT1 : !LOAD ?RX V0x1000{
    text1 = $PROGBITS ?AX{start.o main.o} ;
};
TEXT2: !LOAD ?RX{
    text2 = $PROGBITS ?AX{func.o} ;
};
TEXT3: !LOAD ?RX{
    text3 = $PROGBITS ?AX{libfunc.o(c:¥usrlib¥libusr.a)} ;
};
DATA1: !LOAD ?RW V0x100000{
    sdata = $PROGBITS ?AWG ;
};
DATA2: !LOAD ?RW{
    sbss1 = $NOBITS ?AWG{start.o} ;
};
DATA3: !LOAD ?RW{
    sbss2 = $NOBITS ?AWG ;
};
__tp_symbol1 @ %TP_SYMBOL {TEXT1} ;
__tp_symbol2 @ %TP_SYMBOL {TEXT2} ;
__tp_symbol3 @ %TP_SYMBOL {TEXT3} ;

__gp_symbol1 @ %GP_SYMBOL &__tp_symbol1{DATA1} ;
__gp_symbol2 @ %GP_SYMBOL &__tp_symbol2{DATA2} ;
__gp_symbol3 @ %GP_SYMBOL &__tp_symbol3{DATA3} ;

```

なお、C 言語で生成される .sbss セクションに配置される変数のうち、外部結合仮定義 (int i;) であるものについては、ANSI の規定により、複数の仮定義が許されており、リンカが複数の中からサイズを考慮して変数領域を生成しています。

つまり、.sbss セクションに関しては、入力ファイル名による配置指定が難しく、また、すべての sbss 属性 (\$NOBITS,?AWG) に入力ファイル名を指定してしまうと、リンカが生成した変数の領域を配置するセクショ

ンがなくなってしまう。こういった場合は、入力ファイル名を指定しない "\$NOBITS ?AWG" 属性のセクションを作成しておいてください。

上記の例では、sbss1 セクションのほか、sbss2 が定義されており、その入力セクション名指定が省略してあるためこの問題は発生しません。

- (17) V850 コアにて内蔵 RAM を使用し、かつ内蔵 RAM 専用セクションを使用したセグメントを生成したい場合

```
SIDATA : !LOAD ?RW V0xffffe000{
    .tidata.byte = $PROGBITS ?AW .tidata.byte ;
    .tibss.byte = $NOBITS ?AW .tibss.byte ;
    .tidata.word = $PROGBITS ?AW .tidata.word ;
    .tibss.word = $NOBITS ?AW .tibss.word ;
    .sidata = $PROGBITS ?AW .sidata ;
    .sibss = $NOBITS ?AW .sibss ;
};
```

このとき指定するアドレスは、そのデバイスが持つ内蔵 RAM の先頭アドレスにするのが一般的です。使用するデバイスの内蔵 RAM の先頭アドレスについては、使用する CPU のユーザーズ・マニュアル ハードウェア編を参照してください。

なおこの例では、内蔵 RAM にて指定できるセクションすべてを記述してありますので、もし必要のないセクションがあれば削除してください。

