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い申し上げます。

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

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ものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
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て、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がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

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

RENESAS

ユーザーズ・マニュアル

WB77016

ワークベンチ

操作編（Windows™ ベース）

μPD77015
μPD77016
μPD77017
μPD77018
μPD77018A
μPD77019
μPD77110
μPD77111
μPD77112
μPD77113
μPD77114

資料番号　U11506JJ3V0UMJ1（第 3 版）
発行年月　February　2001 NS　CP（K）

[メ　モ]

保守／廃止

目 次 要 約

[メ　モ]

保守／廃止

MS-DOS，Windows，Windows NT，および Microsoft 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。
PC/AT は米国 IBM 社の商標です。
InterTools は，米国 TASKING, Inc.の商標です。

- 本資料の内容は予告なく変更することがありますので，最新のものであることをご確認の上ご使用ください。
- 文書による当社の承諾なしに本資料の転載複製を禁じます。
- 本資料に記載された製品の使用もしくは本資料に記載の情報の使用に際して，当社は当社もしくは第三者の知的財産権その他の権利に対する保証または実施権の許諾を行うものではありません。上記使用に起因する第三者所有の権利にかかわる問題が発生した場合，当社はその責を負うものではありませんのでご了承ください。
- 本資料に記載された回路，ソフトウエア，及びこれらに付随する情報は，半導体製品の動作例，応用例を説明するためのものです。従って，これら回路・ソフトウエア・情報をお客様の機器に使用される場合には，お客様の責任において機器設計をしてください。これらの使用に起因するお客様もしくは第三者の損害に対して，当社は一切その責を負いません。

M7A 98.8

保守／廃止

本版で改訂された主な箇所

箇　所	内　容
全　般	対象デバイスにμPD77018A, 77019, 77110, 77111, 77112, 77113,77114 を追加。
全　般	対象デバイスに対応して，メニュー・コマンド，モデル・ファイルを変更。
p.21	1.1 特　徴に説明を追加
p.23	1.2 オーダ名称と対象 OS に Window 98, Window NT 4.0 を追加。
p.26	2.2 SP77016 のセットアップを追加。
p.32	3.3 C エラー・ファイルを追加
p.37	3.7 ASC II プロジェクト・ファイル出力を追加
p.47	3.8 セグメント割り当て順序ファイルを追加
p.57	第 5 章　チュートリアルを追加
p.61	第 6 章　メニュー・コマンドとダイアログを変更
p.117	付録 A　リンカを追加
p.145	付録 C　モデルを追加
p.161	付録 E　μPD77016 ライブラリアンを追加
p.163	付録 F　リリース・ノートを追加

本文欄外の★印は，本版で改訂された主な箇所を示しています。

巻末にアンケート・コーナを設けております。このドキュメントに対するご意見をお気軽にお寄せください。

保守／廃止

はじめに

対象者　このマニュアルは，ディジタル・シグナル・プロセッサμPD77016ファミリおよびμPD77116の機能を理解し，それを用いたソフトウエア，ハードウエアなどのアプリケーション・システムを設計するユーザを対象とします。

μPD77016ファミリは，μPD7701xファミリ（μPD77015, 77016, 77017, 77018A, 77019）とμPD77111ファミリ（μPD77110, 77111, 77112, 77113, 77114）の総称です。特に機能面に違いがない場合は，それぞれのファミリを該当する製品に読み替えてご使用ください。機能面に違いがある場合は，製品名をあげて説明しています。

このマニュアルには，μPD77116，NA77113に関する記述がありますが，本バージョンのWB77016は，μPD77116，NA77113には正式に対応しておりませんので，注意してください。

目　的　WB77016は，μPD77016ファミリの応用システムを設計，開発する際に，プログラムの編集からオブジェクト・プログラムの作成，ソフトウエア・シミュレータの起動を一貫して行うサポート・ソフトウエアです。
このマニュアルは，WB77016を用いて効率よくプログラムを開発していただくことを目的としています。

構　成　このマニュアルでは，大きく分けて次の内容で構成しています。

- ●第1章　概　説
- ●第2章　セットアップ
- ●第3章　ファイル・フォーマット
- ●第4章　WB77016ウインドウ
- ●第5章　チュートリアル
- ●第6章　メニュー・コマンドとダイアログ
- ●付録A　リンカ
- ●付録B　エラーとワーニング
- ●付録C　モデル
- ●付録D　WB77016のキー
- ●付録E　μPD77016ライブラリアン
- ●付録F　リリース・ノート
- ●付録G　索　引

読 み 方　**このマニュアルを読むにあたっては，電気，論理回路，マイクロコンピュータに関する一般知識や，** Microsoft™ Windows™ 95, 98, Windows NT™ 4.0 **の操作に関する一般的知識が必要となります。**

一通り WB77016 **の機能を理解しようとするとき**

→目次に従ってお読みください。

メッセージの意味，原因などを知りたいとき

→付録 B　**エラーとワーニング**をご覧ください。

凡　　例

↵	**：キャリッジ・リターン・キーを示します。**
↑	**：↑キーを示します。**
↓	**：↓キーを示します。**
→	**：→キーを示します。**
←	**：←キーを示します。**
Esc	**：**Esc **キーを示します。**
Ctrl	**：**Ctrl **キーを示します。**
カナ	**：かなキーを示します。**
Alt	**：**Alt **キーを示します。**
Ctrl＋	**：**Ctrl **キーを押しながら他のキーを押すことを示します。**
Shift＋	**：**Shift **キーを押しながら他のキーを押すことを示します。**
Alt＋	**：**Alt **キーを押しながら他のキーを押すことを示します。**
F1－F12	**：ファンクション・キーを示します。**
注	**：本文中につけた注の説明**
注意	**：気をつけて読んでいただきたい内容**
備考	**：本文中の補足説明**
数の表記	**：**2 **進数…××××または** 0b××××
	10 **進数…××××**
	16 **進数…**0x××××

保守／廃止

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

μPD77016 **ファミリに関する資料**

資料名 / 品名	パンフレット	データ・シート	ユーザーズ・マニュアル		アプリケーション・ノート	
			アーキテクチャ編	命令編	基本ソフトウエア編	ライブラリ編
μPD77016	U12395J	U10891J	U10503J	U13116J	U11958J	U12021J
μPD77015		U10902J				
μPD77017						
μPD77018						
μPD77018A		U11849J				
μPD77019						
μPD77019-013		U13053J				
μPD77110		U12801J	U14623J			
μPD77111						
μPD77112						
μPD77113		U14373J				
μPD77114						
μPD77116	－	U14624J	－	－	－	－

開発ツールに関する資料

資　料　名		資料番号
SM77016　**ユーザーズ・マニュアル**		U11602J
WB77016　**ユーザーズ・マニュアル**	**言語編**	U10078J
	操作編	**このマニュアル**
ID77016　**ユーザーズ・マニュアル**		U10118J
IE-77016-98，IE-77016-PC　**ユーザーズ・マニュアル**	**ハードウエア編**	U13044J
μPD77016　**スタータ・キット　ユーザーズ・マニュアル**		U13032J
IE-77016-CM-LC　**ユーザーズ・マニュアル**		U14139J
RX77016　**ユーザーズ・マニュアル**	**機能編**	U14397J
	コンフィギュレーション・ツール編	U14404J
RX77016　**アプリケーション・ノート**	HOST API 編	U14371J

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

[メ　モ]

保守／廃止

目　　次

保守／廃止

図の目次（1/2）

図番号　　タイトル，ページ

図の目次（2/2）

保守／廃止

表の目次

表番号 **タイトル，ページ**

[メ　モ]

保守／廃止

第1章　概　　説

μPD77016 ワークベンチ（以下 WB77016）は，16 ビット固定小数点ディジタル・シグナル・プロセッサμPD77016 ファミリ[注]，μPD77116，NA77113 用の統合化開発環境です。アセンブラ，リンカ，エディタ，make ユーティリティを一体化したプログラム開発環境を提供します。

このマルチタスク設計環境は，μPD77016 ライブラリアン（WB77016 に添付）やμPD77016 シミュレータ（以下 HSM77016），μPD77016 ディバッガ（以下 ID77016）など，ほかの DSP ツールと共に使用することができ，μPD77016 アルゴリズムの実装とテストを大幅に加速します。

WB 77016 は，より高速で効率的な DSP プログラム開発を可能にする多くの機能を提供します。

注　μPD77016 ファミリは，μPD7701x ファミリ（μPD77015，77016，77017，77018，77018A，77019）とμPD77111 ファミリ（μPD77110，77111，77112，77113，77114）の総称です。

注意　このドキュメントにμPD77116，NA77113（μPD77113 **の** DSP **コア製品）についての記述がありますが，現在の** WB77016 **は**μPD77116，NA77113 **には正式には対応していません。**

★ 1.1 特　　徴

WB77016 には，次の機能があります。

- マルチファイル・エディタ
- プロジェクト管理（make ユーティリティ）
- アセンブラ
- リンカ
- HEX コンバータ

1.1.1 統合型マルチファイル・エディタ

このエディタを使用すると，μPD77016 ファミリのソース・コードをきわめて効率的に作成し，検証することができます。プログラムのアセンブル時にエラーが検出されると，このエディタは自動的にエラーの位置で一時停止するので，ユーザは，そこでディバグしたのち make ユーティリティを再度起動して必要な操作を行うことができます。

複数のファイルを同時に表示，編集できるため，ソース・コードの作成やディバグを簡単に行うことができます。

1.1.2 プロジェクト管理（make ユーティリティ）

プログラム開発サイクル全体の処理が自動化されるため，手作業の全ステップを最小にすることができます。ユーザは，アセンブラやリンカなどのスタンドアロン・ツールを手動で呼び出す手間を省けます。ソース・コードが変更されたときだけ作業を行うので，開発時間を大幅に短縮できます。モジュール・バージョン管理の自動化により，ユーザが，旧プログラマ・モジュール・バージョンを処理する必要がないため，エラーが発生する心配もありません。

1.1.3 C ソース拡張機能

HSM77016 と ID77016 による C ソース・レベルのディバグのために，InterTools™77016 C コンパイラが生成するアセンブラ・ファイルを扱う機能があります。

注意 InterTools 77016 C コンパイラは，日本タスキング株式会社の製品です。

お問い合わせ先

〒105-0014　東京都港区芝 1-15-13　芝エステービル　6 F

Tel：03-3457-6831　　Fax：03-3457-6834

1.1.4 C エラー・ファイル

InterTools 77016 C コンパイラが生成した C エラー・ファイルは，File メニューの “ Open…”コマンドでオープンすることができます。C エラー・ファイルの情報は，Message ウインドウにロードされ，C ソース・ファイルのエラーを調べることができます。ソース・ファイルのエラーを見付けるには，エラーが表示されている行をダブルクリックします。C ソース・ファイルがオープンされ，エラーが想定される行が強調表示されます。

1.1.5 C とアセンブラのディバグ情報

C ソース・レベルのディバグ情報を出力リンク・ファイルに提供するには，次のようにします。

(1) ディバグ情報を含めて C ソース・ファイルを InterTools 77016 C コンパイラでコンパイルします。詳細については， InterTools 77016 C コンパイラ　マニュアルを参照してください。
(2) ファイルを WB77016 プロジェクトに追加します。
(3) Assembler Settings ダイアログで，“ Include Debug Information”をチェックします。
(4) Linker Settings ダイアログで，“ Include Debug Information”をチェックします。
(5) プロジェクトを構築します。

これらの処理で，HSM77016 または，ID77016 の C ソース・レベルのディバグ機能を使用して，出力リンク・ファイルを設定することができます。

注意 C ソース・レベルまたはアセンブラのディバグ情報は，エラー発生元に応じて，自動的にプロジェクト中のファイルに割り当てられます。

保守／廃止

1.1.6 アセンブラのディバグ・ディレクティブ

ディバグ・ディレクティブが WB77016 のアセンブラ・コマンドに追加され，InterTools 77016 C コンパイラと WB77016 との間で情報を共有できます。

ディバグ・ディレクティブは InterTools 77016 C コンパイラにより自動的に生成され，アセンブラ・ソースには C ソース・ディバグ情報が挿入されます。次にそのディレクティブ情報を示します。

注意　これらのディレクティブは，ユーザが変更するものではありません。

名　称	機　能
.file ディレクティブ	現在のソース・ファイルのシンボルを定義します。
.line ディレクティブ	C ソース文の行番号を識別します。
.sym ディレクティブ	グローバル変数やローカル変数，関数，型定義，セグメント名，HSM77016 や ID77016 が把握している必要があるほかの有用な情報を定義します。いくつかのパラメータを使うと，属性を各シンボルに付与することができます。
.func ディレクティブ	C 関数の開始行を指定します。
.endfunc ディレクティブ	C 関数の終了行を指定します。
.block ディレクティブ	C ブロックの境界を指定します。
.endblock ディレクティブ	C ブロックの境界を指定します。
.stag ディレクティブ	構造体の始まりを示します。
.etag ディレクティブ	列挙体の始まりを示します。
.utag ディレクティブ	共用体の始まりを示します。
.member ディレクティブ	構造体，列挙体，共用体のメンバを指定します。
.eos ディレクティブ	構造体，列挙体，共用体を終了します。
.prolog ディレクティブ	関数 prolog 中の命令数を指定します。prolog を実行すると，関数 stack-frame が対応可能になり，ID77016 でローカル変数を監視できるようになります。

1.1.7 Pragma

InterTools 77016 C コンパイラが生成した次の pragma は，アセンブラとリンカにより処理されます。

PRAGMA "my_calls_known"

この pragma は，ほかのコード・セクションへの呼び出しを含まないか，直接呼び出ししか含まない（つまり，ポインタ経由の呼び出しを含まない）関数のコード・セクションに対して発行されます。

PRAGMA "I_call" *<name_of_callee>*

この pragma は，既知の呼び出しを含むコード・セクションからのすべての呼び出しに対して発行されます。

★ 1.2 オーダ名称と対象 OS

ホスト・マシン	対象 OS	オーダ名称（媒体）
PC-9800 シリーズ	Windows 95, Windows 98, および	μSAA17WB77016（CD-ROM）
IBM PC/AT™	Windows NT 4.0	

保守／廃止

1.3 WB77016 パッケージ内容

WB77016 のパッケージには次のものが含まれます

1.3.1 ツール製品

- WB77016　　Windows アプリケーション
- μPD77016 Librarian　Windows アプリケーション
- Model Editor　　Windows アプリケーション

1.3.2 日本語ドキュメント

- μPD77016 オンライン・マニュアル
- モデル・エディタ・ヘルプ
- WB77016 Tips
- μPD7701x Programming Tips

★ 1.4 対象デバイス

WB77016 の対象デバイスは次のとおりです。

- μPD77015，77016，77017，77018，77018A，77019
- μPD77110，77111，77112，77113，77114

注意　このドキュメントにμPD77116，NA77113 についての記述がありますが，現在の WB77016 **は**μPD77116，NA77113 **には正式には対応していません。**

1.5 表記法

このドキュメントで使用されている用語を表 1－1に示します。

表 1－1　表記法

用　語	説　明
モノスペース	モノスペース・テキストは，プログラム例やプログラム出力に使用されます。
太字	太字は，文字どおりに使用すべき固有の用語を示します。これらの用語は，表示のとおり正確に入力する必要があります。ただし，大文字と小文字の区別は必要ありません。
斜体	斜体文字を使用したテキストは，仮の表現です。ユーザは実際に入力する値に置き換える必要があります。
カギかっこ［…］	カギかっこによって任意のフィールドやパラメータが表現されます。
垂直バー ｜	垂直バーは，バーの両側に表されたエントリの 1 つを入力することを要求します。
Windows	Microsoft Windows 95，Windows 98，Windows NT4.0 オペレーティング・システム
μPD77116	命令キャッシュ，および命令とデータの DMA を持ったディジタル・シグナル・プロセッサ
DSP	ディジタル・シグナル・プロセッサμPD77016 ファミリ
IE-77016-98/PC	インサーキット・エミュレータ（IE-77016-PC または IE-77016-98）

第 2 章　セットアップ

WB77016 アプリケーションには，WB77016 をユーザ・システムにインストールするのに必要なすべてのタスクを実行するセットアップ・ウィザードが添付されています。WB77016 をインストールするには，4.5MB 以上の空きハード・ディスク容量が必要です。

2.1　WB77016 のセットアップ

ここでは，WB77016 単体をインストールする場合について説明します。

注意　SP77016 をご購入のお客様は 2.2　SP77016 のセットアップをお読みください（SP77016 は，WB77016, HSM77016, ID77016 がパッケージされた商品です）。WB77016 単体のセットアップ手順とは異なります。

2.1.1　WB77016 をインストールする前に確認しておくこと

ホスト・マシンに旧バージョンの WB77016 がインストールされている場合は，次の点を確認してください。

・Windows￥system ディレクトリに，MODELDLG.DLL ファイルが残っていないかどうか確認してください。残っている場合は，この MODELDLG.DLL を使用するアプリケーションが必要であれば，アプリケーションが置かれたフォルダへファイルを移動し，不要であれば削除してください。

備考　このファイルは，スタータ・キットに添付されている旧バージョンの WB77016 からインストールされたファイルです。system ディレクトリに MODELDLG.DLL ファイルが残っていると，WB77016 でプロセッサ・モデルを指定する際に，エラーが発生したり，または旧バージョンの MODELDLG.DLL が起動されてしまい，正しい動作ができなくなる場合があります。

・旧バージョンの WB77016 と同じフォルダにバージョンアップしたい場合は，旧バージョンをアンインストール（SP77016 の場合：アンインストーラによりファイルを削除，WB77016 の場合：手動でファイルを削除）してから，新バージョンをインストールすることを推奨します。

2.1.2　WB77016 のインストール手順

インストール手順を示します。

（1）OS を起動します。

（2）CD-ROM ドライブに WB77016（CD-ROM）をセットします。

（3）タスク・バーから「ファイル名を指定して実行...」を選択して，WB77016￥disk1￥Setup.exe を選択します。

（4）インストール画面が表示されます。

セットアップは，ハードディスクにすべてのファイルをコピーし，タスク・バーのプログラム・メニューに新たに WB77016 に関するメニューを追加します。

保守／廃止

＜補足＞

¥Japanese ディレクトリに次の日本語ドキュメント・ファイルがあります。WB77016 をインストールしたディレクトリに上書きすることで，各 DSP ツールから日本語の On-Line-HELP が使用できます。

・uPD77016.hlp および uPD77016.cnt：μPD7701x 日本語 On-Line-Manual

・Mdldlg32.hlp および Mdldlg32.cnt：モデル・エディタ日本語 HELP

注意　オリジナルの HELP ファイルを残したい場合は，あらかじめ別ディレクトリに退避またはリネームなどの処理を行っておいてください。
その他のファイルは，任意のディレクトリにコピーするなどしてご利用ください。

2.1.3　WB77016 のアンインストール手順

次にアンインストール手順を示します。

(1) エクスプローラから，WB77016 をインストールしたディレクトリとそのディレクトリ下にあるファイルを削除します。

(2) プログラム・メニューから，WB77016 をインストールしたファイルのフォルダとショートカット一式を削除します。

1. [スタート]ボタンをクリックし，[設定]をポイントします。
2. [タスク・バー]をクリックし，[[スタート]メニューの設定]タブをクリックします。
3. [削除]をクリックし，WB77016 をインストールしたフォルダとショートカットを探します。
4. ショートカットをクリックし，[削除]をクリックします。

★　2.2　SP77016 のセットアップ

次の手順によりセットアップを行ってください。

備考　旧バージョンと同じフォルダにバージョンアップを行う場合は，旧バージョンをアンインストーラにより，アンインストールしてから，新バージョンをインストールすることを推奨します。旧バージョンをアンインストールしたくない場合は，別のフォルダへ新バージョンをインストールすることを推奨します。

2.2.1　SP77016 のインストール手順

(1) OS を起動してください。

(2) SP77016（CD-ROM）を CD-ROM ドライブにセットします。

(3) エクスプローラなどから，CD-ROM ドライブの “ATAIR”ディレクトリにある“setup.exe”をダブルクリックしてインストール・プログラムを起動します。

(4) インストール・プログラムの指示に従ってください。

(5) ID77016 をご使用になる場合は，さらにデバイス・ドライバの組み込みが必要です（2.2.3，2.2.4 参照）。
IE-77016-98/PC が，ホスト・マシンに正しく設置されている必要があります。

2.2.2　日本語ドキュメントのインストール手順

エクスプローラなどで直接ファイルをコピーするなどしてご利用ください。

日本語ドキュメントは CD-ROM ドライブの“Japanese￥Atair”ディレクトリに用意されています。

次のファイルは，SP77016 をインストールしたディレクトリ（ディフォールトは￥Program Files¥Atair Dsp Tools）に上書きすることで，各 DSP ツールから日本語の On-Line-HELP を使用できます。

- uPD77016.hlp および uPD77016.cnt ：μPD7701x 日本語 On-Line-Manual
- Mdldlg32.hlp および Mdldlg32.cnt ：モデル・エディタ日本語 HELP

注意　オリジナルの HELP ファイルを残したい場合は，あらかじめ別ディレクトリに退避またはリネームなどの処理を行っておいてください。
その他のファイルは，任意のディレクトリにコピーするなどしてご利用ください。

2.2.3　デバイス・ドライバ組み込み手順（Windows 95/98 共通）

SP77016 をインストールした際に，ID77016 もインストールした場合は，IE-77016-98/PC デバイス・ドライバも組み込む必要があります。

デバイス・ドライバを組み込むときは，IE-77016-98/PC がホスト・マシンに正しく設置されているか確認してください。詳細は，IE-77016-98，IE-77016-PC **ユーザーズ・マニュアル　ハードウエア編**を参照してください。

（1）デバイス・ドライバの組み込み

デバイス・ドライバの組み込み手順を次に示します。設定変更については，（2）**デバイス・ドライバの設定変更**を参照してください。

1. IE-77016-98/PC をホスト・マシンに接続します。
2. Windows 95/98 を起動します。
3. CD-ROM ドライブに CD-ROM を挿入します。ここでは，CD-ROM ドライブを Q:ドライブと仮定してを説明します。
4. [スタート]ボタン ― [設定] ― [コントロールパネル]からコントロールパネルを開きます。
5. コントロールパネルから[ハードウェア]を選択し，ハードウエア ウィザードを起動します。
6. ハードウエアの自動検出で[いいえ]を選択します。
7. [ハードウエアの種類(H):]で[その他のデバイス]を選択します。
8. [ディスク使用(H)…]ボタンをクリックし，IE-77016.INF ファイルの位置として“Q¥ATAIR”ディレクトリを指定します。
9. デバイス・ドライバのインストール後，JTAG クロック周波数，テスト・アクセス・ポートおよびボード I/O アドレスを設定しなければなりませんので，コンピュータの再スタートが要求されても，ここでは[いいえ]を選択します。
10. デバイス・ドライバの設定を変更します（（2）を参照）。

（2）デバイス・ドライバの設定変更

1. **コントロールパネルから**[**システム**]**を選択し，**[**デバイス　マネージャ**]**タブを表示します。**
2. **リストの**“Test Access Port Adapter”**より**“IE-77016-PC EM1 Evaluation Board”**を選択します。**
3. [**プロパティ**(R)]**ボタンをクリックします。**
4. JTAG **クロックの変更**

   [IE-77016-PC EM1 Evaluation Board **のプロパティ**]**ダイアログで**[Settings]**タブを選択します。クロック周波数は，**5**，**2.5**，**1.25MHz **および** 625kHz **から選択できます（ディフォールトは** 5MHz **です）。非常に長いケーブルが使われるか，**ID77016 **とのリンクが信頼できないようならば，低速のクロック周波数を選択します。**
5. **内部および外部リンクの属性の変更**

   [IE-77016-PC EM1 Evaluation Board **のプロパティ**]**ダイアログで**[Settings]**タブを選択します。内部リンク（**IE-77016-CM-EM6 **とのリンク）および外部リンク（ユーザ・ターゲットとのリンク）のリンク名とテスト・アクセス・ポートの設定を行います。各リンク名はユーザが入力しなければなりません（これらのリンク名は，**ID77016 **の起動時**[Target System Model]**ダイアログにおいて表示されます）。各リンクに割り当てられるテスト・アクセス・ポートを選びます（内部および外部へのリンクは，選ばれたテスト・アクセス・ポートを経てアクセスされます）。**
6. I/O **アドレスの変更**