- (18) 割り込みハンドラのリンク・ディレクティブ

```
【例 1】
#pragma interrupt INTP100 func func への分岐が INTP100 となる
「このとき内部的に追加されるディレクティブ」

INTP100 : !LOAD ?RX V0xc0{
    INTP100 = $PROGBITS ?AX INTP100 ;
};

【例 2】
.section "INTP120", text INTP120 セクションを定義する
「このとき内部的に追加されるディレクティブ」

INTP120 : !LOAD ?RX V0x100{
    INTP120 = $PROGBITS ?AX INTP120 ;
};
```

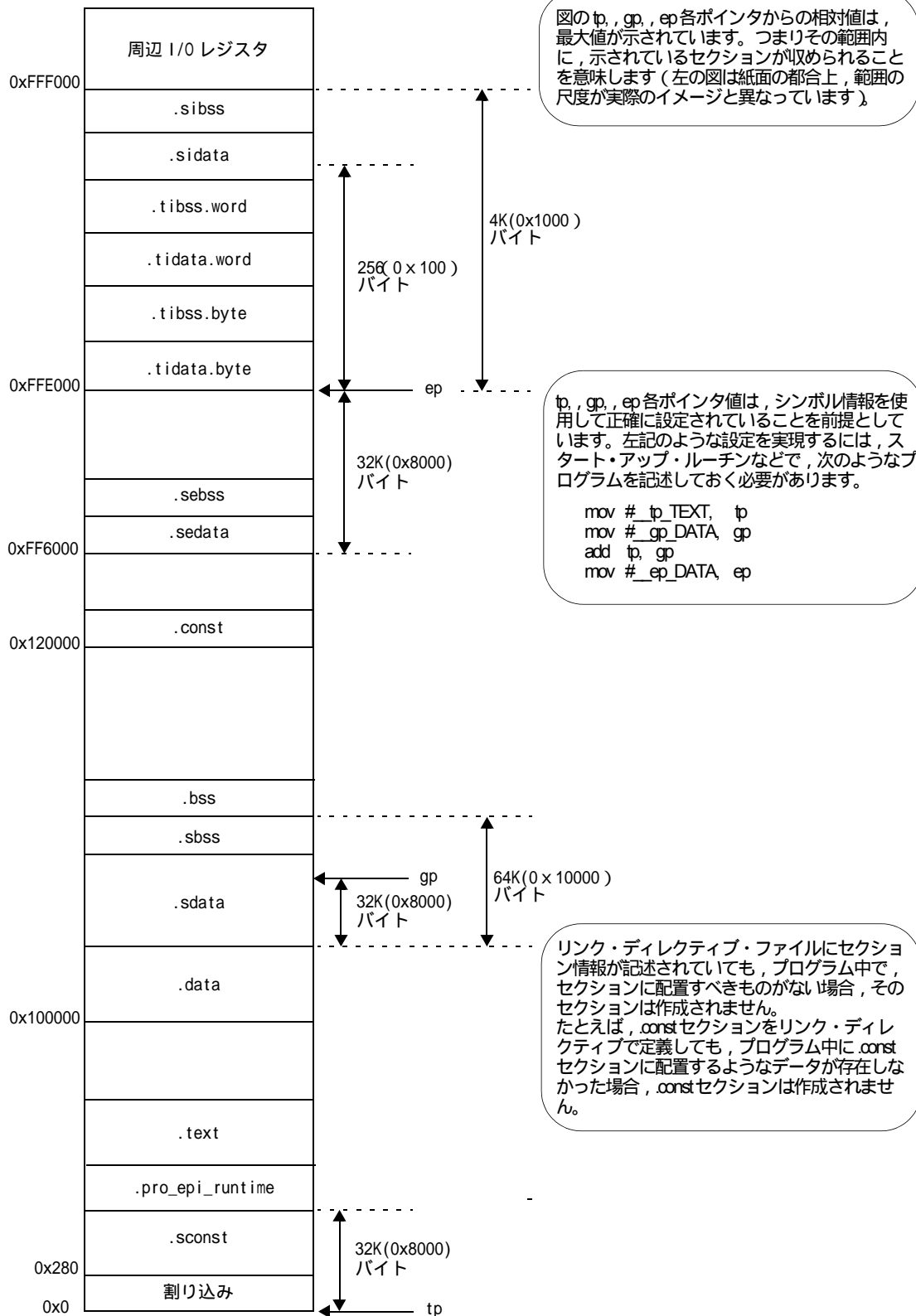
割り込みハンドラの割り付けは、リンカが自動的に行いますので、ユーザがリンク・ディレクティブに特別な記述を行う必要はありません。

C ソース・ファイル内で “ #pragma interrupt 指令 ” によって割り込みハンドラを作成した場合、およびアセンブラ・ソース内で “ .section 疑似命令 ” によって割り込み要求名を指定したセクション定義を行った場合、デバイス・ファイルで規定されたアドレスに、割り込みハンドラとして定義されたセクションが割り付けられます。なお、この例で #pragma interrupt に direct を指定した場合、関数 func への分岐ではなく、関数本体が INTP100 セクションとなります。

【リンク・ディレクティブ・ファイルの例】

<pre> SCONST : !LOAD ?R V0x280{     .sconst = \$PROGBITS ?A .sconst; }; </pre>	<p>} 内蔵ROMへ SCONST セグメントを配置。 割り込みハンドラ・アドレス群の後ろから配置するため 0x280 から配置 (0x280 は V853 の例。他の CPU の場合は変更が必要)。</p>
<pre> TEXT : !LOAD ?RX {     .pro_epi_runtime= \$PROGBITS ?AX.pro_epi_runtime;     .text = \$PROGBITS ?AX; }; </pre>	<p>} 開始アドレス指定されていないため、TEXT セグメントは SCONST セグメントの直後から配置される。つまり SCONST セグメントの大きさによって開始アドレスは変化する。ただし、SCONST セグメントに配置すべきデータがなければ、定義されている割り込みハンドラ・アドレスの直後に TEXT セグメントは配置されるので注意が必要 (0x280 番地は無視される)。その場合は TEXT セグメントにアドレスを指定する。</p>
<pre> DATA : !LOAD ?RW 0x100000{     .data = \$PROGBITS ?AW;     .sdata = \$PROGBITS ?AWG;     .sbss = \$NOBITS ?AWG;     .bss = \$NOBITS ?AW; }; </pre>	<p>} gp 相対で参照する DATA セグメントを 0x100000 番地に配置 (0x100000 番地 ~ は RAM 領域であることが前提条件)。</p>
<pre> CONST : !LOAD ?R V0x120000{     .const = \$PROGBITS ?A .const; }; </pre>	<p>} CONST セグメントを配置 (0x120000 番地 ~ は ROM 領域であることが前提条件)。</p>
<pre> SEDATA : !LOAD ?RW V0xff6000 {     .sedata = \$PROGBITS ?AW .sedata;     .sebss = \$NOBITS ?AW .sebss; }; </pre>	<p>} ep 相対でアクセスする SEDATA セグメントを配置。 内蔵 RAM の先頭アドレスから 32K バイト内に配置する (0xff6000 番地 ~ は RAM 領域であることが前提条件)。</p>
<pre> SIDATA : !LOAD ?RW V0xffe000 {     .tidata.byte = \$PROGBITS ?AW .tidata.byte;     .tibss.byte = \$NOBITS ?AW .tibss.byte;     .tidata.word = \$PROGBITS ?AW .tidata.word;     .tibss.word = \$NOBITS ?AW .tibss.word;     .sidata = \$PROGBITS ?AW .sidata;     .sibss = \$NOBITS ?AW .sibss; }; </pre>	<p>} ep 相対で参照する SIDATA セグメントを内蔵 RAM 領域に配置。 このセグメントの先頭アドレスは、内蔵 RAM 領域の先頭であることが一般的。 なお、CPU により内蔵 RAM の先頭アドレスは異なるため、使用する CPU に合わせる。</p>
<pre> __tp_TEXT @ %TP_SYMBOL; __gp_DATA @ %GP_SYMBOL &amp; __tp_TEXT {DATA}; __ep_DATA @ %EP_SYMBOL; </pre>	<p>} tp, gp, ep のシンボル情報を生成する。この場合、__tp_TEXT は 0x0 番地、__gp_DATA は「.sdata セクションの先頭 - __tp_TEXT+0x8000」の値、__ep_DATA は内蔵 RAM の先頭アドレス (0xffe000 番地) が設定される。</p>

【配置イメージ】



# 総合索引

## 【B】

bss 属性 ... 87

## 【C】

const 属性 ... 87

## 【D】

data 属性 ... 87

## 【E】

ep ... 92

ep シンボル ... 111

## 【G】

gp ... 89

gp シンボル ... 110

## 【P】

PM+ ... 16

## 【S】

sbss 属性 ... 87

sdata 属性 ... 87

## 【T】

text 属性 ... 87

tp ... 88

tp シンボル ... 109

## 【あ行】

[新しいリンク・ディレクティブ]ダイアログ ... 45

インストール ... 17

エラー・メッセージ ... 72

エレメント・ポインタ (ep) ... 92

オブジェクト・ファイルの追加 ... 21

オブジェクト・ファイル表示 ... 30

オブジェクト・ファイル名 ... 106

[オブジェクト・ファイルを選択]ダイアログ ... 55

[オプション]ダイアログ ... 67

## 【か行】

[環境選択]ダイアログ ... 47