   [IE-77016-PC EM1 Evaluation Board **のプロパティ**]**ダイアログで**[**リソース**]**タブを選択します。**[**設定の変更**(c)…]**ボタンをクリックし，**IE-77016-98/PC **のベース・アドレスを設定します。**I/O **アドレス値には次のアドレスが使用できます。**

   IE-77016-98：0x0d0 **（ディフォールト）**, 0x1d0, 0x2d0 **など**

   IE-77016-PC：0x300 **（ディフォールト）**, 0x320 **など**
7. **コンピュータの再起動を行います。**

2.2.4　デバイス・ドライバ組み込み手順（Windows NT4.0）

ID77016 **とデバイス･ドライバ･ソフトウエアをインストールする前に，**IE-77016-98/PC**（以下** IE77016 Control Board**）を設置することを推奨します（詳細は，**IE-77016-98, IE-77016-PC **ユーザーズ・マニュアル　ハードウエア編を参照してください）。**

Windows NT4.0 **の環境下では，デバイス・ドライバのインストールは** ID77016 **セットアップ・プログラムにより実行されます。デバイス・ドライバの組み込みには管理者権限が必要です。**ID77016 **およびデバイス・ドライバのインストールが終了した後に，引き続きデバイス・ドライバのコンフィギュレーションが行われます。**

・手動で IE77016 Control Board **のコンフィギュレーションを開始するには**

（1）[**スタート**]**ボタン　―　**[**設定**]　―　[**コントロールパネル**]**によりコントロールパネルを開きます。**

（2）[IE77016 Control Board]**をダブル・クリックします。**

・IE77016 Control Board **のコンフィギュレーションを行うには**

IE77016 Control Board **デバイス・ドライバは複数の** Control Board **をサポートします。新しい** Control Board **を付加するには**[Add…]**ボタンを，設定を変更するにはボード名のリストから設定変更を希望する** Control Board **を選び，**[Settings…]**ボタンを，設定を削除するためには，ボード名のリストから，設定変更を希望する** Control Board **を選び**[Remove]**ボタンをクリックします。**

Board Name：

IE77016 Control Board に対しての通称を入力します。

Base I/O Address：

ボード・ハードウエア設定にしたがって IE77016 Control Board の I/O アドレスを設定します。ディフォールト・アドレスは 0x300 です。

Clock Rate：

クロック周波数は，5, 2.5, 1.25MHz および 625kHz から選択できます（ディフォールトは 5MHz です）。非常に長いケーブルが使われるか，ID77016 とのリンクが信頼できないようならば，低速のクロック周波数を選択します。

Internal link：

内部リンク（IE-77016-CM-EM6） のリンク名と，そのテスト・アクセス・ポート番号の設定を行います（これらのリンク名は，ID77016 の起動時 Target System Model ダイアログにおいて表示されます）。

External Link：

外部リンク（ユーザ・ターゲットとのリンク） のリンク名と，そのテスト・アクセス・ポート番号の設定を行います（これらのリンク名は，ID77016 の起動時 Target System Model ダイアログにおいて表示されます）。

・ドライバの再スタート

コンフィギュレーションの変更を行った場合，ドライバの再スタートが必要です。

（1）[スタート]ボタン ― [設定] ― [コントロールパネル]によりコントロールパネルを開きます。

（2）[デバイス]をダブル・クリックします。

（3）リストから“IE77016 Control Boards”を選択します。

（4） [停止]ボタンをクリックします。

（5） [開始]ボタンをクリックします。

［メ　　モ］

保守／廃止

第 3 章　ファイル・フォーマット

3.1　アセンブラ入力ファイル

3.1.1　アセンブラ・ソース・ファイル

WB77016 エディタ・ウインドウで編集されるほとんどのファイルは，通常アセンブラ・ソース・ファイルです。ソース・ファイルは，μPD77016 アセンブラ言語構文に従うソース・テキストを含むテキスト形式のファイルです。アセンブラ・ソース・ファイルのディフォールトの拡張子は，".ASM"です。

3.1.2　ヘッダ・ファイルとインクルード・ファイル

ソース・テキストでヘッダ・インクルード・ファイルを定義する際の詳細については，WB77016 **ユーザーズ・マニュアル　言語編**を参照してください。アセンブラ・ヘッダ・ファイルとインクルード・ファイルの拡張子は，それぞれ " .H"と " .INC"です。

3.1.3　バックアップ・ファイル

バックアップ・ファイルは，Options メニューの"Editor…"コマンドで Create Backup File オプションをチェックしていると，自動的に作成されます。WB77016 エディタのバックアップ・ファイルのディフォールトの拡張子は " .BAK"です。

注意　WB77016 エディタは主にアセンブラ・ソース・ファイル（純粋な ASCII ファイル）を編集するためのものですが，エディタ・ウインドウには任意のファイルを表示することができます（ただし，作業可能な形式で表示されるのは ASCII ファイルだけです）。エディタ出力は，どんな拡張子でも格納できる純粋な ASCII ファイルです。

3.2　アセンブラ出力ファイル

3.2.1　オブジェクト・モジュール・ファイル

オブジェクト・モジュール・ファイルには，WB77016 アセンブラによって生成されたコードが入っています。リロケータブル・オブジェクト・モジュール・ファイルは，μPD77016 ファミリの実行可能コードを生成する WB77016 リンカによって処理されるようになっています。オブジェクト・モジュール・ファイルの拡張子は，".REL"です。

保守／廃止

3.2.2　プリント・ファイル

アセンブラ・プリント・ファイルには次のリストを付加することができます。

・アセンブル・リスト:
ソース・コード（アセンブラ命令，データ割り当て文など）と生成されたオブジェクト・コードが表示されます。

・シンボル・リスト:
VALUE ATTRIBUTE，RTYP，および NAME などの各ユーザ定義シンボルの構成要素が表示されます。

・クロスレファレンス・リスト:
シンボル・リストで表示されるユーザ定義シンボルに加えて，シンボルが使用されている場所のアドレスが記述されます。

3.2.3　エラー・ファイル

エラー・ファイルは，エラーの種類，番号および行番号情報を含めてアセンブル中に発生したエラーを収集します。行番号情報は，エラーが発生したソース・コードの行を示します。

注意　アセンブラ出力は，Options メニューの“Assembler…”コマンドを選択することによって制御できます。アセンブラ出力ファイルの名前は，アセンブラ・ソース・ファイルの名前に基づいて自動的に設定されます。たとえば，アセンブラ・ソース・ファイル名が“file01.asm”ならば，対応するオブジェクト・モジュール・ファイル名は，“file01.rel”になります。

★ 3.3　C エラー・ファイル

File メニューの“Open…”コマンドを実行すると，Message ウインドウにロードする C エラー・ファイルを選択する標準のファイル選択ダイアログが表示されます。C エラー・ファイルのフォーマットは，WB77016 の Message ウインドウで正しくリストされることを保証するために次のようになっている必要があります。

:*XX*:*name.c*:*nnn*:*message*

XX：　エラーを検出したコンパイラ・フェーズを識別します。
name.c：　エラーが発生したソース・ファイルの名前です。
nnn：　エラーが発生したファイル name.c の行番号です。

保守／廃止

3.4　HEX コンバータ・ファイル

HEX コンバータ・ファイル・タイプは，次の 2 つのカテゴリに分類されます。

・HEX フォーマット・ファイル

拡張子".HDX"，".HDY"，".HXI"，".HBI"，".HBE"，".MSK"

・バイト・ファイル

拡張子".XHD"，".XB0"，".XB1"，".YHD"，".YB0"，".YB1"，".IH0"，".IH1"，".IB0"，".IB1"，".IB2"，".IB3"

注意　HEX コンバータ出力は，Options メニューの Hexconversion…コマンドを選択することによって制御できます。

表 3－1 に，使用可能な HEX コンバータ出力ファイル・タイプのリストを示します。

表 3－1　HEX コンバータ出力ファイル・タイプ

ファイル拡張子	マップ・バイト	メモリ・タイプ	アドレシング・モード	データ例
.HDX	1,0	INT+EXT	バイト	3412
.XHD	1,0	EXT	ワード	3412
.XB0	1	EXT	ワード	34
.XB1	0	EXT	ワード	12
.HDY	1,0	INT＋EXT	バイト	3412
.YHD	1,0	EXT	ワード	3412
.YB0	1	EXT	ワード	34
.YB1	0	EXT	ワード	12
.HXI	3,2,1,0	INT+EXT	バイト	34120138
.IH0	3,2	EXT	ワード	3412
.IH1	1,0	EXT	ワード	0138
.IB0	3	EXT	ワード	34
.IB1	2	EXT	ワード	12
.IB2	1	EXT	ワード	01
.IB3	0	EXT	ワード	38
.HBI	3,2,1,0	IRAM INT	バイト	34120138
.HBE	3,2,1,0	IRAM EXT	バイト	34120138
.MSK	(3,2,)1,0 (3,2,)1,0 (3,2,)1,0	IROM INT XROM INT YROM INT	バイト バイト バイト	3412(0138) 3412(0138) 3412(0138)

保守／廃止

マップ・バイト

配列では，命令ワードが 0，1，2，3 の順で 4 バイトから成り，データ・ワードが 0，1，の順で 2 バイトから成り立っていると想定しています。

メモリ・タイプ

メモリ・タイプの詳細については，6.7.16　Hexconversion…を参照してください。

バイト・アドレス

メモリ内の各バイトのアドレスが次の範囲内にあることを意味しています。

・インストラクション・メモリ:　0-216143

・データ・メモリ:　0-131071

ファイル".HDX"，".HDY"，".HXI"のバイト・アドレスは，65535 の制限を越えることができます。アドレスのオフセットがタイプ 2 の特殊レコードにコーディングされているのは，このためです。バイト・アドレスのオフセットは，次のように計算されます。

Offs = (Addr%65536)*4096

(in C language "wOffs=(lAddr>>4)&0xf000;")

ワード・アドレス

インストラクション・メモリおよびデータ・メモリの両方について，0-65535 までの範囲内ワードの物理アドレスを記述します。

データ例

インストラクション・メモリについてはインストラクション"r01=0x1234"（opcode 0x38011234）を，データ・メモリについては擬似インストラクション"dw0x1234"の割り当てを提供します。

3.4.1　16 進オブジェクト・ファイル・フォーマット

バイナリ形式のファイルは，16 進オブジェクト形式を使用してテキスト・ファイルとして表示できます。このファイル・フォーマットを使用すると，特別な通信ソフトウエアを使用しなくても（モデムによって）コンピュータ間でファイル転送を簡単に行えます。16 進オブジェクト・ファイル・フォーマットは，インストラクション・メモリおよびデータ ROM メモリ内容を 1 組の ASCII テキスト・レコードとして表現するものです。

ファイル・フォーマットは，次の 3 種類の ASCII テキスト・レコード・タイプから成り立っています。

・データ・レコード（データ・レコード・フォーマットは表 3－2 に示します）

・ EOF レコード

・アドレス・オフセット・レコード

表3−2　データ・レコードの構成

1	2	3	4	5	6	7	8
:	XX	XXXX	XX	XX..	XX	CR	LF

1. レコード・マーク（1 バイト）

   レコードの始まりを示します。“ : ”に固定です。

2. データ長（1 バイト）

   このレコードのデータ項目中のバイト数を示します。

3. ロケーション・アドレス（2 バイト）

   このレコード中の最初のデータ・アドレスを示します。

4. レコード・タイプ（1 バイト）

   0…データ・レコード

   1…EOF

   2…アドレス・オフセット

5. データ（0-16 バイト）

6. チェックサム（1 バイト）

   このレコードの項目 2 から 5 までのすべてのバイトを初期値 00 から順番に引き算していった演算結果です。

7. CR（改行）

8. LF（行送り）

注意　レコードの先頭と最後の間にスペースはありません。

3.5 リンカ出力ファイル

3.5.1 ロード・モジュール・ファイルまたはリンク・ファイル

ロード・モジュール・ファイルには，リンカが生成したオブジェクト・コードが入ります。リンク・ファイルは，μ PD77016 実行可能ファイルです。これらのファイルは，HSM77016 へのインタフェースを作成します。新しいシミュレーション・セッションは，リンク・ファイルをインポートすることによって常に開始されます。WB77016 から HSM77016 を起動する際の詳細については，6.5.5 Simulate を参照してください。リンク・ファイルの拡張子は，“.LNK”です。

保守／廃止

3.5.2　マップ・ファイル

マップ・ファイルは，次のリストから構成されます。

・ セグメント情報リスト

このリストには，リンク・マップ・リスト（ファイル番号，ファイル名，モジュール名が入ります）とセグメント・マップ・リスト（各セグメントの開始アドレスと終了アドレス，サイズ，名前，モジュール番号が入ります）が入っています。セグメント・マップ・リストは，メモリ領域の順で整列されます。

・パブリック・シンボル・リスト

このリストには，シンボル名，モジュール番号（シンボルが属しているモジュールの番号），メモリ・セグメント（IM，XRAM，YRAM，XROM，YROM），シンボル値（シンボルがレーベルの場合，値はレーベル・アドレスを含みます。-EQU ディレクティブの場合は，レーベルに定義されている値が入ります）が入っています。

・ローカル・シンボル・リスト

このリストには，シンボル名，シンボル・タイプ（パブリック・シンボルは“pub”，EQU ディレクティブは“con”，外部シンボルとしてインポートされたシンボルは“ext”，ローカル・シンボルは“sym”），シンボル値が入っています。

・プリロード・モジュール・クラスとそのメンバは，リンカが生成するマップ・ファイルにリストされています。2 つのセグメント“i1”と“i2”から成るプリロード・モジュール“preload1”のリスト例を次に示します。

Preload class 'preload1':

MODULE	SEGMENT	START(BNK/BLK)	END(BNK/BLK)	SIZE
M1	i1	0x8000(0/0)	0x8000(0/0)	0x1
M2	i2	0x8005(0/0)	0x8006(0/0)	0x2

3.5.3　エラー・ファイル

エラー・ファイルは，エラーの種類，番号および行番号情報などを含めてリンク中に発生したエラーを収集します。行番号情報は，エラーが発生したソース・コードの行を示します。

注意　リンカ出力は，Options メニューの“Linker|Output…”コマンドを選択することによって制御できます。リンカ出力ファイルの名前は，プロジェクト・ファイルの名前に基づいて自動的に設定されます。たとえば，プロジェクト・ファイル名が“MAIN.PRJ”ならば，対応するロード・モジュール・ファイル名は“MAIN.LNK”になります。

保守／廃止

3.6　プロジェクト・ファイル

プロジェクト・ファイルには，ロード・モジュール・ファイルを構築するためのすべての情報が入ります。プロジェクト・ファイルは，Project メニューの各コマンドを使って，作成，オープン，クローズ，修正などが行われます。プロジェクト・ファイルの拡張子は，".PRJ"です。

注意　プロジェクト・ファイルか，プロジェクトに含まれるファイルのどちらかを別のディレクトリに移動しても，プロジェクト設定には影響がありません。ファイルをプロジェクトに復元するには，プロジェクト・ウインドウからファイルを削除し，再びそれらを追加してください。

★ 3.7　ASCII プロジェクト・ファイル出力

プロジェクト・ファイルは，WB77016 固有のバイナリ・ファイル形式で格納されます。プロジェクト・ファイルの情報をプロジェクト開発ドキュメンテーションで利用できるようにするために，プロジェクト・ファイルを ASCII テキスト形式で格納することができます。

プロジェクト・ファイルを ASCII テキスト形式で保存するには，Project メニューから"Save Project As…"コマンドを実行します。"Save Project As…"コマンドによって表示されるファイル選択ダイアログには，現在のプロジェクト設定をテキスト形式で格納するファイル・フィルタ（*.txt）があります。

Project メニューの" Open Project…"，"Save Project"，"Close Project"コマンドは，バイナリ形式のプロジェクト・ファイルに関係するものです。

3.7.1　ASCII プロジェクト・ファイル・フォーマット

テキスト形式のプロジェクト・ファイルは，それぞれプロジェクト開発の追跡，プロジェクト設定の再構築を目的としたものです。ASCII プロジェクト・ファイルには，いくつかのセクションがあります。

各セクションは，[　] で囲まれたセクション・タイトル" [*sectiontitle*] "で始まります。各セクションには，"*itemname=itemdata*"という形式のセクション固有の項目名で始まるたくさんのデータ項目があります。各行には，1 つのデータ・エントリしか入れることはできません。最初のカラムにセミコロン" ;"がある行は，コメントとして扱われます。

次に，ASCII 形式のプロジェクト・ファイル・セクションについて説明します。

(1) ページ・ヘッダ

ページ・ヘッダは，プロジェクト・ファイルの先頭にコメントとして挿入されます。この中には，プロジェクト出力を作成したツールの名前とバージョン，ASCII プロジェクト・ファイルが作成された日付けと時間が入ります。

例

```
;*******************************************
;*µPD77016 Workbench 2.31 - Project Dump      *
;*            Fri Mar 27 08:54:08 1998         *
;*******************************************
```

保守／廃止

(2) FileList セクション

このセクションでは，プロジェクト中のファイルのパスと名前，各ファイルに適用されるアセンブラ設定がリストされます。このセクションには，ソース・ファイルの数に応じていくつかのブロックを入れることができます（各ソース・ファイルに 1 つのブロックが割り当てられます）。このセクションのリストは，WB77016 の Project ウインドウのファイル・リストと Assembler Settings ダイアログの設定に対応します。

(a) セクション・ヘッダ

[FileList]

(b) セクション・エントリ

File*n* = *filename*

n は，1 から始まる連続番号です。*filename* は，ソース・ファイルのパスと名前です。パス宣言は，対応するファイルに指定された make オプションに応じて絶対パスであるか，相対パスであるかのどちらかです。

AssembleListFile=Yes｜**No**

アセンブル・リストのファイル出力を指定します。

AssembleListDisplay=Yes｜**No**

アセンブル・リストのディスプレイ出力を指定します。

SymbolListFile=Yes｜**No**

シンボル・リストのファイル出力を指定します。

SymbolListDisplay=Yes｜**No**

シンボル・リストのディスプレイ出力を指定します。

X-referenceListFile=Yes｜**No**

クロスレファレンス・リストのファイル出力を指定します。

X-referenceListDisplay=Yes｜**No**

クロスレファレンス・リストのディスプレイ出力を指定します。

CreateErrorFile=Yes｜**No**

エラーのファイル出力を指定します。

IncludeDebugInformation=Yes｜**No**

ディバグ情報を出力ファイルの中に入れます。

RelativePath=Yes｜**No**

このソース・ファイルの相対パス設定を指定します。

IncludeFile*n* = *includefilepath*

n は，1 から始まる連続番号です。*includefilepath* は，ソース・ファイルに含まれるファイルのパスと名前です。

(c) 例

```
[FileList]
File1=H:¥ASMFILES¥GSM¥GSMCODEC.ASM
AssembleListFile=Yes
AssembleListDisplay=Yes
SymbolListFile=No
SymbolListDisplay=No
X-referenceFile=Yes
X-referenceListDisplay=No
CreateErrorFile=No
IncludeDebugInformation=Yes
RelativePath=No
IncludeFile1=H:¥ASMFILES¥GSM¥INCLUDE1¥ABC.INC
IncludeFile2=H:¥ASMFILES¥GSM¥INCLUDE1¥DEF.INC
```

(3) ProcessorModel セクション

このセクションには，プロジェクトに対して選択されたプロセッサ・モデルが入ります。このセクションのリストは，選択されたプロセッサ・モデルから引き出されます。

(a) セクション・ヘッダ

[ProcessorModel]

(b) セクション・エントリ

ModelFile=*modelpath*

modelpath **は，モデルまたはモデル定義ファイルのパスと名前です。**

System=*systemname*

systemname **は，評価システムの名前です。**

Processor=*processortype*

processortype **は，**μ PD77016 **ファミリのプロセッサ・タイプです。**

(c) 例

```
[ProcessorModel]
ModelFile=H:¥ASMFILES¥GSM¥UPD77016.INI
System=µPD77016 Evaluation Board EM6
Processor=µPD77016
```

保守／廃止

(4) MiscellaneousOptions セクション

このセクションには，アセンブラ・プリンタ出力ファイルおよびリンカ・プリンタ出力ファイルのページ長とページ幅の設定とアセンブル終了設定およびリンク終了設定が入ります。このセクションのリストは，WB77016 の Miscellaneous ダイアログから引き出されます。

(a) セクション・ヘッダ

[MiscellaneousOptions]

(b) セクション・エントリ

ErrorCount=*number*

WarningCount=*number*

PageWidth=*number*

PageLength=*number*

number は，次の範囲の正の整数です。

ErrorCount，WarningCount:　1-100

PageLength:　0，10-999（0=行はカウントされない）

PageWidth:　60-132

(c) 例

```
[MiscellaneousOptions]
ErrorCount=10
WarningCount=10
PageWidth=80
PageLength=30
```

(5) Directories セクション

このセクションには，WB77016 の Directories ダイアログで作成されたディレクトリ設定が入ります。

(a) セクション・ヘッダ

[Directories]

(b) セクション・エントリ

ProjectPath=*projectpath*

projectpath は，現在のプロジェクト・ファイルのパスと名前です。

OutputPath=*outputpath*

outputpath は，出力リンク・ファイルが格納されるパスです。

IncludePath*n* **=** *includepath*

n は，1 から始まる連続番号です。

includepath は，インクルード・ファイル・ロケーションへのパスです。

(c) 例

```
[Directories]
ProjectPath=H:¥ASMFILES¥GSM¥GSM.PRJ
OutputPath=H:¥ASMFILES¥GSM
IncludePath1=H:¥ASMFILES¥GSM¥INCLUDE1
IncludePath2=H:¥ASMFILES¥GSM¥INCLUDE1
```

(6) LinkerSettings セクション

このセクションには，WB77016 の Linker Settings ダイアログで作成された設定が入ります。これらの設定は，マップ・ファイル生成，ディバグ情報，エラー・ファイル生成，警告出力から成り立っています。

(a) セクション・ヘッダ

[LinkerSettings]

(b) セクション・エントリ

SegmentInfoFile=Yes｜No

セグメント情報のファイル出力を指定します。

SegmentInfoDisplay=Yes｜No

セグメント情報のディスプレイ出力を指定します。

PublicSymbolsFile=Yes｜No

パブリック・シンボル情報のファイル出力を指定します。

PublicSymbolsDisplay=Yes｜No

パブリック・シンボル情報のディスプレイ出力を指定します。

LocalSymbolsFile=Yes｜No

ローカル・シンボル情報のファイル出力を指定します。

LocalSymbolsDisplay=Yes｜No

ローカル・シンボル情報のディスプレイ出力を指定します。

CreateErrorFile=Yes｜No

エラーのファイル出力を指定します。

IncludeDebugInformation=Yes｜No

ディバグ情報を出力ファイルに入れます。

CombineSegments=Yes｜No

セグメントを共通属性と組み合わせます。

CCodeAnalysis=Yes｜No

C **関数呼び出しのネスティングを分析します。**

CStaticStackOverlay=Yes｜No

静的スタック・セグメントを分析し，重複割り当てを引き起こします。

IgnoreMagicNumber=Yes｜No

ターゲット DSP **タイプをエンコーディングするマジック番号を無視します。**

WarningL015=Yes｜No

静的スタック・オーバレイ・モードでのリンク中に重複が検出された場合，警告メッセージを出力します。

WarningL017=Yes｜No

静的スタック・オーバレイ・モードでのリンク中にレジスタ呼び出しが検出された場合，警告メッセージを出力します。

WarningL024=Yes｜No

モジュール内のジャンプまたは呼び出しが検出された場合，警告メッセージを出力します。

SegmentOrderFile=*filename*

filename **は，セグメント順序ファイルのパスと名前です。**

(c**) 例**

```
[LinkerSettings]
SegmentInfoToDisk=No
SegmentInfoDisplay=No
PublicSymbolsToDisk=Yes
PublicSymbolsDisplay=No
LocalSymbolsToDisk=Yes
LocalSymbolsDisplay=No
CreateErrorFile=No
IncludeDebugInformation=Yes
CombineSegments=Yes
CCodeAnalyses=No
CStaticStackOverlay=No
IgnoreMagicNumber=No
WarningOverlappedArea=No
WarningRegisterCall=No
WarningSegmentJumpCall=No
```