起動方法 ... 18

グループ化 ... 34

グローバル・ポインタ (gp) ... 89

警告メッセージ ... 72

[検索]ダイアログ ... 53

コンパイラ ... 16

## 【さ行】

最大メモリ・サイズ ... 98

システム構成 ... 15

出力セクション ... 102

新規作成 ... 20

シンボル ... 111

シンボル表示 ... 31

[シンボルを追加]ダイアログ ... 64

シンボル・ディレクティブ ... 78, 88

シンボル種別 ... 111

整列条件 ... 99, 104, 112

セクション ... 79

セクション属性 ... 103

セクションの追加 ... 21

セクションの属性 ... 86

セクション表示 ... 29

[セクションを追加]ダイアログ ... 60

セクション・タイプ ... 86, 103

セグメント ... 34, 79

セグメント属性 ... 97

セグメント名 ... 96, 112

セグメント・タイプ ... 97

セグメント・ディレクティブ ... 77, 95

## 【た行】

テキスト・ポインタ (tp) ... 88

デバイス・ファイル ... 15

動作環境 ... 16

ドラッグ・アンド・ドロップ機能 ... 36

## 【な行】

[名前を付けて保存]ダイアログ ... 51

入力セクション ... 105

## 【は行】

背景色 ... 28, 30, 69  
情報メッセージ ... 76  
[開く]ダイアログ ... 49  
フィリング値 ... 98  
ベース・シンボル名 ... 112  
ホール・サイズ ... 98, 104  
ホスト・マシン ... 16  
ポップ・アップ表示時間 ... 69

## 【ま行】

マッピング・ディレクティブ ... 77, 101  
ミラー・イメージ ... 27, 70  
メイン・ウインドウ ... 25  
    コンテキスト・メニュー ... 33  
    ツールバー ... 44  
    プロパティ・ビュー・エリア ... 37  
    メッセージ・ビュー・エリア ... 40  
    メニューバー ... 41  
    メモリ・マッピング・ビュー・エリア ... 26  
メッセージ ... 71  
メモリの追加 ... 21  
メモリ表示 ... 27  
[メモリを追加]ダイアログ ... 57  
文字色 ... 69  
元に戻す回数 ... 69

(メモ)

## 【発 行】

NEC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

電話（代表）：044(435)5111

—— お問い合わせ先 ——

---

## 【ホームページ】

NECエレクトロニクスの情報がインターネットでご覧になれます。

URL(アドレス) <http://www.necel.co.jp/>

---

## 【営業関係，技術関係お問い合わせ先】

半導体ホットライン

（電話：午前 9:00～12:00，午後 1:00～5:00）

電 話     : 044-435-9494

E-mail    : info@necel.com

---

## 【資料請求先】

NECエレクトロニクスのホームページよりダウンロードいただくか，NECエレクトロニクスの販売特約店へお申し付けください。

---