(7) HexConverterSettings **セクション**

このセクションには，WB77016 **の** Hex Conversion **ダイアログで作成された設定が入ります。これらの設定は，**HEX **コンバータ出力ファイル・タイプから成り立っています**

(a**) セクション・ヘッダ**

[HexConverterSettings]

(b**) セクション・エントリ**

HxiFile=Yes｜No

インストラクション・メモリからの HEX **フォーマット・ファイル出力データ変換**

IhFile=Yes｜No

インストラクション・メモリからバイト・フォーマット（16 **ビット）ファイル出力データ変換**

IbFile=Yes｜No

インストラクション・メモリからのバイト・フォーマット（8 **ビット）ファイル出力データ変換**

HbiFile=Yes｜No

起動用内蔵 RAM **からの** HEX **フォーマット・ファイル出力データ変換**

保守／廃止

HdxFile=Yes｜**No**

X **データ・メモリからの** HEX **フォーマット・ファイル出力データ変換**

XhdFile=Yes｜**No**

X **データ・メモリからのバイト・フォーマット（**16 **ビット）ファイル出力データ変換**

XbFile=Yes｜**No**

X **データ・メモリからのバイト・フォーマット（**8 **ビット）ファイル出力データ変換**

HdyFile=Yes｜**No**

Y **データ・メモリからの** HEX **フォーマット・ファイル出力データ変換**

YhFile=Yes｜**No**

Y **データ・メモリからのバイト・フォーマット（**16 **ビット）ファイル出力データ変換**

YbFile=Yes｜**No**

Y **データ・メモリからのバイト・フォーマット（**8 **ビット）ファイル出力データ変換**

XDmaHex=Yes｜**No**

X-DMA **メモリからの** HEX **フォーマット・ファイル出力データ変換**

XDmaWord=Yes｜**No**

X-DMA **メモリからのワード・フォーマット（**16 **ビット）ファイル出力データ変換**

XDmaByte=Yes｜**No**

X-DMA **メモリからのバイト・フォーマット（**8 **ビット）ファイル出力データ変換**

YDmaHex=Yes｜**No**

Y-DMA **メモリからの** HEX **フォーマット・ファイル出力データ変換**

YDmaWord=Yes｜**No**

Y-DMA **メモリからのワード・フォーマット（**16 **ビット）ファイル出力データ変換**

YDmaByte=Yes｜**No**

Y-DMA **メモリからのバイト・フォーマット（**8 **ビット）ファイル出力データ変換**

MskFile=Yes｜**No**

HEX **出力ファイルには，内蔵命令** ROM **領域，**X **データ** ROM **領域，**Y **データ** ROM **領域のコピーが入ります。**

MaskClockSelection=1/1｜**1/2**｜**1/4**｜**1/8**｜**1/3**

PLL **回路クロック分周器を指定します（**MskFile **が許可されている場合のみ）。**

MaskClockOutput=Yes｜**No**

CLKOUT **端子を許可または禁止します（**MskFile **が許可されている場合のみ）。**

MaskClockRate=x1-x16

1-16 **のクロック・レートの係数を指定します（**MskFile **が許可されていて，プロセッサ・タイプが** μ PD77111 **ファミリの場合のみ）。**

MaskClockDivideRate=1/1-1/16

1/1-1/16 **のクロック・レートの係数を指定します（**MskFile **が許可されていて，プロセッサ・タイプが**μ PD77111 **ファミリの場合のみ）。**

MaskHaltDivideRate=1/1-1/16

1/1-1/16 **のクロック分周レートを指定します（**MskFile **が許可されていて，プロセッサ・タイプが** μ PD77111 **ファミリの場合のみ）。**

MaskClockOutput=Yes｜**No**

クロック出力を許可または禁止します（MskFile **が許可されていて，プロセッサ・タイプが**μ PD77111 **ファミリの場合のみ）。**

MaskClockWakeUp=Yes｜No

ウェイクアップ関数を許可または禁止します（MskFile **が許可されていて，プロセッサ・タイプが** μ PD77111 **ファミリの場合のみ）。**

NA77113=Yes｜No

NA77113 **の追加マスク設定を許可します（**MskFile **が許可されていて，プロセッサ・タイプが** μ PD77111 **ファミリの場合のみ）。**

PllBypassMode=Yes｜No

PLL **バイパス・モードを許可または禁止します（**MskFile **が許可されていて，プロセッサ・タイプが** μ PD77111 **ファミリの場合のみ）。**

ClockRateOffsetMode=Yes｜No

クロック・レート・オフセット・モードを許可または禁止します（MskFile **が許可されていて，プロセッサ・タイプが**μ PD77111 **ファミリの場合のみ）。**

（c）例

```
[HexConverterSettings]
HxiFile=No
IhFile=No
IbFile=No
HbiFile=No
HdxFile=No
XhdFile=No
XbFile=No
HdyFile=No
YhFile=No
YbFile=No
XDmaHex=No
XDmaWord=No
XDmaByte=No
YDmaHex=No
YDmaWord=No
YDmaByte=No
MskFile=No
```

（8）MakeOptions セクション

このセクションは，WB77016 **の** Make Settings **ダイアログで行われた設定から成り立っています。これらの設定は，構築プロセス自身とプロジェクト内のファイルに適用される構築オプションから成り立っています。**

（a）セクション・ヘッダ

[MakeOptions]

保守／廃止

(b) セクション・エントリ

StopOn=Warnings｜**Errors**｜**Fatal Errors**｜**All sources processed**

アセンブルやリンクを停止する条件を指定します。

AfterAssembling=Run linker｜**Stop**

アセンブルの後に実行する操作を指定します。

RelativeModelPath=Yes｜**No**

モデル・パスをプロジェクト・ファイルのロケーションに相対的に格納するか，または絶対的に格納するかを指定します。

BeepOnEnd=Yes｜**No**

ビープ音を鳴らして make **操作の終了を示します。**

(c) 例

```
[MakeOptions]
StopOn=Warnings
AfterAssembling=Run linker
RelativeModelPath=Yes
BeepOnEnd=Yes
```

(9) SegmentClasses **セクション**

このセクションは，対応する WB77016 **の** *classname* Segment Class Properties **ダイアログで行われた設定から成り立っています。これらの設定は，所定のクラスのメンバであるセグメントのリストと同様にクラス・プロパティから成り立っています。このセクションには，ユーザ定義のセグメント・クラスの数に応じていくつかのブロックを入れることができます（各セグメント・クラスに** 1 **つのブロックが割り当てられます)。**

(a) セクション・ヘッダ

[SegmentClasses]

(b) セクション・エントリ

Class*n* **=** *classname*

n **は，**1 **から始まる連続番号です。**

classname **は，ユーザが定義するクラス名です。**

Type=boot｜**ini**｜**inid**｜**ovl**｜**preload**｜**dma**

ユーザ定義のセグメント・クラスのタイプを指定します。

Description=*description*

description **は，ユーザが入力するセグメント・クラスの説明です。**

Source=file｜**segment**

ホスト・ブートか，セルフ・ブートを指定します（ini，inid)。

Type=*memorytype memoryarea*

memorytype **は，セルフ・ブート時にメモリ・タイプを指定します。**

memoryarea **は，セルフ・ブート時に指定されたメモリ領域です。**

File=*hexfilename*

hexfilename **は，ホスト・ブート時の出力** HEX **ファイルのパスと名前です（**ini，inid)。

保守／廃止

HexFileFormat=ByteAddress｜**WordAddress**

HEX **ファイルの** HEX **ファイル・アドレシング・フォーマット。**

StartingFormat=Reset｜**Reboot**

ブート・パラメータをアドレス 0x4000:Y **で始まるセグメントにセットします（**Reset**）。**

ブート・パラメータをパブリック・シンボルとしてプロジェクトにエクスポートします（Reboot**）。**

DataFormat=Byte｜**Word**

各命令（4 **バイト）を** 4 **つのデータ・メモリ・アドレス（**4 **データ・メモリ・ワード）にコピーしま**

す→ Byte

各命令を 2 **つのデータ・メモリ・アドレス（**2 **データ・メモリ・ワード）にコピーします→** Word

Fragments=*n*

n **は，**1-256 **の範囲のブート・データ・フラグメントの数です。**

Allocation=block｜**start at** *address*｜**bank** *number*

address **はセグメント開始アドレスです。**

number **はキャッシュ・バンクの数です。**

ClassMember*n* **=** *module*,*segement*,*initialization*

このリストには，所定のクラスのメンバであるセグメントが入っています。

n **は，**1 **から始まる連続番号です。**

module **は，セグメントを含むモジュールの名前です。**

segment **は，所定のクラスのメンバであるセグメントの名前です。**

initialization **は，**zero **または** data **のどちらかであるセグメント初期化モードです。**

NonOverlayGroup*n* **=** *groupname*

n **は，**1 **から始まる連続番号です。**

groupname **は，非重複グループの名前です。**

NonOverlayGroupMembern=*segmentname*

n **は，**1 **から始まる連続番号です。**

segmentname **は，非オーバレイ・グループのメンバの名前です。**

（c）例

```
[SegmentClasses]
Class1=boot_classs
Type=boot
Description=This is a boot class segment
Type=xmem int
StartingFormat=Reboot
DataFormat=Word
ClassMember1=FIR_MAIN,FIRBOOT
ClassMember2=INIT_SEC,INIT_FIRBOOT
ClassMember3=INIT_COUNT,Init_COUNTBOOT
```

保守／廃止

（10）AllocateSegments セクション

このセクションは，WB77016 の Edit Segments ダイアログのセグメント属性リストから成り立っています。このセクションには，セグメント数に応じていくつかのブロックを入れることができます（各セグメントに 1 つのブロックが割り当てられます）。

（a）セクション・ヘッダ

[SegmentAllocation]

（b）セクション・エントリ

Segment*n* **=** *modulename*,*segmentname*

n は，1 から始まる連続番号です。

*modulenam*e は，対応するセグメントが入っているモジュールの名前です。

segmentname は，定義したセグメントの名前です。

Allocation=*memoryarea*,**relocatable**｜**start at** *address*｜**aligned on** *boundary*

memoryarea は，セグメントがあるメモリ領域です。

address は，セグメント開始アドレスです。

*boundar*y は，セグメント整列境界です。

（c）例

```
[SegmentAllocation]
Segment1=MODULEABC,SEGMENTDEF
Allocation=ram,int,start at 0x200
```

★ 3.8 セグメント割り当て順序ファイル

ファイル中のセグメントの順序は，DSP メモリでのセグメントの割り当て順序を指定します。連続するセグメントは，DSP メモリの次に可能なアドレスに割り当てられます。セグメント順序ファイルは，連続するセグメントがギャップなしに割り当てられることを保証するものではないことに注意してください。セグメント順序ファイルは，属性がリロケータブルか，整列のセグメントで処理されます。セグメント順序ファイルは，ユーザによって作成され，Edit Segments ダイアログを経由して公表されます。リンク中に，セグメント順序ファイルは，評価され，望まれるセグメント割り当てが実行されます。

セグメント順序ファイルは，プレーンな ASCII テキスト・ファイルなので，どのテキスト・エディタでも編集することができます。この中には，次の形式で 1 行ごとに 1 つのエントリがあります。

modulename segmentname

modulename は，セグメントが入っているファイルの名前または NAME ディレクティブによって指定されるモジュールの名前です。

segmentname は，有効なセグメントの名前です。

セグメント順序ファイルのどこにでもコメントを挿入することができ，その場合，セミコロン “ ;”を前に付ける必要があります。

セグメント順序ファイルのパスは，絶対的にするか，現在のプロジェクト・パスに相対的にするかを指定できます。

保守／廃止

次に，セグメント順序ファイルの内容の例を示します。

```
__NEW__ DMA$DMA_EX1$xseg2
__NEW__ DMA$DMA_EX1$xseg1
```

こうすると，リロケータブル・セグメント__NEW__.DMADMA_EX1xseg2 が，リロケータブル・セグメント__NEW__.DMADMA_EX1xseg1 の前に割り当てられます（現在のコンフィギュレーションで）。

2 つのリロケータブル・セグメント DMADMA_EX1xseg1 と DMADMA_EX1xseg2 がセグメント順序ファイル内で参照されなかった場合，xseg1 の方が xseg2 より大きいので，割り当て順序は DMADMA_EX1xseg2 の前に DMADMA_EX1xseg1 となります。この機能は，DMA メモリに固有のものではなく，すべてのタイプのセグメント（インストラクション・メモリ，X データ・メモリ，Y データ・メモリの各セグメント）に使用可能なものです。

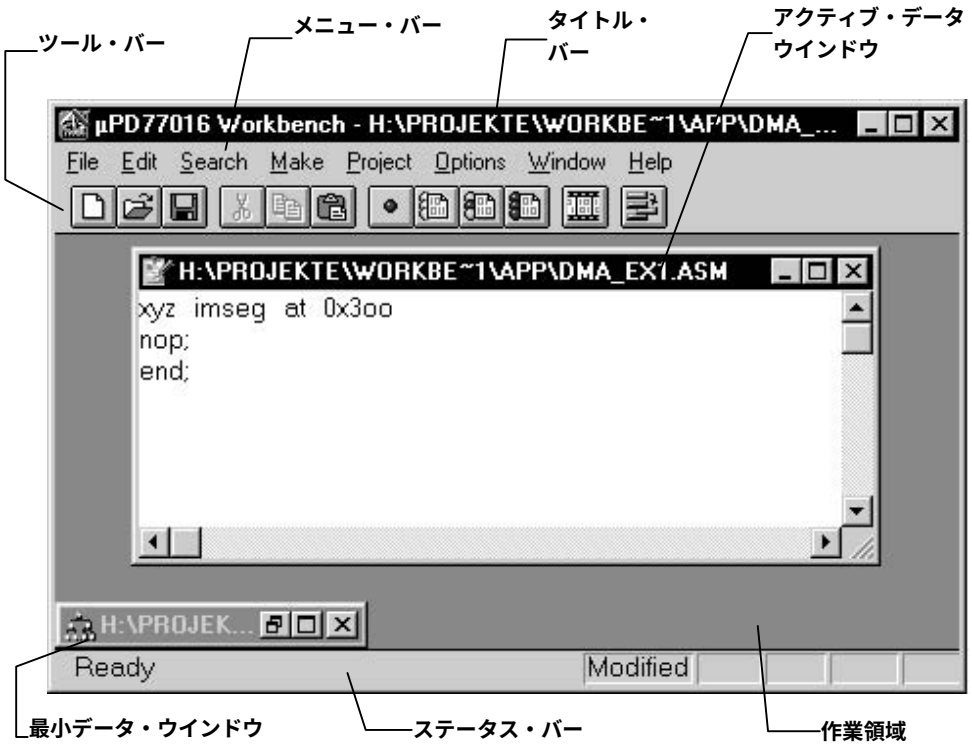
第 4 章　WB77016 ウインドウ

ユーザが行う作業のほとんどは，WB77016 のウインドウ内で実行されます。WB77016 ウインドウは複数オープンすることもできますが，一度にアクセスできるウインドウは 1 つだけです。メニュー・コマンドやユーザの操作は，普通このアクティブ・ウインドウに対して適用されます。

4.1　メイン・ウインドウ

図 4－1 に，WB77016 のメイン・ウインドウを示します。

図 4－1　WB77016 メイン・ウインドウ



4.1.1　メイン・ウインドウの構成要素

(1) メニュー・バー

タイトル・バーの下に置かれているバーを使用すると，WB77016 のすべてのコマンドにアクセスできます。

(2) タイトル・バー

タイトル・バーは WB77016 のメイン・ウインドウの上端にあります。タイトル・バーには，アプリケーション・プログラム名が表示され，システム・メニュー・ボックス，メイン・ウインドウの画面サイズを変更する 2 つのボタン，および閉じる（Close）ボタンがあります。ファイルがロードされている場合はアクティブ・ファイルの名前も表示されます。

保守／廃止

(3) ツール・バー

メニュー・バーの下にあるバーを使うと，マウス・クリックで WB77016 **の頻繁に使用されるコマンドに素早くアクセスすることができます。**

(4) ステータス・バー

メイン・ウインドウの下端にあるバーには，ユーザ・サポート・メッセージを表示します。

(5) 作業領域

データ・ウインドウを表示する領域です。

(6) ウインドウの種類

作業領域に表示されるものには次の種類があります。

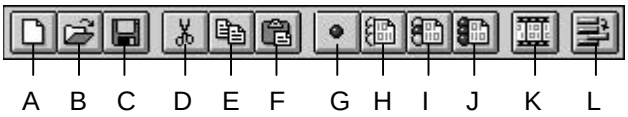
- **エディタ・ウインドウ**
- **プロジェクト・ウインドウ**
- **メッセージ・ウインドウ**
- **マップ・ファイル表示ウインドウ**
- **ステータス・ダイアログ**

4.1.2 ツール・バー

WB77016 **アプリケーションには，ボタン・クリックでファイル処理，クリップボード・インタフェース，**make **コマンドに素早くアクセスできるツール・バーが，メニュー・バーの下に付いています。ツール・バーは，**Options **メニューの“** Toolbar”**コマンドによってオン／オフの切り替えができます。コマンドの左側にあるチェック・マーク（✓）は，アクティブなツール・バーを示します。**

図 4－2 **にツール・バーを示します。**

図 4－2　**ツール・バー**



A　：New **ボタン**
B　：Open **ボタン**
C　：Save **ボタン**
D　：Cut **ボタン**
E　：Copy **ボタン**
F　：Paste **ボタン**
G　：Assemble **ボタン**
H　：Link **ボタン**
I　：Make **ボタン**
J　：Build All **ボタン**
K　：Simulate **ボタン**
L　：Edit Segments **ボタン**

New **ボタン**

ディフォールト名“ Untitled number”**で新規のエディタ・ウインドウをオープンすることによって，新しいソース・ファイルの準備をします。“** Untitled”**ファイルは，一時的な編集バッファとして使用されます。ユーザは，保存時に “** Untitled”**ファイルに名前を入力するように促されます。**

Open ボタン

次のファイルをロードする標準ファイル選択ダイアログを表示します。

・アセンブラまたは C 言語ソース・ファイル

・ヘッダ・ファイル

・エラー・ファイル

・バックアップ・ファイル

・任意のテキスト・ファイル

Save ボタン

現在アクティブなエディタ・ウインドウにファイルを元のファイル名で保存します（現在のファイル名とパスは，エディタ・ウインドウのタイトル・バーに表示されます）。したがって，エディタ・ウインドウがアクティブな場合にだけ，“ Save”コマンドを使うことができます。“ Untitled”ファイルを保存すると，“ Untitled”ファイルの名前を変更し保存する標準ファイル選択ダイアログが表示されます。

Cut ボタン

現在選択されているデータを除去し，それをクリップボードに移動します。この操作を行うと，クリップボード上の既存のデータは上書きされるので注意してください。

Copy ボタン

現在選択されているデータのコピーを作成し，そのコピーをクリップボードに移動します。この操作を行うと，クリップボード上の既存のデータは上書きされるので注意してください。

Paste ボタン

カーソル位置にクリップボードからデータを挿入するか，現在選択されているデータを置換します。この操作をしても，クリップボード・データは影響を受けません。

Assemble ボタン

ファイルが最新であるかどうかに関わらず，現在アクティブなエディタ・ウインドウのソース・ファイル，またはプロジェクト・ウインドウ上で選択されたソース・ファイルをアセンブルします。

Link ボタン

ファイルが最新であるかどうかに関わらず，現在のプロジェクトに入っているファイルをリンクします。リンカ出力オプションを Options メニューの “ Linker Settings…”コマンドによって変更した場合には，“ Link”コマンドを選択する必要があります。

Make ボタン

ファイルが最新であるかどうかに関わらず，現在のプロジェクトを再構築します。“ Make”コマンドを使用すると，プロジェクト管理が現在のプロジェクトのファイルをアセンブルし，プログラムにリンクさせます。“ Make”は，（MS-DOS™ タイム・スタンプに従って）変更されたプロジェクトのファイルだけを再構築します。“ Make”コマンドは，プロジェクト・ウインドウがアクティブになっている場合にのみ使用可能です。

Build All ボタン

ファイルが最新であるかどうかに関わらず，プロジェクトのすべてのファイルを再構築します。“Build All” コマンドは，プロジェクト・ウインドウがアクティブになっている場合にのみ可能です。

Simulate ボタン

ファイルが最新であるかどうかに関わらず，プロジェクトのすべてのファイルを再構築します。構築プロセスがエラーなしで終了すると（リンク・ファイルが生成されます），HSM77016 が起動します[注]。シミュレータの入力ファイルは，WB77016 のリンカによって生成されるリンク・ファイルです。

注　HSM77016 がセットアップされている必要があります。

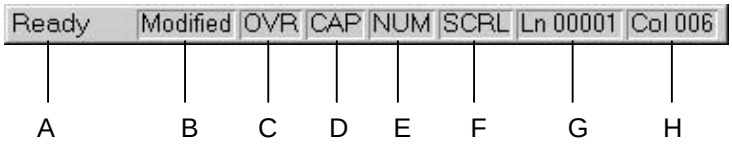
Edit Segments ボタン

Edit Segments ダイアログをオープンします。

4.1.3　ステータス・バー

ステータス・バーの右側には，キー・ステータス・フィールド，行とカラムのインディケータが表示されます。ステータス・バーは，Options メニューの“ Statusline”コマンドによってオン／オフの切り替えができます。コマンドの左側にあるチェック・マーク（✓）は，アクティブなステータス・バーを示します。
図 4－3 にステータス・バーを示します。

図 4－3　ステータス・バー



A：メッセージ・フィールド
B：テキスト・ステータス・フィールド
C, D, E, F：キー・ステータス・フィールド
G：行インディケータ
H：カラム・インディケータ

メッセージ・フィールド

メッセージ・フィールドは，任意の WB77016 メニュー・コマンドに関するヒント，および現在の WB77016 の動作に関するメッセージを表示します。

テキスト・ステータス・フィールド

このフィールドは，現在アクティブなエディタ・ウインドウのステータスを報告します。エディタ・ウインドウがオープンした後にエディタ・ウインドウのテキストが変更された場合，テキスト・ステータス・フィールドは“ Modified”になります。

保守／廃止

キー・ステータス・フィールド

これらのフィールドは，編集モード・ステータス，およびキーボード上の CAPS LOCK，NUM LOCK，SCROLL LOCK **キーのステータスを示します。**

・CAP:　CAPS LOCK **になっていることを示します。**

・NUM:　**数値キーパッドがアクティブであることを示します。**

・SCRL:　SCROLL LOCK **になっていることを示します。**

編集モード

編集モードは，“ INS”(Insert)，**“** OVR”(Overwrite)，**“** VIEW”(Viewer)**のいずれかになります。上書きモードと挿入モードを切り換えるには，**Insert **キーを押します。エディタ・ウインドウがビューワ・モードであれば，それは読み取り専用であり，編集できません。ビューワ・モードをオン／オフにするには，**Ctrl+O **を押します。**

行インディケータ

カーソルがあるエディタ・ウインドウの行を示します。

カラム・インディケータ

カーソルがあるエディタ・ウインドウのカラムを示します。

4.2　エディタ・ウインドウ

エディタ・ウインドウは，主にアセンブラ・ソース・ファイルを編集したりその内容を見たりする場合に使用します。タイトル・バーには，編集ファイルの名前，またはまだ新規ファイル名で保存されていないエディタ・ウインドウに対しては“ UNTITLED”**が表示されます。**

エディタ・ウインドウの内容は，次のメニューで管理します。

・File **メニューには，エディタ・ウインドウ内のファイルについてロード，保存，クローズ，プリンタ出力などを行うためのすべてのコマンドが載っています。**

・Edit **メニューを使用すると，テキストのカット，コピー，ペーストを行うことができます。**

・Search **メニューを使用すると，アセンブラ・ソース・ファイル中のテキストの検索を行うことができます。**

・Options **メニューの“** Editor…”**コマンドを使用すると，ユーザ要件に従ってエディタを構成することができます。**

注意　File**，**Edit**，および** Search **メニュー・コマンドは，アクティブなエディタ・ウインドウにだけ適用されます。**
Options **メニューの“** Editor…”**コマンドを使用した設定は，すべてのエディタ・ウインドウに適用されます。**

保守／廃止

4.3　メッセージ・ウインドウ

メッセージ・ウインドウには，現在の WB77016 の動作（アセンブル，リンク），アセンブル／リンクされたファイルの名前，エラー・メッセージ，ワーニング・メッセージが表示されます。

ソース・プログラム・リスト内のエラーを見つけ出すには，エラー・メッセージが表示されている行をダブル・クリックします。ソース・ファイルがオープンし，エラーが発生したと思われる行が強調表示されます。

注意　メッセージ・ウインドウをオープンまたは再オープンするには，Window メニューの“ Message”コマンドを選択します。Make メニューのコマンドを選択すると，自動的にメッセージ・ウインドウがオープンします。

4.4　プロジェクト・ウインドウ

プロジェクト・ウインドウには，プログラムを構築するためのファイルのリストが表示されます。プロジェクト・ウインドウのタイトル・バーには，アクティブなプロジェクトの名前が表示されます。プロジェクトに含まれるソース・ファイルをエディタ・ウインドウにロードするには，希望するファイルがリストされている行をダブル・クリックします。

プロジェクト・ウインドウは，複数ファイル選択をサポートし，選択したファイルについて次の動作を実行します。

- プロジェクト内のファイルとモデル定義ファイルの相対ファイル・パスを格納します。
- 選択したファイルのローカル・アセンブラ・オプションを設定します。
- 選択したファイルをアセンブルします。
- 選択したファイルを削除します。

注意　オープンできるプロジェクトは一度に 1 つだけです。プロジェクトのクローズは，プロジェクト・ウインドウのクローズではなく，Project メニューから“ Close project”コマンドを選択するか，または Window メニューから“ Close all”コマンドを選択して行います。Window メニューの“ Project”コマンドを使用すると，クローズしたプロジェクト・ウインドウを再オープンできます。

保守／廃止

4.5　ステータス・ダイアログ

ステータス・ダイアログには，アセンブルまたはリンク中に次のような情報が表示されます。

・現在の入力ファイルの名前
・現在の WB77016 の動作（アセンブル，リンク）
・処理済み行数（総数およびファイル別個数）
・発生したワーニングの個数（総数およびファイル別個数）
・発生したエラーの個数（総数およびファイル別個数）

ステータス･ダイアログの Abort ボタンを使用すると,現在の WB77016 の動作を中断することができます。make が終了したあと，Abort ボタンは OK ボタンに変わります。

注意　OK ボタンをクリックすると，ステータス・ダイアログはクローズします。ステータス・ダイアログは，アセンブルまたはリンク中に自動的に再オープンします。

make を終了したあと，ステータス・ダイアログには次のメッセージのうちの 1 つが表示されます。

- Success:　リンカ出力ファイルを生成した場合
- Errors occurred:　プロジェクトの構築でエラーが発生した場合
- User break:　ユーザが Abort ボタンを押した場合
- Linker stop:　リンカがストップした場合

  次の原因が考えられます

  - “Warnings”コントロールが Options メニューの“ Make…”コマンドで設定され，アセンブル中にエラーは発生しなかったがワーニングが発生した場合
  - “All sources processed”コントロールが Options メニューの“ Make…”コマンドで設定された場合
  - “Stop”コントロールが Options メニューの“ Make…”コマンドで設定された場合
  - Is up to date:　make を最後に実行して以来プロジェクトに対し変更がなかった場合

［メ　　モ］

★

第5章　チュートリアル

5.1　プロジェクト管理

プロジェクト管理は，1 つのプロジェクトに属するすべてのファイルを管理するものです。ほとんどのプログラムは，複数のファイルで構成されます。プロジェクト管理は再アセンブルおよびリンクを必要とするファイル群を自動的に識別します。プロジェクトが再構築されると，プロジェクト管理はプロジェクト・ファイル情報を更新します。

プロジェクト・ファイル情報に含まれるものは次のとおりです。

- プロジェクトの全ファイル
- ディスク上のファイルの場所
- Make，アセンブラ，およびリンカのオプション
- 各ソース・モジュールのヘッダ・ファイル
- 出力ファイルの格納先

5.1.1　プロジェクトのサンプリング

WB77016 エディタで編集，保存された 3 つのファイルからなるプログラムがあると仮定します。3 つのファイルとは，メインのソース・ファイル“MAIN.ASM”と 2 つの補助ファイル“SUP1.ASM”および“SUP2.ASM”です。

(1) 新規プロジェクト・ファイル

最初のステップでは，Project メニューの“Open Project…”コマンドを選択してプロジェクト・ファイルを作成します。このコマンドは，プロジェクト・ファイルを選択または作成する標準ファイル選択ダイアログを表示します。File Name フィールドに名前を入力して，新しいプロジェクト・ファイルを作成します。

(2) 項目の追加

次のステップは，プロジェクトに項目を追加する作業です。Project メニューの“Add Item…”コマンドを選択します。このコマンドは，ファイルをプロジェクトに追加する標準ファイル選択ダイアログを表示します。一度に複数のファイルを追加することができます。考えられるファイルのタイプは次のとおりです。

- アセンブラ・ソース・ファイル（拡張子“.ASM”）
- リロケータブル・アセンブラ出力ファイル（拡張子“.REL”）
- μPD77016 ライブラリアンで作成したライブラリ・ファイル（拡張子“.LIB”）
- リンカ出力ファイル（拡張子“.LNK”）。ロード・モジュール・ファイルを選択している場合は，それがプロジェクト内のただ 1 つのファイルでなければならないことに注意してください。

すべてのファイルを選択したら，OK ボタンをクリックして，ファイルをプロジェクト・ウインドウに追加します。再び Add Item ダイアログが表示されたら，別のファイルをプロジェクトに追加します。ファイル選択が終了したら，Close ボタンをクリックして，ダイアログを終了します。これで，プロジェクト・ウインドウに 3 つの選択ファイルが表示されます。

保守／廃止

(3) オプションの設定

Options メニューに移動し，対応するコマンドを選択して，目的のアセンブラ，リンカ，HEX コンバータの出力オプションを設定します。Options メニューの“Directories…”コマンドを使用して，インクルード・ファイル，ソース・ファイル，出力ファイルのパスを指定します。

これでプロジェクトが構成されます。

注意　プロセッサ・モデルが異なれば，同じソース・ファイルでも違ったリンク結果になる場合があります。
Options メニューの“Processor Model…”コマンドを使用して，プロジェクトのファイルと一致するモデルを選択します。

(4) Make

プログラムを構築するには，Make メニューで“Make”コマンドを選択します。Make ユーティリティによって，プロジェクトのすべてのファイルに対してアセンブルおよびリンクを行います。アセンブルおよびリンクがエラーを起こさずに進行している場合には，Status ダイアログにアセンブルおよびリンクが進行中であることが表示されます。

5.1.2　プロジェクトの管理

(1) ソース・ファイルの編集

ソース・ファイルを変更または表示するには，そのファイルの名前が表示されているプロジェクト・ウインドウ内の行をダブルクリックします。ファイルの内容がエディタ・ウインドウに表示されます。

(2) 項目の削除

プロジェクトからファイルを削除するには，そのファイルの名前が表示されているプロジェクト・ウインドウ内の行をクリックし，Project メニューの“Delete Item”コマンドを選択します。

(3) インクルード・ファイルとヘッダ・ファイル

プロジェクトに“MAIN.ASM”および“SUP1.ASM”で使用する宣言などが入っているインクルード・ファイルまたはヘッダ・ファイルがあると仮定します。再び Make を実行すると，プロジェクト管理は，プロジェクトのファイル間の依存関係を自動的に調べます。

ファイル“MAIN.ASM”と“SUP1.ASM”がヘッダ・ファイルに依存していれば，それらのファイルを再アセンブルする必要があります。ファイル“SUP2.ASM”は最新のものです。プロジェクト管理は，最新でないファイル群のみを再アセンブルし，リンクすることを保証しています。

プロジェクト内の 1 つのファイルに関連するインクルード・ファイルを変更したり表示したりするには，次のようにします。

- プロジェクト・ウインドウで目的のファイルを選択します（たとえば，“MAIN.ASM”）。
- Project メニューの“Include Files…”コマンドを選択し，“MAIN.ASM”に関連するインクルード・ファイルを表示します。
- リストの中からインクルード・ファイルを選択し，Edit ボタンをクリックしてインクルード・ファイルをエディタ・ウインドウにロードします。

5.1.3　エラー追跡

メッセージ・ウインドウ内に構文エラーまたはワーニングのメッセージが表示されたら，そのエラー・メッセージが表示されている行をダブルクリックします。対応するソース・ファイルがエディタ・ウインドウに表示され，カーソルはエラーの発生位置へ移動します。

エラーが複数ある場合，Search メニューの“Next error”コマンドまたは“Previous error”コマンドを選択して，カーソルを次のエラーまたは前のエラーの位置へ移動します。

注意　ソース・ファイルを変更すると，そのオブジェクト・ファイルが古くなるので，プロジェクト管理が再アセンブルを実行します。

5.1.4　Make の停止

Options メニューの“Make…”コマンドを選択して，Make を停止させる条件を選択できます。ディフォールトの設定は，break a make on errors（ファイルをアセンブルしてエラーが発生したとき Make 動作が停止する）です。

注意　Status ダイアログの Abort ボタンをクリックしても Make を停止できます。

［メ　　モ］

保守／廃止

★ 第 6 章　メニュー・コマンドとダイアログ

6.1　File メニュー

6.1.1　New

“New”コマンドを使用すると，ディフォールト名“UNTITLED”のエディタ・ウインドウが新規にオープンされます。これらのタイトルがついていないファイルは一時的な編集バッファとして使用されます。“UNTITLED”ファイルに“ Save”コマンドを適用すると，タイトルがついていないファイルを新しいファイル名で保存するための標準ファイル選択ダイアログが表示されます。

6.1.2　Open...

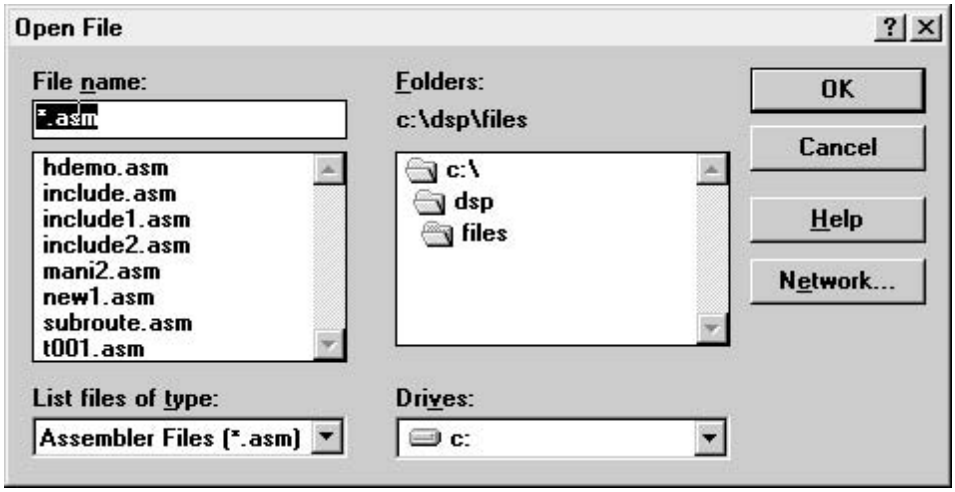
“Open…”コマンドを使用すると，ファイルをエディタ・ウインドウに表示する Open File ダイアログが表示されます。通常，このファイルは，アセンブラ・ソース・ファイル，C 言語ソース・ファイル，ヘッダ・ファイル，エラー・ファイル，バックアップ・ファイル，または任意のテキスト・ファイルです。

Open File ダイアログを図 6－1 に示します。

ファイルをオープンするには，ファイル名フィールドにファイル名とパスを入力するか，ディレクトリ／ドライブからファイルを選択します。ファイルの種類リスト（List files of type）からファイル・タイプ（拡張子）を選択すると，その拡張子のファイルが表示されます。希望するファイル・タイプを選択してください。

注意　Open File ダイアログは，既存のファイルをエディタ・ウインドウに表示します。ファイル・リストにないファイル名を入力すると，ユーザにファイル名の確認またはキャンセルを促すメッセージが表示されます。新規ファイルを作成するときは，“New”コマンドを使用してください。

図 6－1　Open File ダイアログ



保守／廃止

6.1.3　Merge...

“Merge…”コマンドを使用すると，現在アクティブなエディタ・ウインドウのカーソル位置にファイルを挿入する標準ファイル選択ダイアログが表示されます。

ファイルを挿入するには，次のようにします。

1. エディタ・ウインドウ内の挿入したい位置にカーソルを置きます。
2. “Merge…”コマンドを選択します。ファイル選択ダイアログが表示されるので，マージするファイルを選択します。選択したファイルの内容がカーソル位置に挿入されます。

6.1.4　Save

“Save”コマンドは，現在アクティブなエディタ・ウインドウのファイルを元のファイル名で保存します（ファイル名は，WB77016 のタイトル・バーに表示されます）。“UNTITLED”ファイルを保存しようとすると，Save As ダイアログをオープンするので，“UNTITLED”ファイルに名前をつけてから保存します。

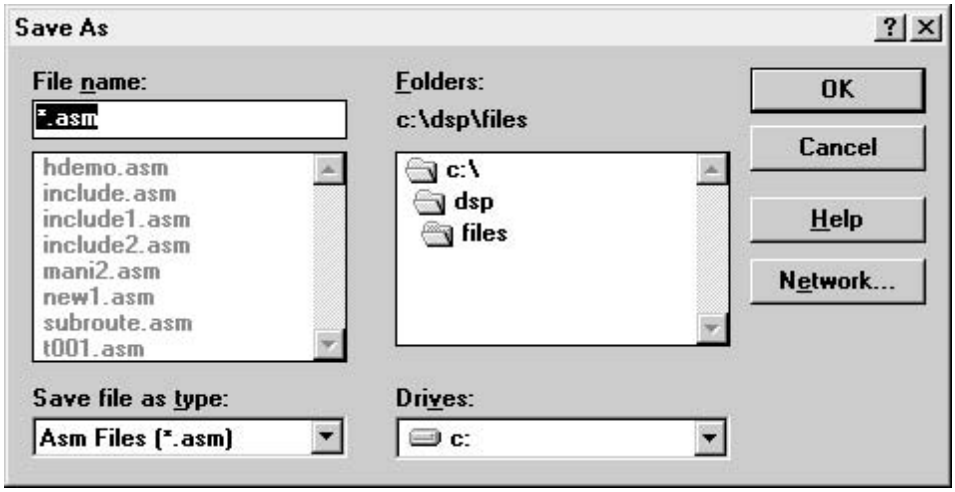
6.1.5　Save As...

“Save As…”コマンドを使用すると，アクティブなエディタ・ウインドウの内容を，異なるディレクトリまたは別のドライブに別のファイル名で保存する標準ファイル選択ダイアログが表示されます。

Save As ダイアログを図 6－2 に示します。

通常，保存するファイルは，アセンブラ・ソース・ファイルです。異なるファイル・タイプで保存する場合には，ファイル名に使用したい拡張子をつけて入力します。ファイル・タイプは Save files as type から選択することもできます。

図 6－2　Save As ダイアログ



6.1.6　Close

“Close”コマンドは，現在アクティブなエディタ・ウインドウをクローズします。エディタ・ウインドウのファイルの変更を保存するかどうか確認を求められます。

エディタ・ウインドウをクローズするには，次のようにします。

- エディタ・ウインドウの“閉じる”ボタンをクリックします。
- エディタ・ウインドウのシステム・メニューから“Close”コマンドを選択します。
- File メニューから“Close”コマンドを選択します。

6.1.7　Print...

“Print…”コマンドは，アクティブなエディタ・ウインドウの内容を印刷します。プリンタの用意ができていないときは，エラー・メッセージが表示されます。このコマンドは，エディタ・ウインドウが現在アクティブな場合にだけ使用できます。

6.1.8　Print Preview

“Print Preview”コマンドは，アクティブなドキュメントを印刷されたときのように表示します。このコマンドを選択すると，メイン・ウインドウが印刷プレビュー・ウインドウに置き換わり，1 ページまたは 2 ページが印刷フォーマットで表示されます。印刷プレビュー・ツール・バーには，1 ページか一度に 2 ページを表示したり，ドキュメント中を前後に移動したり，ページをズームインとズームアウトしたり，印刷ジョブを初期化したりするオプションがあります。

6.1.9　Printer Setup...

“Printer Setup…”コマンドは，プリンタのコンフィギュレーションを設定する標準の Windows ダイアログを表示します。ユーザの Windows システムには，1 つ以上のプリンタ・ドライバがインストールされている可能性があります。“Printer Setup…”コマンドを使用すると，WB77016 からどのプリンタを使用するかを選択することができます。使用可能なセットアップ・オプションは，インストールされているプリンタの機能に依存します。プリンタのセットアップに関するヘルプを表示させるには，Setup ダイアログの Help ボタンをクリックするか，または F1 を押します。このコマンドは，エディタ・ウインドウが現在アクティブな場合にだけ使用できます。

6.1.10　Exit

WB77016 のセッションを終了するには，次のようにします。

- File メニューの中から“Exit”コマンドを選択します。
- WB77016 のメイン・ウインドウのシステム・コントロール・ボックスをダブルクリックします。
- メイン・ウインドウのシステム・メニューから“Close”コマンドを選択します。

注意　WB77016 を終了する場合，エディタ・ウインドウに変更があれば，それを保存するかどうか確認を求められます。オープンしているプロジェクトは自動的に保存されます。Make 実行中は，WB77016 を終了できません。

6.2　Edit メニュー

エディタ・ウインドウの中のテキストは，カット，コピー，ペーストすることができます。変更したいテキストを選択して，Edit メニューの中のコマンドを実行してください。

6.2.1　Cut

“Cut”コマンドは，選択したデータをエディタ・ウインドウから削除し，クリップボードに入れます。

6.2.2　Copy

“Copy”コマンドは，選択したデータのコピーを作成して，そのコピーをクリップボードに入れます。

6.2.3　Paste

“Paste”コマンドは，クリップボードからのデータをカーソル位置に挿入します。現在選択しているものがあれば，それはクリップボードの内容で置き換えられます。

6.2.4　Delete

“Delete”コマンドは，エディタ・ウインドウから選択されたデータを削除しますが，クリップボードには何も入れません。

6.2.5　Select All

このコマンドをプロジェクト・ウインドウで使用すると，その中に含まれるすべてのプロジェクト項目が選択されます。このコマンドをエディタ・ウインドウで使用すると，その中に含まれるテキストが選択されます。

6.3　Editor Support メニュー

マウス・ポインタがエディタ・ウインドウの中にあるときに右クリックすると，Editor Support メニューが現われます。

Editor Support メニューを図 6－3 に示します。

図 6－3　Editor Support メニュー

6.3.1　Editor Support メニューのコマンド

（1）Insert

File name: エディタ・ファイル名をカーソル位置のところに挿入します。

date: 現在の日付けをカーソル位置のところに挿入します。

time: 現在の時刻をカーソル位置のところに挿入します。

（2）Delete

line: カーソルが置かれている行を削除します。

to end of line: カーソル位置以降の行を削除します。

to end of word: カーソル位置以降の単語を削除します。

selection: 選択したテキストを削除します。

（3）Move

top of file: **カーソルをファイルの先頭へ移動します。**

bottom of file: **カーソルをファイルの最後へ移動します。**

beginning of line: **カーソルを行頭へ移動します。**

end of line: **カーソルを行末へ移動します。**

find matching: **カーソルを対応するかっこへ移動します。カーソルは，検索を開始するかっこの位置の左側に置かれていなければなりません。かっこが閉じていない場合は，カーソルの位置は変わりません。**

（4）Convert

uppercase selection: **選択したテキスト中のすべての文字を大文字に変換します。**

lowercase selection: **選択したテキスト中のすべての文字を小文字に変換します。**

（5）Marker

set: **マーカ** 1-10 **をカーソル位置にセットします。マーカが置かれている行および列は，マーカ選択ボックスの中に表示されます。既存のマーカを再びセットすると，前のマーカ位置はクリアされ，新しいマーカが現在位置になります。**

goto: **カーソルをマーカ** 1-10 **のところへ移動します。**

（6）Center Line

“ Center Line”**コマンドは，エディタ・ウインドウをスクロールして，カーソルのある行をウインドウの中央に表示します。**

6.3.2　エディタの行の長さと行番号

エディタの行の最大長は 512 **文字です。**

エディタ・ファイルには 15,000 **行まで入れることができます。**

保守／廃止

6.4　Search メニュー

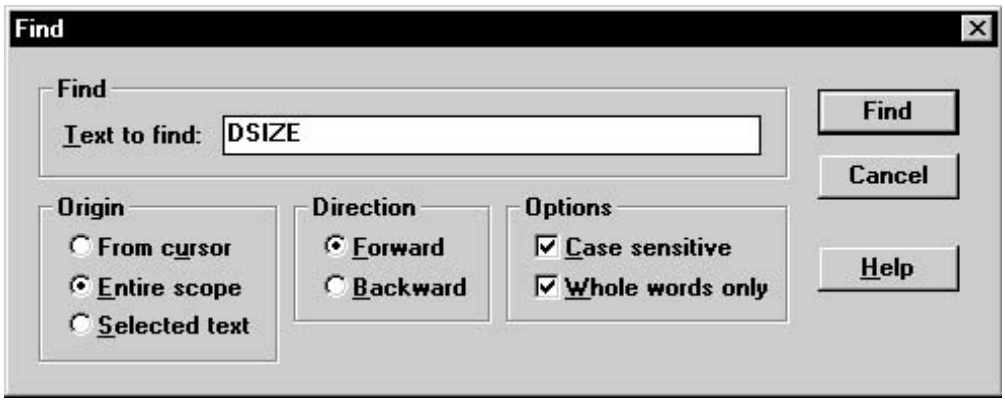
Search メニュー・コマンドは，エディタ・ウインドウ内のテキストの検索および置換を行ったり，特定行番号またはエラーの位置へジャンプする場合に使用します。

6.4.1　Find...

“Find…”コマンドは，エディタ・ウインドウ内のテキストを検索する Find ダイアログを表示します。

Find ダイアログを図 6－4 に示します。

図 6－4　Find ダイアログ



Find ダイアログの構成要素

Text to find

この入力フィールドは，検索文字列を入力するために使用されます。検索操作は，アクティブなエディタ・ウインドウの内容に適用されます。

From cursor

これを選択すると，カーソル位置から検索が開始されます。

Entire scope

ファイル全体を検索します（Direction ボタンで検索をファイルの先頭から始めるか，最後から始めるかを決定します）。

Selected text

選択したテキスト領域の中だけを検索する場合に，このボタンを選択します（Direction ボタンで検索を強調表示テキストの先頭から始めるか，最後から始めるかを決定します）。

Forward

Origin で指定した位置から前方を検索します。

Backward

Origin で指定した位置から後方を検索します。

Case sensitive

検索の際に大文字と小文字を区別するには，このコマンドをチェックします。

Whole words only

完全単語（英字だけで構成された文字列）だけを検索するには，このコマンドをチェックします。

Find

検索を開始します。

Cancel

ダイアログをクローズし，タイトル・バーに表示されている操作をキャンセルします。

6.4.2　Next

“Next”**コマンドは，検索テキストをカーソル位置から前方に検索します。**

6.4.3　Previous

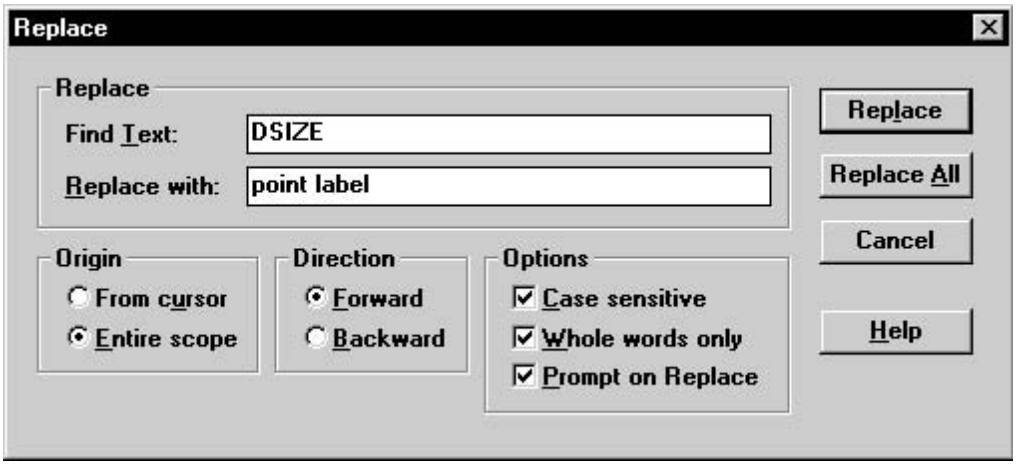
“Previous”**コマンドは，検索テキストをカーソル位置から後方に検索します。**

6.4.4　Replace...

“Replace…”**コマンドは，エディタ・ウインドウでテキストを検索し，このテキストをユーザ定義の文字列で置き換えるダイアログを表示します。**

Replace **ダイアログを図** 6－5 **に示します。**

図 6－5　Replace **ダイアログ**



Replace **ダイアログの構成要素**

Find Text

この入力フィールドは，変更前テキストを入力するために使用されます。

Replace with

この入力フィールドは，変更後テキストを入力するために使用されます。

From cursor

カーソル位置からテキストの置き換えを開始します。

Entire scope

ファイル全体を検索します（Direction **ボタンで置き換えをファイルの先頭から始めるか，最後から始めるかを決定します)。**

Forward

Origin **で指定した位置から前方を検索対象とし，テキストを置き換えます。**

Backward

Origin **で指定した位置から後方を検索対象とし，テキストを置き換えます。**

Case sensitive

このコマンドをチェックすると，置き換えの際に大文字と小文字が区別されます。

Whole words only

完全単語（英字だけで構成された文字列）だけを検索するには，このコマンドをチェックします。

Prompt on Replace

置き換えのたびに，その確認，拒否，キャンセルを求めるダイアログを起動します。

Replace

このボタンは，1 **件のみ検索テキストを置き換えます。**

Replace All

このボタンは，検索対象すべてのテキストを置き換えます。検索対象が見つかるたびに，エディタ・ウインドウの内容がスクロールされてテキストが表示されます。

Cancel

ダイアログをクローズし，タイトル・バーに表示されている操作をキャンセルします。

6.4.5　Next Error

“Next Error”**コマンドは，カーソルをエディタ・ウインドウ内の次のエラー，ワーニングが発生した位置へ移動します。このコマンドは，メッセージ・ウインドウ内に関連するエラー・メッセージまたはワーニング・メッセージがある場合にのみ有効です。**

6.4.6　Previous Error

“Previous Error”**コマンドは，カーソルをエディタ・ウインドウ内の前のエラー，ワーニングが発生した位置へ移動します。このコマンドは，メッセージ・ウインドウ内に関連するエラー・メッセージまたはワーニング・メッセージがある場合にのみ有効です。**

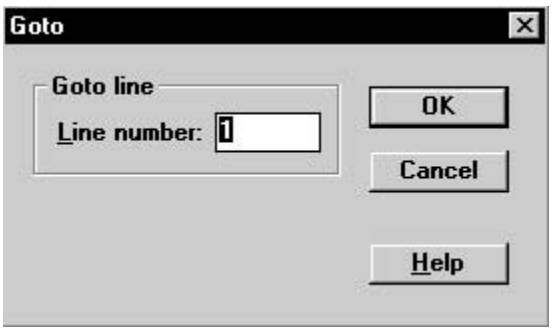
保守／廃止

6.4.7　Goto Line...

“Goto Line…”コマンドは，移動先の行番号の入力を求めるダイアログを表示します。

Goto ダイアログを図 6－6 に示します。

図 6－6　Goto ダイアログ



Goto ダイアログの構成要素

Line number

この入力フィールドは，目的の行番号を入力するために使用されます。

OK

ダイアログをクローズし，カーソルをアクティブなエディタ・ウインドウの指定された行の先頭に置きます。番号の形式が正しくないか，行番号が範囲外であると，エラー・メッセージが表示されます。

Cancel

ダイアログをクローズし，タイトル・バーに表示されている操作をキャンセルします。

6.5　Make メニュー

6.5.1　Assemble

“Assemble”コマンドは，アクティブなエディタ・ウインドウの中のファイル，またはプロジェクト・ウインドウの中の強調表示されているアセンブラ・ファイルを，最新かどうかに関係なく，アセンブルします。

6.5.2　Link

“Link”コマンドは，現在オープンしているプロジェクトのファイルを，最新かどうかに関係なくリンクします。リンカ出力オプションを Options メニューの“Linker Settings…”コマンドで変更している場合に，このコマンドを選択してください。

注意　プロセッサ・モデルを変更したり，プロセッサ・モデル・ファイルに変更があると，エラーおよびワーニング数などのリンク結果が違ってくる場合があります。プロセッサ・モデルは Options メニューの“Processor Model…”コマンドで変更または表示することができます。

6.5.3　Make

“ Make”コマンドを使用すると，プロジェクト管理は現在オープンしているプロジェクトのファイル群をアセンブルし，リンクして 1 つのプログラムにします。“ Make”は，変更されているプロジェクトのファイルだけを再構築します。

注意　ファイルを再構築するかどうかは，オペレーティング・システムが各ファイルに刻印するタイム・スタンプによってのみ決定されます。プロジェクト管理は，それぞれのロード・モジュール・ファイルの時刻と日付けを，関係するソース・モジュール・ファイルの時刻および日付けと比較します。したがって，システムの時刻および日付けは正しく設定してください。

このコマンドを選択すると，メッセージ・ウインドウと Status ダイアログが表示されます。これらには，現在の WB77016 の動作に関する情報，エラー・メッセージ，およびアセンブル，リンクの進行状況（エラー数，ワーニング数，行番号など）が表示されます。

6.5.4　Build All

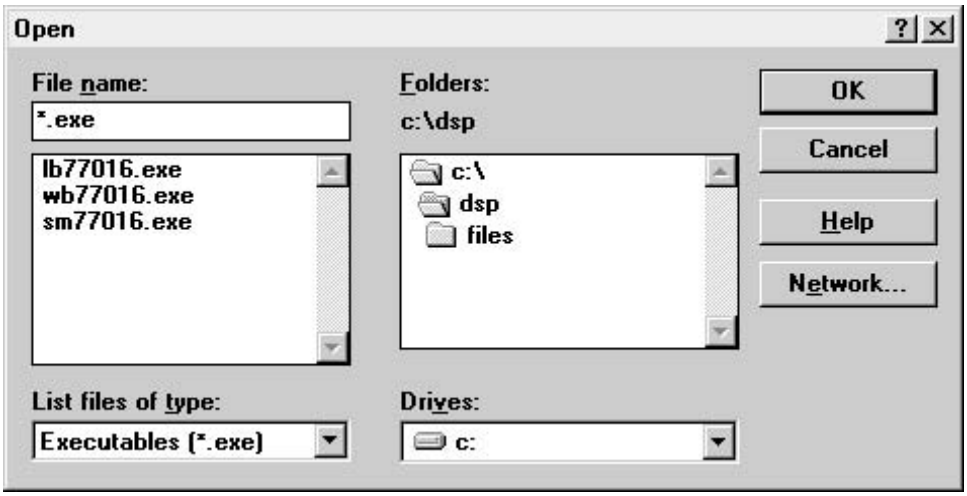
“Build All”コマンドを使用すると，プロジェクト管理は最新かどうかに関係なくプロジェクトの全ファイルを再構築します。

6.5.5　Simulate

“Simulate”コマンドは，“Make”を起動します。“Make”の入力ファイルは，現在のプロジェクト・ファイルです。“Make”がエラーを起こさずに実行できると（リンク・ファイルが生成され），HSM77016 が起動します。この HSM77016 の入力ファイルは，WB77016 が作成するリンク・ファイルです。

Select Simulator ダイアログを図 6－7 に示します。

図 6－7　Select Simulator ダイアログ



HSM77016 がカレント・ディレクトリになかった場合には，手動選択用のダイアログが表示されます。そこで，シミュレータの名前とシミュレータの実行可能ファイルが置かれているパスを指定してください。

注意　“Simulate”コマンドは，プロジェクトがオープンされているときだけ有効です。

保守／廃止

6.6　Project メニュー

Project メニューには，プロジェクト・ファイルの作成，オープン，クローズを行なったり，ファイルをプロジェクトに追加，削除したり，プロジェクト内の特定のファイルに対してローカル・オプションを設定したりするためのコマンドがあります。

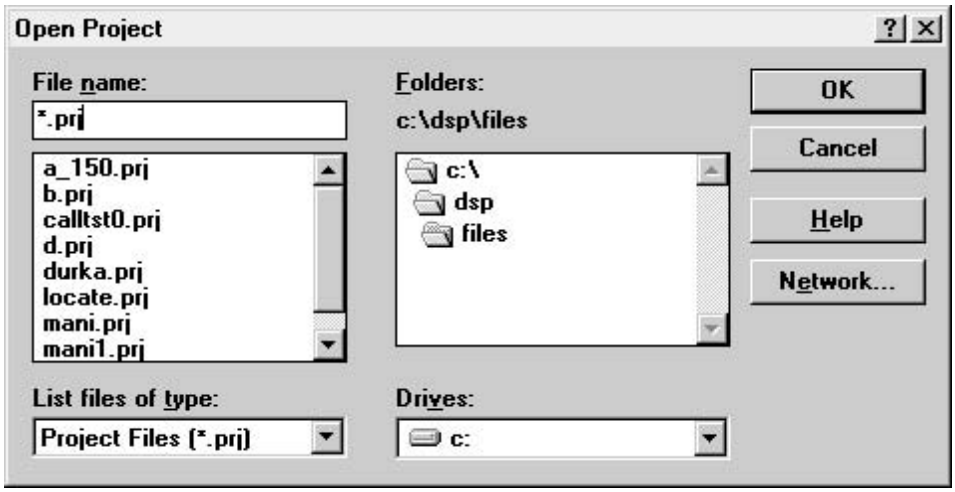
6.6.1　Open Project...

このコマンドは，プロジェクト・ファイルを選択したり，プロジェクト名を入力して新規にプロジェクト・ファイルを作成するための Open Project ダイアログを表示します。

プロジェクト・ファイルには，ロード・モジュール・ファイルを構築するためのすべての情報が格納されています。プロジェクト・ファイルのディフォールト拡張子は “ .PRJ”です。プロジェクト・ファイルの詳細については，第 3 章　ファイル・フォーマットを参照してください。

Open Project ダイアログを図 6－8 に示します。

図 6－8　Open Project ダイアログ



6.6.2　Save Project

“Save Project”コマンドは，オープンしているプロジェクト・ファイルを保存します。

保守／廃止

6.6.3　Save Project As...

このコマンドを使用すると，プロジェクト・ファイルの名前を指定し，それを目的のディレクトリに保存する標準ファイル選択ダイアログが表示されます。

プロジェクト内のすべてのファイルを含むプロジェクト全体を別のディレクトリに移動するには，次のようにします。

1. プロジェクト・ウインドウに表示されている全ファイルを選択します。
2. Options メニューの“Make…”コマンドを選択します。
3. “Selected project items”ボックスをチェックします。
4. Project メニューの“Save As…”コマンドを選択します。
5. ディレクトリとプロジェクト名を指定します。
6. プロジェクト内のすべてのファイルを目的の新しい場所に移動します。ファイルは，以前のプロジェクト・ファイルとの相対的な位置に移動させるようにしてください。

プロジェクトを新しい場所に移動するとき，モデル定義ファイルも移動したい場合があります。このような場合，プロジェクトの保存と移動をする前に，Make Settings ダイアログの“Model definition file”ボックスをチェックしてください。モデル定義ファイルのパスはプロジェクト・ファイルを起点にして格納されており，新しい場所に移動できます。モデル定義ファイルは，以前のプロジェクト・ファイルとの相対的な位置に移動させるようにしてください。

6.6.4　Close Project

プロジェクトを閉じるには，“Close Project”コマンドを選択します。

注意　このコマンドを選択した場合や WB77016 **セッションを終了した場合，プロジェクトは自動的に保存されます。**

保守／廃止

6.6.5　Add Item...

ファイルをプロジェクトのファイル・リストに追加するには，“Add Item…”コマンドを選択します。標準ファイル選択ダイアログが表示され，ファイル・タイプとファイル名を選択します。次のファイル・タイプを選択できます。

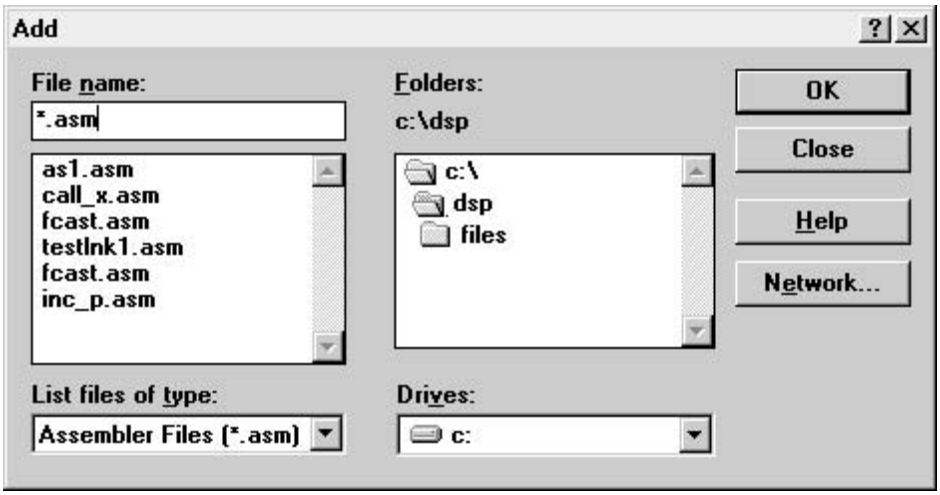
- アセンブラ・ソース・ファイル（拡張子“.ASM”）
- リロケータブル・アセンブラ出力ファイル（拡張子“.REL”）
- μPD77016 ライブラリアンで作成したライブラリ・ファイル（拡張子“.LIB”）
- リンク・ファイル（拡張子“.LNK”）。リンク・ファイルを選択している場合は，それがプロジェクト内のただ 1 つのファイルでなければならないことに注意してください。Build がリンク・ファイルの入っているプロジェクトに適用されると，プロジェクト・ファイル名に従って出力リンク・ファイルの名前がつけられ，HEX コンバータが起動します。

ファイルをプロジェクトに追加するには，目的のファイルの名前が載っている行をダブル・クリックするか，またはその行をクリックしてから OK ボタンをクリックします。標準のファイル選択方法を使用して一度に複数のファイルを選択できます。

ファイルの追加を中止し，ダイアログを終了するには，Close ボタンをクリックします。

Add Item ダイアログを図 6－9 に示します。

図 6－9　Add Item ダイアログ



6.6.6　Delete Item

ファイルをプロジェクトのファイル・リストから削除するには，目的のファイルを選択し，“Delete Item”コマンドを選択するか，または Delete キーを押します。

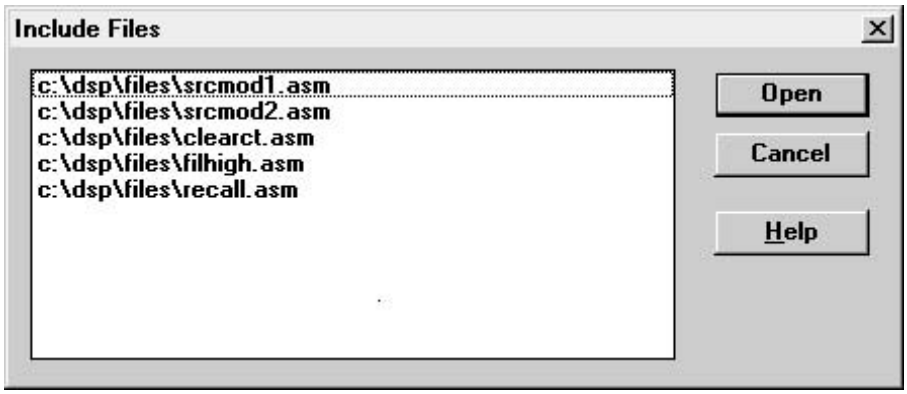
プロジェクト・ウインドウの複数選択機能を使用すると，一度に複数のファイルを削除できます。そのためには，目的のファイルを選択し，Project メニューの“Delete Item”コマンドを適用します。

6.6.7　Include Files...

“Include Files…”**コマンドは，プロジェクト・ウインドウで選択されたファイルに関連するインクルード・ファイルを表示するためのダイアログを表示します。**

Include Files **ダイアログを図** 6－10 **に示します。**

図 6－10　Include Files **ダイアログ**



Include Files **ダイアログの構成要素**

Include Files list

このフィールドは，検出されたインクルード・ファイルのパスと名前を表示します。

Open

インクルード・ファイルを編集するには，Include Files **リストからファイルを選択し，**Open **ボタンをクリックします。インクルード・ファイルがオープンし，エディタ・ウインドウに表示されます。**

Cancel

ダイアログをクローズします。

注意　インクルード・ファイルを変更すると，関連するすべてのファイルの日付けが古くなるため，“Make”**により再アセンブルされます。**

保守／廃止

6.7　Options メニュー

Options メニューでは，WB77016 の設定を変更します。

WB77016 の設定は，次のとおりです。

- Assembler

  アセンブラ出力オプションを変更します。
- Linker

  リンカ出力オプションの変更，セグメントの配置などを行います。
- Hexconversion

  HEX コンバータ出力オプションを変更します。
- Make

  プロジェクト管理のオプションを設定します。
- Directories
- インクルード・パス，ソース・パス，および出力パスを設定します。
- 表示および環境設定
- Editor

  joint や split lines などのエディタ・オプションを変更します。
- Font

  エディタ・ウインドウのフォントの種類，サイズなどを選択します。
- Statusline

  ステータス行表示のオン／オフを切り替えます。
- Miscellaneous

  エラーおよびワーニング番号，ページ長，ページ幅を設定します。

注意　オープンしているすべてのウインドウのサイズ，位置およびすべての Options **メニュー設定は，**WB77016 **終了時に自動的に保存されます。**

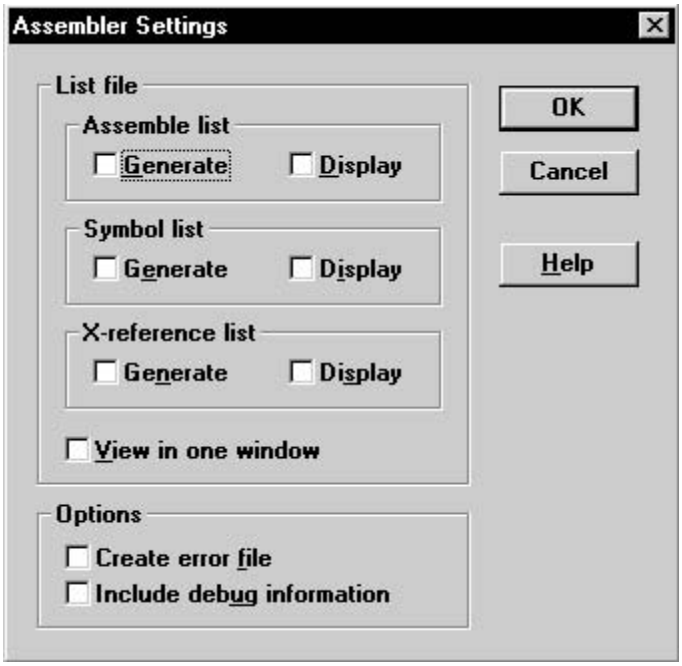
保守／廃止

6.7.1　Assembler...

このコマンドは，出力ファイル・タイプおよび WB77016 **アセンブラのリストを選択するための** Assembler **ダイアログを表示します。**

Assembler **ダイアログを図** 6－11 **に示します。**

図 6－11　Assembler **ダイアログ**



Assembler **ダイアログの構成要素**

Assemble list

ソース・コード（アセンブラ命令，データ割り当て文など）と生成したオブジェクト・コードが表示されます。“Generate”**をチェックすると，アセンブラ・リストがディスクに出力されます。**“Display”**をチェックすると，アセンブラ・リストが画面とディスクに出力されます（**“Generate”**が自動的にチェックされます）。**

Symbol list

“VALUE”，“ATTRIBUTE”，“RTYP”，“NAME”**などのユーザが定義したシンボルの構成要素が表示されます。**“Generate”**をチェックすると，シンボル・リストがディスクに出力されます。**“Display”**をチェックすると，シンボル・リストが画面とディスクに出力されます（**“Generate”**が自動的にチェックされます）。**

X-reference list

シンボル・リストの情報に加えて，そのシンボルが使用されているアドレスも記述されます。“Generate”**をチェックすると，クロスレファレンスがディスクに出力されます。**“Display”**をチェックすると，クロスレファレンスが画面とディスクに出力されます（**“Generate”**が自動的にチェックされます）。**

View in one window

“Display”をチェックすると，ファイルやリストが個別のウインドウに表示されます。出力を 1 つのウインドウにまとめて表示するには，“View in one window”をチェックします。“Display”設定はプロジェクト・ファイルに格納されますが，“View in one window”設定はプロジェクト・ファイルに格納されないので，特定のプロジェクトとは独立して適用されます。この設定は，アプリケーション・プロパティとして保存されます。

Include debug information

リロケータブル・アセンブラ出力ファイルにディバグ情報を付加するには“ASM”を選択し，ディバグ情報を不要にするには“NONE”を選択します。

Create error file

このコマンドをチェックすると，アセンブル時に発生したエラーが，エラー・タイプ，エラー番号，行番号情報などを含めてエラー・ファイルに収集されます。行番号情報は，エラーが発生したソース・コードの行を示します。

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，設定を無効にします

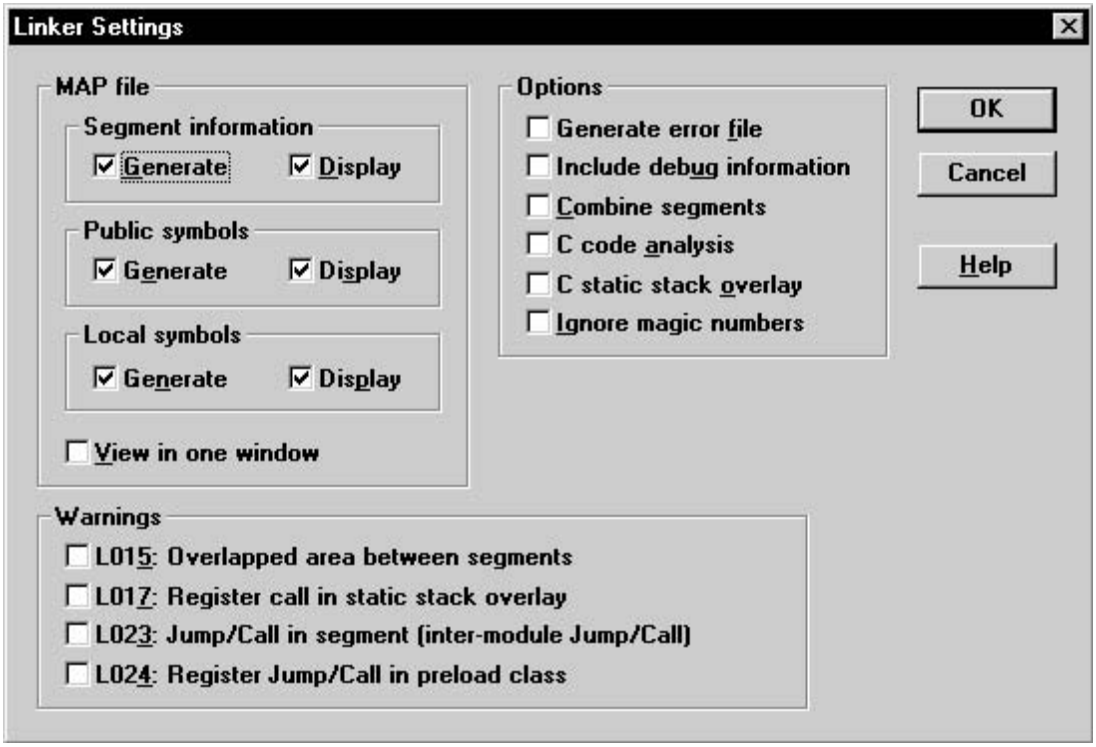
注意　オブジェクト・モジュール・ファイル（リロケータブル・アセンブラ出力ファイル）は，自動的に生成されます。

6.7.2　Linker Settings...

“Linker Settings…”コマンドは，出力ファイル・タイプ，出力オプションおよび WB77016 リンカのリストを選択するための Linker Settings ダイアログを表示します。

Linker Settings ダイアログを図 6－12 に示します。

図 6－12　Linker Settings **ダイアログ**



Linker Settings **ダイアログの構成要素**

Segment information

このコマンドをチェックすると，セグメント・リストが生成されます。セグメント・リストには，次の項目を含めたリンク環境の情報を持つヘッダが表示されます。

- **プロセッサ・モデル**
- **プロセッサ・タイプ**
- **メモリ領域**
- **開始および終了アドレス**
- **サイズおよびセグメント名**

Public symbols

このコマンドをチェックすると，パブリック・シンボル・リストが生成されます。パブリック・シンボル・リストには，次の項目が表示されます。

- **シンボルが入っているモジュールの名前**
- **シンボルの属性と値**
- **シンボルの名前**

Local symbols

このコマンドをチェックすると，ローカル・シンボル・リストが生成されます。ローカル・シンボル・リストには，次の項目が入っているモジュールの名前が表示されます。

・シンボル

・シンボルの属性と値

・シンボルの名前

View in one window

“Display”**をチェックすると，ファイルやリストが個別のウインドウに表示されます。出力を** 1 **つのウインドウにまとめて表示するには，**“View in one window”**をチェックします。**“Display”**設定はプロジェクト・ファイルに格納されますが，**“View in one window”**設定はプロジェクト・ファイルに格納されないので，特定のプロジェクトとは独立して適用されます。この設定は，アプリケーション・プロパティとして保存されます。**

Generate error file

このコマンドをチェックすると，次のエントリを含むテキスト・ファイルが生成されます。

・アセンブルの日付けと時刻

・アセンブルしたファイルの名前（パス名を含む）

・エラー・メッセージとエラーが発生した行番号

注意　エラー・ファイルはプロジェクト内の各ファイルに対して生成され，その名前はソース・ファイル名に拡張子“.ERR”**をつけたものになります。**

Include debug information

このオプションをチェックすると，アセンブラが生成したディバグ情報がリンカ出力ファイルに付加されます。

Combine segments

このコマンドをチェックすると，同じ名前と属性を持っているリロケータブル・セグメントにおいて定義されたすべてのデータとシンボルは 1 つの継続的なセグメントにコピーされます。新しい共通のセグメントは“__NEW__”**というモジュールの中で，元のセグメント名の下に結合されます。**

C code analysis

このオプションをチェックすると，ネスティングした C **関数呼び出しの数が** 15 **を越えているときに，ワーニング**“Warning L016: Call of function <name> from <name> exceeds limit 15 nested calls”**を出力します。**

C static stack overlay

静的スタック・セグメントは，C **言語関数からの自動変数（つまり，静的スタック・セグメントに静的に割り当てられた自動変数）を含むデータ・セグメントです。リンカがこのモードで動作している場合には，**C **関数呼び出しが分析され，自動変数を持たない関数の静的スタック・セグメントまたは同時に存在するスタック・フレームが，可能な場合は重複して割り当てられます。**

すべての C **言語関数呼び出しを正しく解決するには，次の命名規約を使用する必要があります。**

・関数“func”**のコードを含むインストラクション・セグメント名**:　“func_code”

・関数の開始アドレスのレーベル名:　“_func”

・関数の静的スタック・フレームを含むセグメント名:　“func_Xdata”，“func_Ydata”

Ignore magic numbers

マジック番号は,オブジェクト・ファイルとロード・モジュール XCOFF **ファイル内のターゲット** DSP **をエンコーディングします。**WB77016 **リンカは，マジック番号を使用して，プロジェクト内のすべてのファイルが同じターゲット・プロセッサに関連付けられているかどうかを検出します。異なるターゲット** DSP **に対して構築されたライブラリを使用できるようにするために，**Linker Settings **ダイアログの**“Ignore magic numbers”**ボックスをチェックして，マジック番号検出を無効にすることができます。**

L015 Overlapped area between segments

このオプションをチェックすると，領域重複の検出時にワーニングが出力されます。

L017 Register call in static stack overlay

このオプションをチェックすると，レジスタ呼び出しの検出時にワーニングが出力されます。

L023 Jump/Call in segment (inter-module Jump/Call)

このオプションをチェックすると，モジュール間ジャンプまたは呼び出し（あるプリロード・モジュール・クラスから別のプリロード・モジュール・クラスへの相対的なジャンプ／呼び出し）の検出時にワーニングが出力されます。モジュール間呼び出しの例を次に示します。

```
NAME MODULE1
i1 imseg at 0x8100; segment i1 in preload class pc1
call label1;

i2 imseg at 0x8110; segment i2 in preload class pc2
label1:
        ret;
end
```

上記のコード・ブロックを含むプロジェクトのリンクの際には，ワーニング‘ Warning L023: Jump/Call in segment MODULE1:i1, address 0x8100 to segment MODULE1:i2, address 0x8110”**が出力されます。**

L024 Register Jump/Call in preload class

このオプションをチェックすると，プリロード・クラス内のレジスタ・ジャンプまたは呼び出しの検出時にワーニングが出力されます。

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更された設定を無視します。

6.7.3　Edit Segments...

セグメント管理は，WB77016 **のリンカによって実行されます。セグメントの関連設定は，プロジェクト・ソース・コードで定義するか，**Options **メニューの**“Edit Segments…”**コマンドによるプロジェクト・アセンブルの後に定義するかのどちらかです。セグメント管理は，セグメントが共通のクラス属性を共有するためのメンバーシップを取得できるいくつかのセグメント・クラスを提供します。以降の節で説明するすべてのセグメント設定は，プロジェクト（またはその影響下にあるファイル）のリンク時に有効になります。**

6.7.4　Segment classes

リンカは，次のセグメント・クラスをサポートします。

- BOOT class:　μPD77016 **ファミリのブート関数をサポートします。**
- INI class: RAM **初期化のためにユーザ定義のセグメント名を提供します。**
- INID class: RAM **初期化のためにアセンブラ・ソース・ファイルに含まれているディフォールトのセグメント名をサポートします。これらのセグメント名は，**in_x_idata，ex_x_idata，in_y_idata，ex_y_idata，I_ROM **です。**
- OVL class: **セグメントのオーバレイ割り当てをサポートします。**
- Preload module class [注]: **プリロード・モジュール・クラス内のセグメントは，** μPD77116 **のキャッシュ可能なメモリに割り当てられます。**
- DMA class [注]: DMA **セグメント・クラスは，ディフォールトで別個の** DMA **メモリ・バンクに割り当てられる内部データ・セグメントのグループです。**

注　μPD77116 **のみサポートします。**

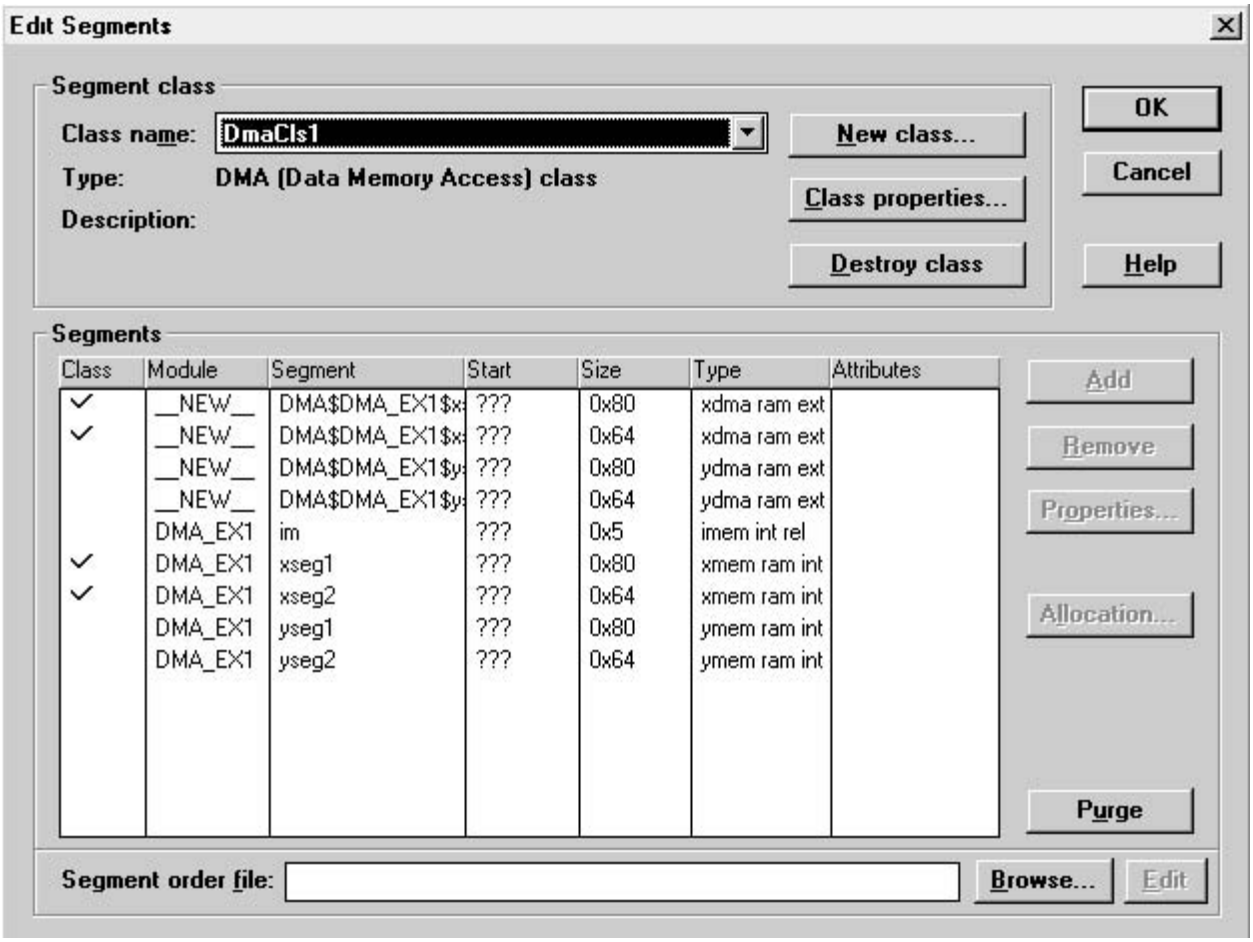
6.7.5　Edit Segments **ダイアログ**

セグメントとセグメント・クラスの変更は，Options **メニューの**“Edit Segments…”**コマンドで実行します。**Edit Segments **ダイアログを使用して，セグメント属性をクラス・メンバーシップとセグメント割り当ての面から表示，変更します。**

Edit Segments **ダイアログを図** 6－13 **に示します。**

保守／廃止

図 6－13　Edit Segments ダイアログ



Edit Segments **ダイアログの構成要素**

Class name

このリスト・ボックスは，現在オープンされているプロジェクトのクラス名を表示します。現在選択されているクラスのメンバであるすべてのセグメントは，セグメント・リストの Class **カラムにチェック・マーク（✓）がついて示されます。**

Type

現在選択されているクラスの種類を表示します（BOOT，INI，INID，Overlay，Preload Module，DMA**）。**

Description

このフィールドは，ユーザが入力した追加情報を表示します。

New class...

このボタンは，次の種類のセグメント・クラスを新規に作成する New Class **ダイアログを表示します。**

- BOOT
- INI
- INID
- Overlay
- Preload Module
- DMA

Class properties...

このボタンは，現在選択されているクラスを参照する properties **ダイアログを表示します。**

- Boot Class properties **ダイアログ**
- RAM Initialization class properties **ダイアログ**
- Overlay Class properties **ダイアログ**
- Preload Module Class properties **ダイアログ**
- DMA Class **ダイアログ**

Destroy class

現在選択されているクラス名（Class name **フィールドに示されている）をクラス名リストから削除します。対応するセグメント・クラスが破壊されると，すべてのクラス・メンバがキャンセルされます。**

Segments

このリストは，各セグメントについて次の情報を表示します。

- Class: Class name **フィールドで現在選択されているクラスのメンバであるすべてのセグメントは，チェック・マーク（✓）で示されます。**
- Module: **セグメントが定義されるモジュールの名前。**
- Segment: **セグメント名。**
- Start: **セグメントの開始アドレス。リロケータブル・セグメントは，**“???”**で示されます。**
- Size: **セグメントのサイズ。長さが不定のセグメントは，**“???”**で示されます。セグメント長はリンク時に決定されます。**
- Type:**メモリの配置属性**

  Imem，xmem，ymem，RAM，または ROM。**内部または外部。絶対，再配置可能，位置指定，または整列。**
- Attributes:RAM **初期化セグメント用のデータ，またはゼロ。非オーバレイ・グループ名。さらに，色の違いによって次の機能が示されます。**

  青：選択したクラスに既に属しているセグメント。

  濃紺：クラスに自動的に追加されるセグメント（たとえば，ブート・クラス，ペア・セグメントなどのディフォールト・メンバ）。

  緑：選択したクラスに追加できるセグメント。

  淡色：現在未使用のセグメントの属性を示します。

  黒：選択したクラスに追加できないセグメント。

Add

セグメント・リストの中の選択したセグメントを Class name フィールドに表示されているクラスに追加します。Class カラムにチェック・マーク（✓）が表示され，セグメントが現在選択されているクラスのメンバであることを示します。セグメントが特定のクラスのメンバである場合，そのセグメント・クラスのすべての属性を継承します。Add ボタンは，選択したセグメントが現在選択されているクラスのメンバになっていない場合にのみ有効です。セグメントをクラスに追加する際に，次の制約があることに注意してください。

・ブート・クラスにはインストラクション・メモリ・セグメントしか追加できません。

・RAM 初期化(INI，INID)クラスには X RAM セグメントまたは Y RAM セグメントしか追加できません。

Remove

選択したセグメントを Class name フィールドに表示されているクラスから削除します。このボタンは，選択したセグメントが Class name フィールドで現在選択されているクラスのメンバである場合にのみ使用可能になります。

Properties

現在選択されているセグメントの Properties ダイアログを表示します。セグメントがオーバレイ・クラスのメンバである場合，Overlapped Segment ダイアログを使用して，そのセグメントを非重複グループのメンバとして指定します。セグメントが RAM 初期化クラスのメンバである場合，Initialized Segment ダイアログを使用して，セグメント初期化タイプを設定します。

Allocation...

セグメントの割り当ては，Segment Attributes ダイアログで調整します。セグメント割り当てを変更するには，Segment List でそれを選択してから，Allocation…ボタンをクリックして Segment Attributes ダイアログを表示します。

Purge

セグメント・リストでは，淡色は現在未使用のセグメントの属性を示します。このような属性は，属性をセグメントに割り当てた後に，セグメントの名前を変更するか削除した場合に発生することがあります。これらの設定を削除したい場合，Purge ボタンを使用します。

Segment order file

Segment order file は，DSP のメモリ中のセグメントの割り当て順序を指定します。

Browse

Edit Segments ダイアログを使用すると，Segment order file を閲覧し選択することができます。

Edit

選択した Segment order file を編集します。

OK

ダイアログをクローズし，変更を受け入れます。

Cancel

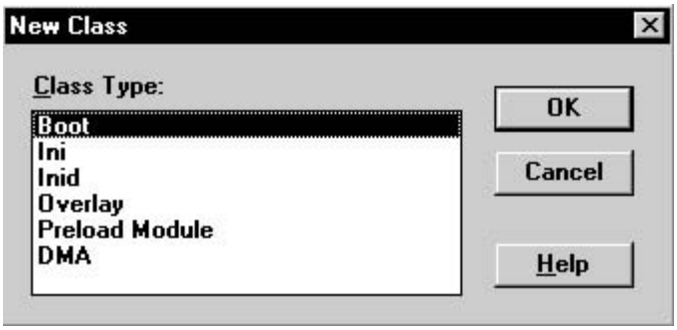
ダイアログをクローズし，変更を廃棄します。

6.7.6　New Class ダイアログ

このダイアログを使用して，作成するセグメントのタイプ・クラスを選択します。

New Class **ダイアログを図** 6－14 **に示します。**

図 6－14　New Class **ダイアログ**



New Class **ダイアログの構成要素**

Class Type

Class Type **は，次の使用可能なクラス・タイプを表示します。**

・BOOT:　μPD77016 **ファミリのブート関数をサポートします。**

・INI: **この** RAM **初期化クラスは，ユーザ定義のセグメント名を提供します。**

・INID: **この** RAM **初期化クラスは，アセンブラ・ソース・ファイルに含まれているディフォールトのセグメント名を使用します。**

・Overlay: **所定のクラスのメンバであるセグメントの重複割り当てを可能にします。**

・Preload Module [注]: **プリロード・モジュール・クラス内のセグメントは，**μPD77116 **のキャッシュ可能なメモリに割り当てられます。**

・DMA [注]: DMA **セグメント・クラスは，ディフォールトで別個の** DMA **メモリ・バンクに割り当てられる内部データ・セグメントのグループです。**

注　μPD77116 **のみ使用できます。**

OK

ダイアログをクローズし，選択したクラスの Properties **ダイアログを表示してクラス属性を設定します。**

Cancel

クラスを新規に作成せずにダイアログをクローズします。

保守／廃止

6.7.7　Boot Class ダイアログ

このダイアログを使用して，選択したブート・クラス・セグメントの属性を変更したり，表示したりします。ブート・クラス・セグメントは，次のμPD77016 ファミリのブート関数をサポートします。

- Reset self-boot:

  リンカは，ブート・パラメータを含むセグメントをアドレス 0x4000:Y に追加し，指定したセグメント（該当するブート・クラスのメンバ）のコピーを指定した X データ・メモリまたは Y データ・メモリに作成します。

- Reset host-boot:

  リンカは，ブート・パラメータおよび指定したセグメント（該当するブート・クラスのメンバ）のコピーを指定した HEX ファイルに出力します。

- Self reboot:

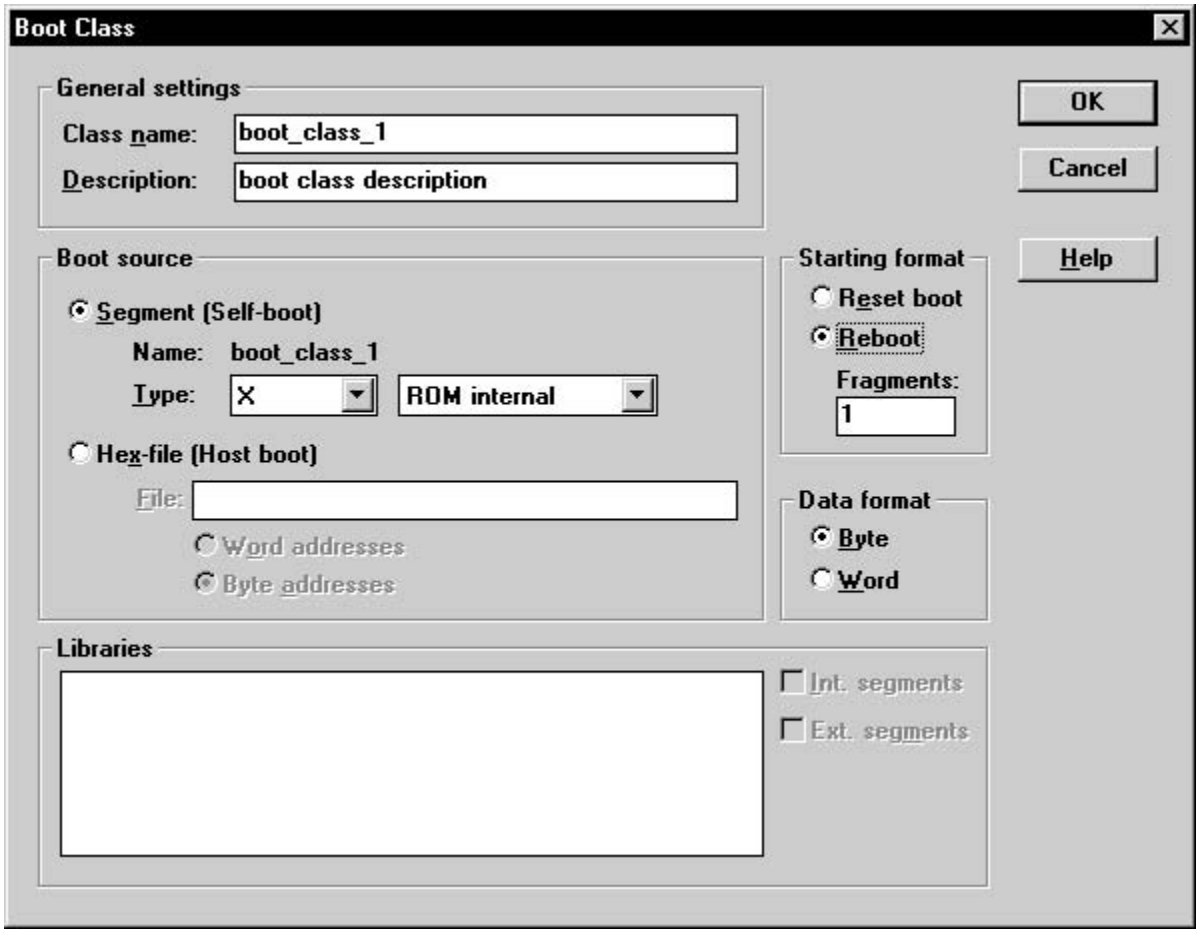
  リンカは，ブート・パラメータを含むパブリック・シンボルをエクスポートし，指定したセグメント（該当するブート・クラスのメンバ）のコピーを指定したデータ・メモリに作成します。

- Host reboot:

  リンカは，ブート・パラメータを含むパブリック・シンボルをエクスポートし，指定したセグメント（該当するブート・クラスのメンバ）のコピーを指定した HEX ファイルに作成します。

Boot Class ダイアログを図 6－15 に示します。

図 6－15　Boot Class ダイアログ



保守／廃止

Boot Class **ダイアログの構成要素**

Class name

このフィールドには，ユーザが入力するブート・クラスの名前が入ります。クラス名には空白を入れられないことに注意してください。

Description

ユーザが入力する追加のクラス情報。このフィールドは，ブート・クラスに注釈を追加するためのものなので，リンカの処理の対象ではありません。

Segment (Self-boot)

このオプションを選択して，セルフ・ブートを現在のプロジェクトのブート・ソースとして設定します。読み取り専用の Name **フィールドには，現在変更されているブート・クラスのセグメント名が表示されます。ブート・データをインストラクション・メモリに転送する元になるデータ・メモリ・タイプは，**X **データ・メモリまたは** Y **データ・メモリのどちらかです。ブート・データ・セグメントは，内部** ROM **または外部** ROM **に置くことができます。**

Hex file (Host-boot)

このオプションを選択して，ホスト・ブートを現在のプロジェクトのブート・ソースとして設定します。ブート実行時にインストラクション RAM **を初期化するのに使用されるホスト・ブート・データは，**File **フィールドに指定した** HEX **ファイルに格納されます。指定した** HEX **ファイルは，リンク中に生成され，ユーザ指定の出力ディレクトリに置かれるか，または現在のプロジェクト・ファイルがあるディレクトリに置かれます。リンク中にファイルを生成できない場合，エラー・メッセージが表示されます。**

Word addresses

このオプションを指定すると，リンカは HEX **ファイルをワード・アドレス・フォーマットで出力します。**

Byte addresses

このオプションを指定すると，リンカは HEX **ファイルをバイト・アドレス・フォーマット（**HEX **フォーマットとも呼ばれる）で出力します。**

Flagments

Starting format **でリブートを選択した場合のみ有効になります。通常はディフォールト設定**“1”**のまま使用します。**μ PD77110 **のように命令** RAM **が** 2 **領域以上あるデバイスを使用する場合に，**1 **つのリブート・クラスに指定したブート・コード・サイズが** 1 **つの** RAM **領域サイズを越える場合は，**Flagment **を**“2”**に変更すると，ブート・コードを別の** RAM **領域へ分けて配置することができます。**

Starting format

Reset boot **オプションを選択すると，ブート・パラメータをアドレス** 0x4000:Y **で始まるセグメントに設定します。**Reboot **オプションを選択すると，ブート・パラメータをパブリック・シンボルとしてプロジェクトにエクスポートします。**

Libraries

プロジェクトで使用可能なライブラリのリストを表示します。設定に従い，内部セグメント/外部セグメント，選択したライブラリに含まれる割り当てセグメントがブート・クラスに追加されます。

Data format

Byte **を選択すると，各命令（**4 **バイト）は** 4 **つのデータ・メモリ・アドレス（**4 **データ・メモリ・ワード）にコピーされます。**Word data format **を選択すると，各命令は** 2 **つのデータ・メモリ・アドレス（**2 **データ・メモリ・ワード）にコピーされます。**

Int. segments (i.e. Add All Internal RAM Segments)

このオプションをチェックすると，内部インストラクション・メモリに割り当てられている選択ライブラリのすべてのセグメントがブート・クラスに追加されます。

Ext. segments (i.e. Add All external RAM Segments)

このオプションをチェックすると，外部インストラクション・メモリに割り当てられている選択ライブラリのすべてのセグメントがブート・クラスに追加されます。

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更を無視します。

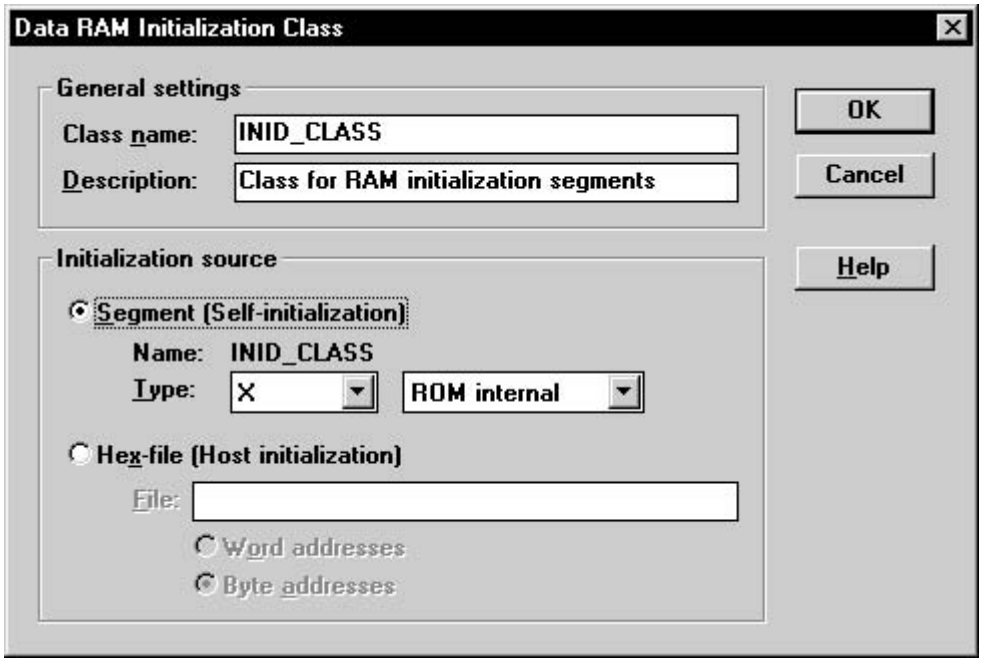
保守／廃止

6.7.8　Data RAM Initialization Class ダイアログ

このダイアログを使用して，選択した初期化（INI または INID）クラスの属性を設定または表示します。

Data RAM Initialization Class ダイアログを図 6－16 に示します。

図 6－16　Data RAM Initialization Class ダイアログ



Data RAM Initialization ダイアログの構成要素

Class name

このフィールドには，ユーザが入力する INI または INID クラスの名前が入ります。クラス名は空白にできないので注意してください。INID クラスの場合，クラス名は“INID_CLASS”に設定され，ユーザは変更することができません。

Description

ユーザが入力する追加のクラス情報です。このフィールドは，初期化クラスに注釈を追加するためのものなので，リンカの処理の対象ではありません。

Segment (Self-initialization)

このオプションを選択して，セルフ初期化を設定します。

セグメント名とセグメント・タイプを選択してください。

Hex file (Host initialization)

このオプションを選択して，ホスト初期化を設定します。

Word addresses

このオプションを指定すると，リンカは HEX ファイルをワード・アドレス・フォーマットで出力します。

Byte addresses

このオプションを指定すると，リンカは HEX **ファイルをバイト・アドレス・フォーマット（**HEX **フォーマットとも呼ばれる）で出力します。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

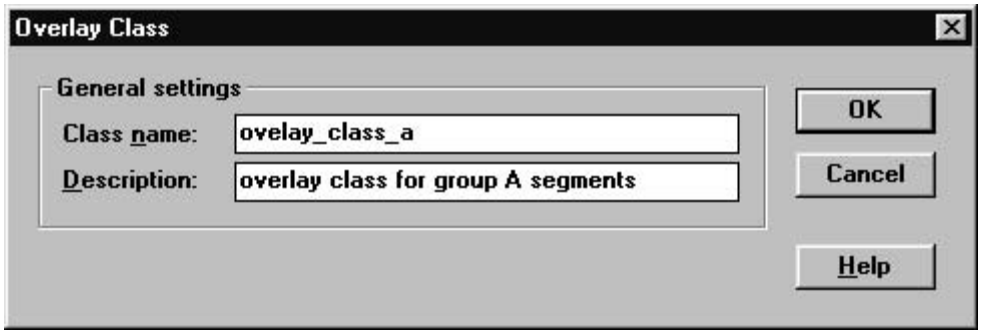
ダイアログをクローズし，変更を無視します。

6.7.9　Overlay Class **ダイアログ**

このダイアログを使用して，オーバレイ・クラスを指定します。オーバレイ・クラスのメンバであるセグメントは，データ・メモリに重複して割り当てることができます。このことは，複数のセグメントで構成される重複スタックの場合，これらのセグメントが同じメモリ領域を要求できることを意味します。これらのセグメントを非重複グループに追加して，特定のセグメントをお互いに重複しないようにすることができます（6.7.12 Segment Overlay Attributes **ダイアログ参照）。**

Overlay Class **ダイアログを図** 6－17 **に示します。**

図 6－17　Overlay Class **ダイアログ**



Overlay Class **ダイアログの構成要素**

Class name

このフィールドには，ユーザが入力するオーバレイ・クラスの名前が入ります。クラス名には空白にできないこと，および 30 **文字を越えてはならないことに注意してください。**

Description

ユーザが入力する追加のクラス情報。このフィールドは，オーバレイ・クラスに注釈を追加するためのものなので，リンカの処理の対象ではありません。

保守／廃止

オーバレイ・セグメント・クラスの例

セグメントの非オーバレイ・グループを持つオーバレイ・クラスの宣言例を次に示します。

・開始条件:

・オーバレイ・クラスのメンバは，seg1**，**seg2**，**seg3**，**seg4**，**seg5 **です。全セグメントは再配置可能で，サイズが** 5 **ワードとします。**

・非オーバレイ・グループ“stack_1”**のメンバは，**seg1**，**seg2 **です。**

・非オーバレイ・グループ“stack_2”**のメンバは，**seg4**，**seg5 **です。**

・アドレス 0 **で始まる空のメモリ領域がある場合，リンカはこれらのセグメントを次のように割り当てます。**

seg1 – **開始アドレス** 0

seg2 – **開始アドレス** 5

seg3 – **開始アドレス** 0

seg4 – **開始アドレス** 0

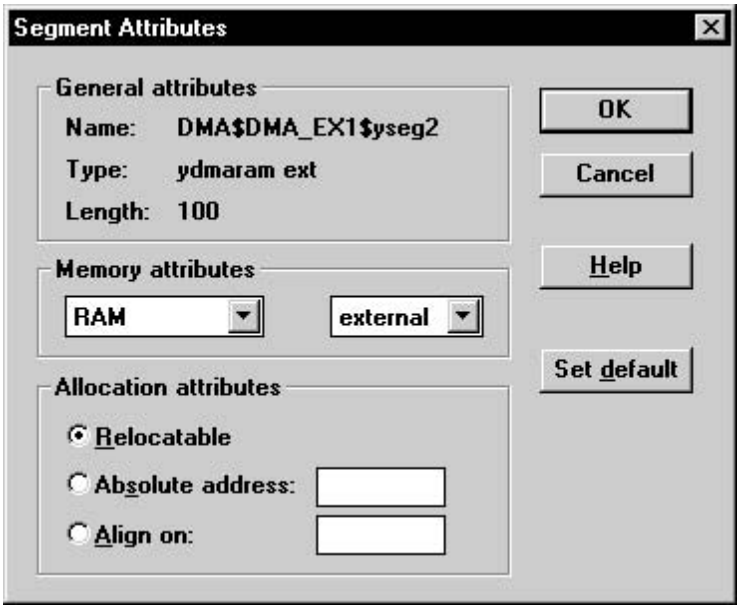
seg5 – **開始アドレス** 5

6.7.10　Segment Attributes **ダイアログ**

このダイアログを使用して，セグメント割り当て属性を変更したり，表示したりします。

Segment Attributes **ダイアログを図** 6－18 **に示します。**

図 6－18　Segment Attributes **ダイアログ**



Segment Attributes **ダイアログの構成要素**

Name

このフィールドは，セグメント名を示します。

Type

このフィールドは，セグメント・タイプを示します。

Length

このフィールドは，セグメント・サイズを示します。

Memory attributes

・セグメントを割り当てるメモリのタイプ，およびターゲット・メモリ領域の割り当て順序を指定します。

RAM **と** ROM **の選択は次のとおりです。**

RAM: **セグメントは** RAM **に割り当てられます。**

ROM: **セグメントは** ROM **に割り当てられます。**

RAM, ROM: **最初にリンカはセグメントを** RAM **に割り当てようとします。**RAM **への割り当てができない場合，リンカはセグメントを** ROM **に割り当てようとします。**

ROM, RAM: **最初にリンカはセグメントを** ROM **に割り当てようとします。**ROM **への割り当てができない場合，リンカはセグメントを** RAM **に割り当てようとします。**

RAM or ROM: **所定の割り当て順序はありません。セグメントを** RAM **か** ROM **のどちらかに割り当てられます。**

・ 内部と外部の選択:

internal: **セグメントは内部メモリに割り当てられます。**

external: **セグメントは外部メモリに割り当てられます。**

int, ext: **最初にリンカはセグメントを内部メモリに割り当てようとし，次に外部メモリに割り当てようとします。**

ext, int: **最初にリンカはセグメントを外部メモリに割り当てようとし，次に内部メモリに割り当てようとします。**

int or ext: **所定の割り当て順序はありません。セグメントを内部メモリか外部メモリのどちらかに割り当てられます。**

Relocatable

リロケータブル・セグメントを指定するには，このボタンをチェックします。リロケータブル・セグメントには，Edit Segments **ダイアログの** Segment List **の割り当てカラムに**“rel”**エントリがあります。リロケータブル・セグメントは，リンクの際に配置されます。**

Absolute Address

配置されたセグメントを指定するには，このボタンをチェックします。入力フィールドにロケーションの開始アドレスを入力するように促されます。配置されたセグメントには，Edit Segments **ダイアログの** Segment List **の割り当てカラムに “** loc”**エントリがあります。**

Aligned on

ファイル・セグメントに整列境界を指定するには，このボタンをチェックします。入力フィールドに 0-0xFFFF **の境界サイズを入力するように促されます。指定された境界が** 0 **ならば，セグメントは，そのサイズより大きい次の** 2 **の累乗のところに整列します。整列したセグメントには，**Edit Segments **ダイアログの** Segment List **の割り当てカラムに**“aln”**エントリがあります。**

Set default

変更されたセグメントをそのディフォールト・プロパティに復元するには，このボタンをクリックします。

OK

ダイアログをクローズし，セグメント属性の設定を受け入れます。

Cancel

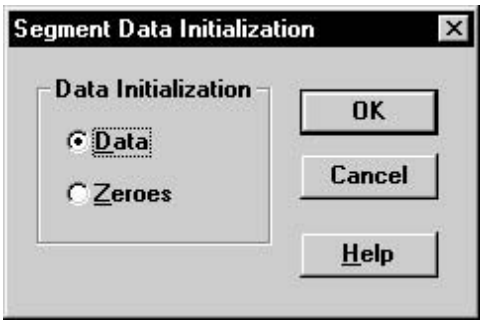
ダイアログをクローズし，セグメント属性の変更を廃棄します。

6.7.11　Segment Data Initialization ダイアログ

このダイアログを使用して，初期化クラス固有のセグメント・パラメータを設定します。

Segment Data Initialization **ダイアログを図** 6－19 **に示します。**

図 6－19　Segment Data Initialization **ダイアログ**



Segment Data Initialization **ダイアログの構成要素**

Data

対応するセグメントをデータ値で初期化します。データ値は，ソース・ファイルで定義し，所定のクラスのメンバである該当 ROM **セグメントから実行時にコピーしなければなりません。**

Zeroes

実行時に対応するセグメントをゼロで初期化します。

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更を無視します。

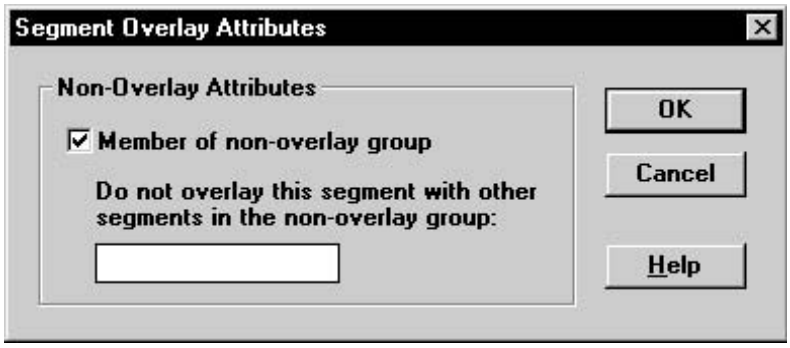
保守／廃止

6.7.12　Segment Overlay Attributes ダイアログ

このダイアログを使用すると，オーバレイ・クラス固有のセグメント・パラメータを設定できます。

Segment Overlay Attributes **ダイアログを図** 6－20 **に示します。**

図 6－20　Segment Overlay Attributes **ダイアログ**



Segment Overlay Attributes **ダイアログの構成要素**

Member of non-overlap group

このオプションを選択して，同じ（非重複）グループのメンバであるセグメントの重複を回避します。

Do not overlay this segment with other segments in the non-overlay group

このフィールドに目的の非重複セグメントの ID **名を入力します。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更を無視します。

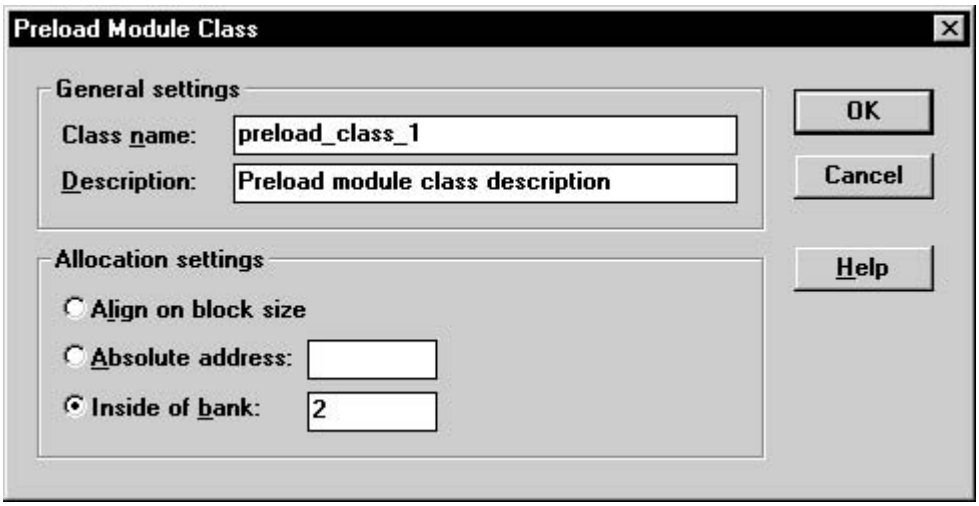
保守／廃止

6.7.13　Preload Module Class ダイアログ

Preload Module Class ダイアログで，プリロード・モジュール・セグメントのクラス属性を設定したり，表示したりできます。このクラスはμPD77116 のみサポートされます。

Preload Module Class ダイアログを図 6－21 に示します。

図 6－21　Preload Module Class ダイアログ



Preload Module Class ダイアログの構成要素

Class name

このフィールドには，ユーザが入力するプリロード・モジュール・クラスの名前が入ります。クラス名には空白を入れられないこと，および 30 文字を越えてはならないことに注意してください。

Description

ユーザが入力する追加のクラス情報。このフィールドは，プリロード・モジュール・クラスに注釈を追加するためのものなので，リンカの処理の対象ではありません。

Aligned block on size

プリロード・モジュール・クラスの最初のセグメントは，ブロック・サイズに基づいて整列して割り当てられ（セグメントはブロックの先頭から始まる），空き領域のその他のリロケータブル・セグメントは，すでに配置されたセグメントの後に置かれます。

Absolute address

プリロード・モジュール・クラスの最初のセグメントは，指定された開始アドレスに割り当てられ，空き領域のその他のリロケータブル・セグメントは，既に配置されたセグメントの後に置かれます。

Inside of bank

プリロード・モジュール・クラスのすべてのセグメントは，指定されたバンクの内部に割り当てられます。

OK

ダイアログをクローズし，設定を受け入れます。

保守／廃止

Cancel

ダイアログをクローズし，変更を無視します。

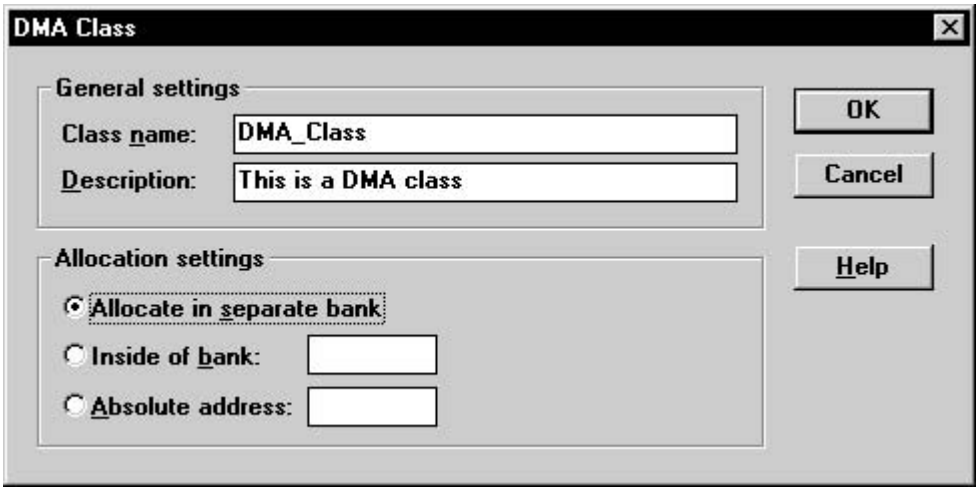
6.7.14　DMA Class ダイアログ

DMA セグメント・クラスは，内部データ・メモリに重複して割り当てられる内部データ・セグメントのグループです。リンク中には，あるクラスの中の各内部データ・セグメントのペア・セグメント（コピー・セグメント）は，外部データ・メモリに割り当てられます。ペア・セグメントの名前は<seg>_P であり，ペア・セグメントの開始アドレスはパブリック・シンボル_<seg>_P にエクスポートされます。内部メモリ・セグメントとそのペア・セグメントとの間で，DMA によってデータを転送することができます。異なる DMA クラスに属する重複した内部メモリ・セグメントは，異なるバンクに割り当てられます。したがって，1 つのクラスからセグメントにアクセスし，同時にほかのクラスとの間でセグメントを送受信することができます。

DMA Class ダイアログを図 6－22 に示します。

この DMA Class ダイアログはμPD77116 のみサポートします。

図 6－22　DMA Class ダイアログ



DMA Class ダイアログの構成要素

Class name

このフィールドには，ユーザが入力する DMA クラスの名前が入ります。クラス名は空白にできないこと，および 30 文字を越えてはならないことに注意してください。

Description

ユーザが入力する追加のクラス情報。このフィールドは，プリロード・モジュール・クラスに注釈を追加するためのものなので，リンカの処理の対象ではありません。

Allocate in separate bank

ディフォールトの設定です。ほかの DMA クラス，またはほかのセグメントが配置されているバンクとは異なるバンクに配置します。

Inside of bank

ほかの DMA **クラス，またはほかのセグメントが配置されているバンクと同じバンクに配置します。**

Absolute address

DMA **クラスを指定した絶対アドレスに配置します。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更を無視します。

6.7.15　Processor Model...

このコマンドは，ターゲットのプロセッサ・タイプ，クロック周波数，メモリ仕様を記述したプロセッサ・モデルを選択するための Select Model **ダイアログを表示します。プロセッサ・モデルは，モデル定義ファイルに格納されています。この定義ファイルは，生のテキスト形式のテキスト・ファイルです。このダイアログを使用して，前のモデル定義ファイルをオープンし，変更することもできます。**

ID77016 **が** Select Model **ダイアログを表示した場合には，このダイアログにはディバグ・デバイスを選択できる**“Connected to”**フィールドが含まれています。**

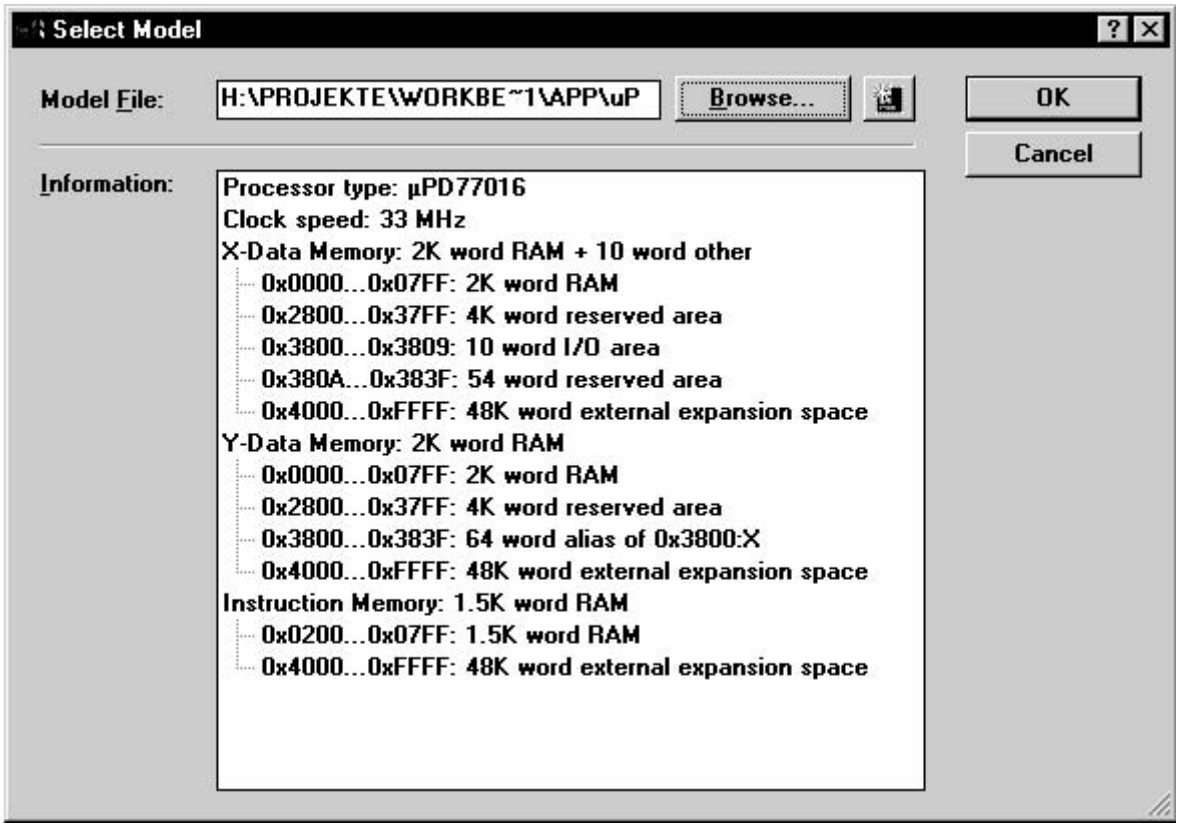
複数の JTAG **デバイスをサポートするドライバ**[注]**が** ID77016 **と接続されている場合，**Select Model **ダイアログには，追加のコントロール・ボックス（**“Configuration”**と**“Unit identifier”**）があり，それぞれを使用して** HSDL(Hierarchical Scan Data Language)**ファイルを指定したり，**JTAG **ユニット** ID **を選択したりできます。マルチ** JTAG **デバイス・ドライバと必要な** HSDL **ファイル・エントリの詳細については，**Atair **社の** Multi Device JTAG Extension **マニュアルを参照してください。**

Select Model **ダイアログを図** 6－23 **に示します。**

注　開発中

保守／廃止

図6ー23　Select Model ダイアログ



Select Model **ダイアログの構成要素**

Model File

このフィールドは，現在使用中のモデル・ファイルのパスと名前を表示します。モデル・ファイルまたはモデル定義ファイルは，完全に記述されたパスとファイル名を入力して指定することができます。指定したファイルの内容は，フォーカス移動時に“Information”**フィールドに表示されます。指定したファイルを見つけることができない場合，**“Information”**フィールドに該当するヒントが表示されます。**

Browse

モデル（定義ファイル）がないか，目的のモデルが定義ファイルの中にない場合，別のモデル（定義ファイル）を選択する必要があります。ファイル選択用のダイアログが表示されます。

Model Wizard

このボタンを押すと，適用していれば Model Wizard **を起動する次のサブ・コマンドが表示されます。**

New...

Model Wizard **を起動し，新しいモデルを作成します。これを選択すると，任意の**μ PD77016 **ファミリに基づいて完全に新しいモデルが作成されます。ほかのすべてのモデル・プロパティはユーザが入力します。**Model Wizard **が終了すると，ファイル選択ダイアログによってモデル名と保管場所を指定するように促されます。**

Clone...

Model Wizard を起動し，現在アクティブなモデルのクローンを作成します。クローン・モデルは，ユーザが属性を変更していない限り，その元になるモデルのすべての属性を継承します。Model Wizard が終了すると，ファイル選択ダイアログによってモデル名と保管場所を指定するように促されます。

Edit...

Model Wizard を起動し，現在アクティブなモデルを編集します。現在のモデルは，変更後のモデルで置き換えられます。

Information

現在選択されているモデルの概要を表示します。表示される情報の範囲は，モデルごとに変わります。それぞれの概要には，少なくとも選択されたプロセッサ・タイプとクロック速度に関する情報があります。補足情報としては，外部メモリやほかの重要な情報があります。このリストは参考用であり，書き込むことはできません。対応するメモリ領域が表示されている行をダブル・クリックして，メモリ・マップ表示を拡張したり，消滅させたりできます。コンフィギュレーション・リストを拡張表示すると，領域，内部／外部メモリ領域，エイリアス，I/O，および予約メモリ範囲の開始／終了アドレスとメモリ・ワード数が表示されます。

Select Model ダイアログのサイズは変更可能であり，“Information”フィールドに表示されるデータ量に応じて調整することができます。サイズ変更はダイアログの右下の角で操作できます。

OK

ダイアログをクローズし，現在のモデルを適用します。

Cancel

ダイアログをクローズし，操作をキャンセルします。前に選択したモデルがそのまま有効です。

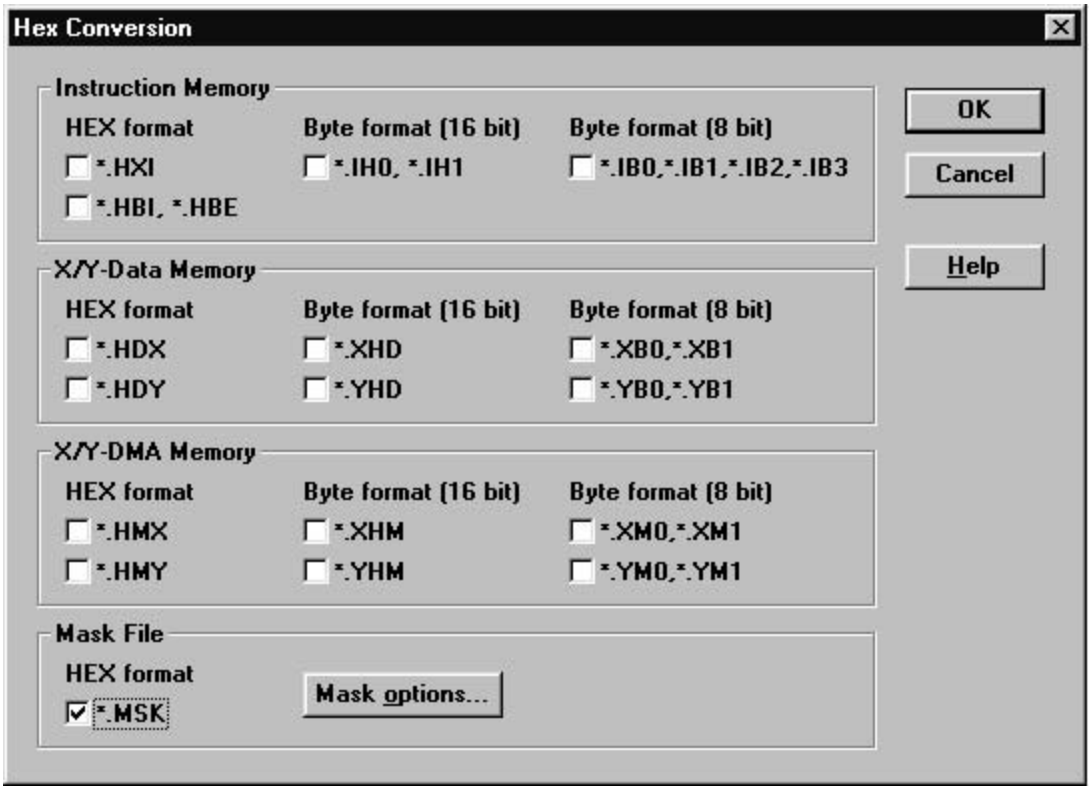
保守／廃止

6.7.16　Hexconversion...

このコマンドは，16 進数出力ファイル・タイプを選択する Hex Conversion **ダイアログを表示します。**

Hex Conversion **ダイアログを図** 6－24 **に示します。**

図 6－24　Hex Conversion **ダイアログ**



Hex Conversion **ダイアログの構成要素**

HEX format

コントロール・グループの次のボックスをチェックします。

*.HXI: **インストラクション・メモリのデータを変換するときチェック。**

*.HBI: **ブート用の内部インストラクション** RAM **のデータを変換するときチェック。**

*.HBE: **ブート用の外部インストラクション** RAM **のデータを変換するときチェック。**

*.HDX: X **メモリのデータを変換するときチェック。**

*.HDY: Y **メモリのデータを変換するときチェック。**

*.HMX: DMA X **メモリのデータを変換するときチェック。**

*.HMY: DMA Y **メモリのデータを変換するときチェック。**

*.MSK: **マスク・ファイルの生成。マスク・ファイルには，内部インストラクション** ROM，X **データ** ROM，Y **データ** ROM **領域のコピーがあります。**

Byte format (16 bit)

コントロール・グループの次のボックスをチェックします。

*.IH0, *.IH1: **インストラクション・メモリのデータを変換するときチェック。**

*.XHD: X **メモリのデータを変換するときチェックするときチェック。**

*.YHD: Y **メモリのデータを変換するときチェックするときチェック。**

*.XHM: DMA X **メモリのデータを変換するときチェックするときチェック。**

*.YHM: DMA Y **メモリのデータを変換するときチェックするときチェック。**

Byte format (8 bit)

コントロール・グループの次のボックスをチェックします。

*.IB0, *.IB1, *.IB2, *.IB3: **インストラクション・メモリのデータを変換するときチェック。**

*.XB0, *.XB1: X **メモリのデータを変換するときチェック。**

*.YB0, *.YB1: Y **メモリのデータを変換するときチェック。**

*.XM0, *.XM1: DMA X **メモリのデータを変換するときチェック。**

*.YM0, *.YM1: DMA Y **メモリのデータを変換するときチェック。**

Mask Options

Mask Settings **ダイアログ（図** 6－25 **参照）を表示し，クロック設定を選択します。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，設定を廃棄します。

HEX **変換ファイルの詳細については，第** 3 **章　ファイル・フォーマット**を参照してください。

6.7.17　Mask Settings **ダイアログ（**μ PD7701x **ファミリ用）**

このダイアログでは，次のデバイスのマスク・オプションの設定が促されます。

μ PD77015

μ PD77017

μ PD77018

μ PD77018A

μ PD77019

マスク設定は，マスク・ファイルを作成しようとする場合には必須です。したがって，このダイアログでは，何のディフォールト設定も提供されません。逓倍率 1/3 **を**μ PD77015**，**77017**，**77018 **に対して設定することはできません。**μ PD77018A**，**77019 **に対しては，次の図に示す** 5 **番目の逓倍率設定値** 1/3 **が設定できます。**

Mask Settings **ダイアログを図** 6－25 **に示します。**

保守／廃止

図 6－25　Mask Settings ダイアログ（μPD7701x ファミリ用）

Mask Settings ダイアログの構成要素

Clock selection

μPD77015，77017，77018，77018A，77019 は PLL 回路を内蔵しているため，内部動作周波数よりも低周波のクロックが選択できます。μPD77015，77017，77018 に対する逓倍率は，1/1，1/2，1/4，1/8 が選択できます。さらに，μPD77018A，77019 については，1/3 のクロック選択ができます。

Clock output

“Allowed”をチェックしておくと，CLKOUT ピンが使用可能になります。

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，設定を廃棄します。

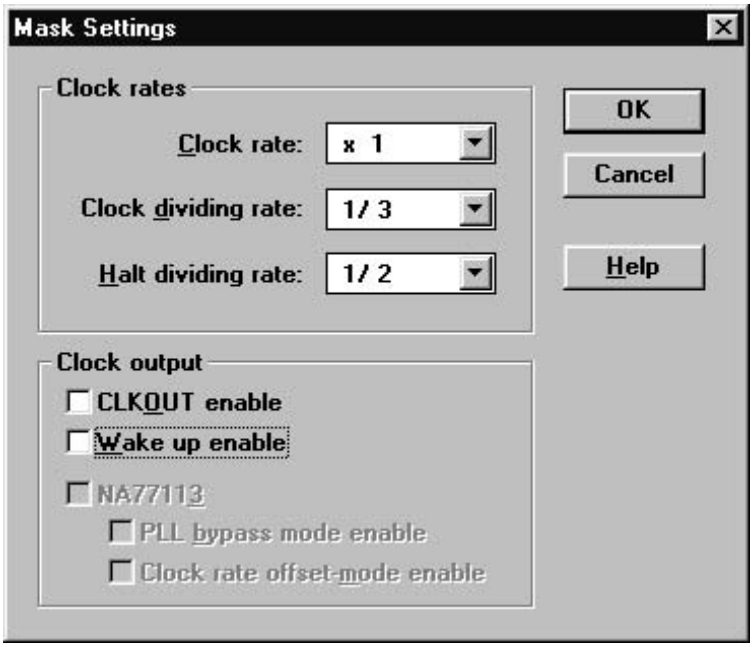
6.7.18　Mask Settings ダイアログ （μPD77111 ファミリ，NA77113 用）

μPD77111 ファミリ用の Mask Settings ダイアログでは，μPD77111，77112，77113，77114，NA77113 などのプロセッサ・タイプのマスク・オプションの設定が促されます。

μPD77111 ファミリ，NA77113 用の Mask Settings ダイアログを図 6－26 に示します。

保守／廃止

図 6－26　Mask Settings **ダイアログ（**μ PD77111 **ファミリ，**NA77113 **用）**



μ PD77111 **ファミリ，**NA77113 **用の** Mask Settings **ダイアログの構成要素**

Clock rate

係数 1-16 **を設定できます。**

Clock dividing rate

このレートは 1/1-1/16 **まで設定できます。**

Halt dividing rate

このレートは“stopped”**にするか，**1/2-1/16 **まで設定できます。**

CLOCKOUT enable

このボックスをチェックすると，クロック出力が有効になります。

Wake up enable

このボックスをチェックすると，ウェイクアップ機能が有効になります。

NA77113 [注]

このオプションは，現在ロードされているモデルのタイプがμ PD77113，77114 **の場合にのみ有効です。この種のモデルを現在ロードしているならば，これらのプロセッサ・タイプのマスク・オプションを指定することができ，次のオプションが有効になります。**

注　NA77113 （DSP **コア）開発時にのみ有効となります。**μ PD77113，77114 **ではチェックしないでください。**

PLL bypass mode enable [注]

このオプションは，現在ロードされているモデルのタイプがμPD77113，77114 の場合にのみ有効です。このオプションが設定されているならば，PLL バイパス・モードが有効になります。

注　NA77113 （DSP コア）開発時にのみ有効となります。μPD77113，77114 **ではチェックしないでください。**

Clock rate offset-mode enable [注]

このオプションは，現在ロードされているモデルのタイプがμPD77113，77114 の場合にのみ有効です。このオプションが設定されているならば，クロック・レート・オフセット・モードが有効になります。

注　NA77113 （DSP コア）開発時にのみ有効となります。μPD77113，77114 **ではチェックしないでください。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更を廃棄します。

6.7.19　Make...

“Make…”**コマンドは，プロジェクト管理の条件を設定するための** Make Settings **ダイアログを表示します。** Make Settings **ダイアログを図** 6－27 **に示します。**

図 6－27　Make Settings **ダイアログ**

Make Settings **ダイアログの構成要素**

Warnings

これをチェックしておくと，ワーニングを発生させたファイルのアセンブルの後にプロジェクトの作成が停止します。

保守／廃止

Errors

これをチェックしておくと，エラーを発生させたファイルのアセンブルの後にプロジェクトの作成が停止します。

Fatal errors

これをチェックしておくと，フェータル・エラーを発生させたファイルのアセンブルの後にプロジェクトの作成が停止します。

All sources processed

これをチェックしておくと，エラーまたはワーニングが発生したかどうかに関係なく，すべてのソースのアセンブルの後にプロジェクトの作成が停止します。ディフォールトでは，エラーが発生したファイルのアセンブルの後に停止することに注意してください。

Stop

これをチェックしておくと，アセンブルの後にプロジェクトの作成が停止します。

Run linker

これをチェックしておくと，アセンブルの後にリンカが起動します。ディフォールトでは，リンカが起動しオブジェクト・モジュール・ファイルが生成されることに注意してください。

Select project items

これをチェックしておくと，Project **メニューの**“Save As…”**コマンドですべてのファイルを含むプロジェクトを別のディレクトリにコピーできます（**6.6　Project **メニュー参照）。**

Model definition file

これをチェックしておくと，Project **メニューの**“Save As…”**コマンドでモデル定義ファイルを新しいディレクトリにコピーできます（**6.6　Project **メニュー参照）。**

Beep on end

これをチェックしておくと，“Make”**の完了がビープ音で知らされます。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更された設定を廃棄します。

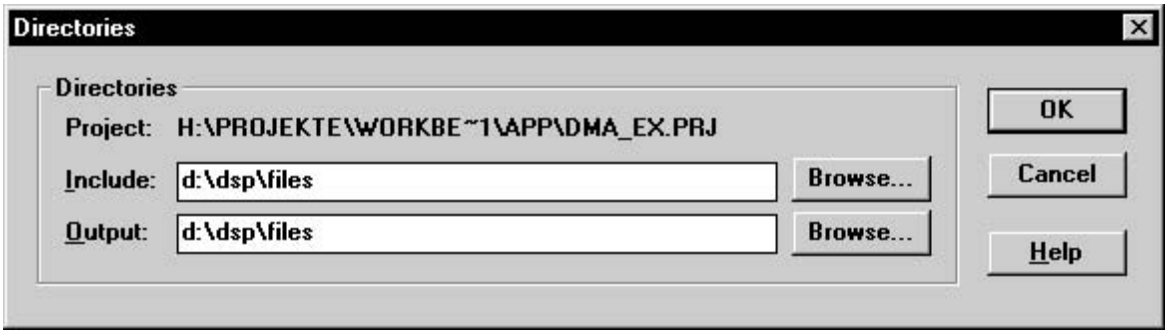
保守／廃止

6.7.20　Directories...

このコマンドは，WB77016 **のプロジェクト管理にインクルード・ファイルとソース・ファイルのある場所，および出力ファイルを格納する場所を通知するための** Directories **ダイアログを表示します。**

Directories **ダイアログを図** 6－28 **に示します。**

図 6－28　Directories **ダイアログ**



Directories **ダイアログの構成要素**

Include

インクルード・ファイルが入っているディレクトリを入力します。標準のインクルード・ファイルは，“#include” **文の中で＜　＞で囲まれています。**

Output

すべてのリンカ出力ファイルとアセンブラ出力ファイルを格納するディレクトリを入力します（リンカ出力ファイルとアセンブラ出力ファイルの詳細については，第 3 **章　ファイル・フォーマット**を参照してください）。WB77016 **のプロジェクト管理は，このディレクトリを使用して，ファイルの検索と書き込み，日付けと時刻のチェックを行います。**

Browse

このボタンは，ハード・ディスクとフロッピィ・ディスクのディレクトリ構成をビジュアルに表示し，リストからパスを選択するための Browse **ダイアログを表示します。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更を廃棄します。

注意　Directories **ダイアログのパス・エントリが空欄の場合，**WB77016 **はカレント・ディレクトリの中のファイルに対して読み書きをします。**

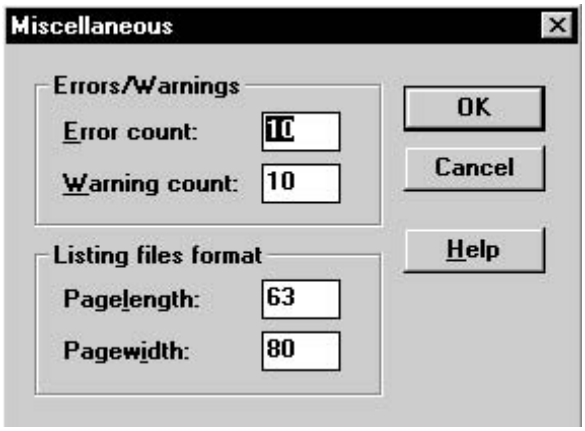
保守／廃止

6.7.21　Miscellaneous...

このコマンドは，アセンブル終了，アセンブラ出力ファイルとリンカ出力ファイルのオプションを設定するための Miscellaneous **ダイアログを表示します。**

Miscellaneous **ダイアログを図** 6－29 **に示します。**

図 6ー29　Miscellaneous **ダイアログ**



Miscellaneous **ダイアログの構成要素**

Error count

このオプションを使用すると，アセンブルまたはリンクが終了するまでに引き渡しができるエラーの数を修正することができます。

Warning count

このオプションを使用すると，アセンブルまたはリンクが終了するまでに引き渡しができるワーニングの数を修正することができます。

Listing files format

アセンブラ・プリント・ファイルのプリンタ出力に対するページ長およびページ幅，リンカ・マップ・ファイルのプリンタ出力に対するページ長およびページ幅を指定します。Miscellaneous **ダイアログの**“Pagelength”**フィールドに表示されるディフォールト値は，**63 **に設定されています。**PRN **ファイルと** MAP **ファイルに対するページの生成を無効にしたい場合，ページ長を** null **にします。開始ページのヘッダを除けば，アセンブラとリンカが生成する** PRN **ファイルと** MAP **ファイルには，改ページ文字やページ番号付のページ・ヘッダはありません。**

OK

ダイアログをクローズし，設定を適用します。

Cancel

ダイアログをクローズし，変更された設定を廃棄します。

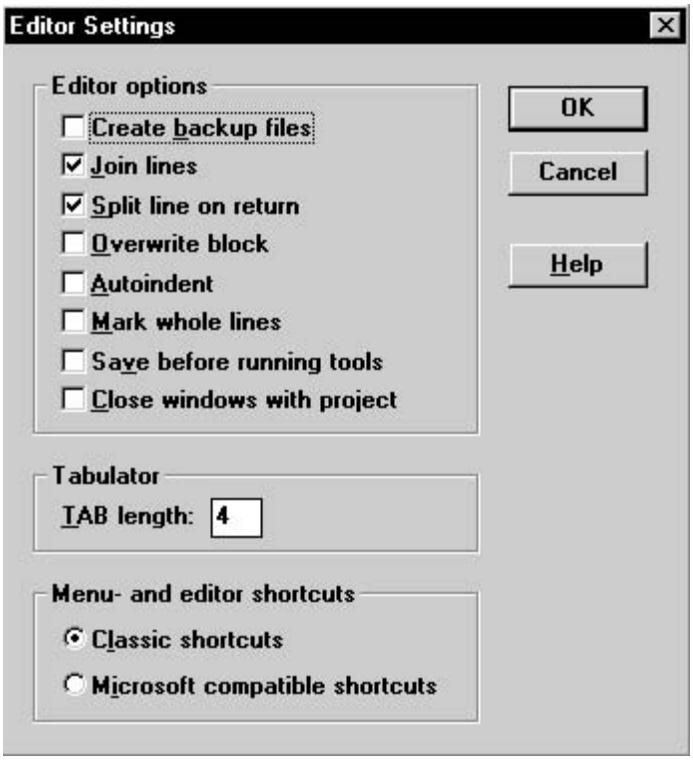
保守／廃止

6.7.22　Editor...

“Editor…”コマンドは，ユーザの必要条件に応じてエディタの処理を設定できる Editor Settings ダイアログを表示します。

Editor Settings ダイアログを図 6－30 に示します。

図 6－30　Editor Settings ダイアログ



Editor Settings ダイアログの構成要素

Create backup files

このコマンドをチェックしていると，アセンブラ・ソース・ファイルを保存するときに，WB77016 はエディタ・ウインドウの中のアセンブラ・ソース・ファイルのバックアップを自動的に作成します。バックアップ・ファイルの拡張子は“.BAK”です。

Join lines

このコマンドをチェックしていると，カーソルが行頭にある場合，BS（バックスペース）キーでその行を前の行と連結することができます。カーソルが行の末尾にある場合には，DEL（削除） キーで次の行と現在行を連結することができます。

Split line on return

このコマンドをチェックしていると，Return キーでカーソル位置にある行を分割することができます。このコマンドをチェックしていない場合には，行は分割されずに，現在カーソルのある行の後に新しい行が挿入され，カーソルはその新しい行の行頭の位置に来ます。

Overwrite block

このコマンドをチェックしていると，マークされているブロックが，挿入されたテキストで上書きされます（テキストを挿入するとは，文字をタイプして入力したりクリップボードからテキストをペーストしたりすることを意味します）。

Autoindent

このコマンドをチェックしていると，Return **キーを押したときに，最初の空白でない文字のところにあるカーソルが前の空白でない行に配置されます。この機能は，読み取り可能なプログラミング・コードを入力するときに便利です。**

Mark whole lines

このコマンドをチェックしていると，マークされている行のカラム幅か，またはマークされている行の一番右の文字まで強調表示されます。

Save before running tools

このオプションを選択すると，“Assemble” **コマンド，**“Make” **コマンド，**“Build”**コマンドを実行する前に，オープンしているか，変更されているソース・ファイルが保存されます。**

Close windows with project

プロジェクトをクローズすると，すべてのドキュメント・ウインドウが自動的にクローズされます。

TAB length

TAB length **入力ボックスを使用して，タブごとに何文字移動するかを指定します。ディフォールト値は** 4 **です。**TAB **は空白として保存されることに注意してください。**

Classic shortcuts

このコマンドを選択すると，WB77016 **のショートカットを前と同様に使用することができます。**

Microsoft compatible shortcuts

このコマンドを使用すると，Microsoft **互換ショートカットを使用することができます。**

OK

ダイアログをクローズし，設定を受け入れます。

Cancel

ダイアログをクローズし，変更された設定を廃棄します。

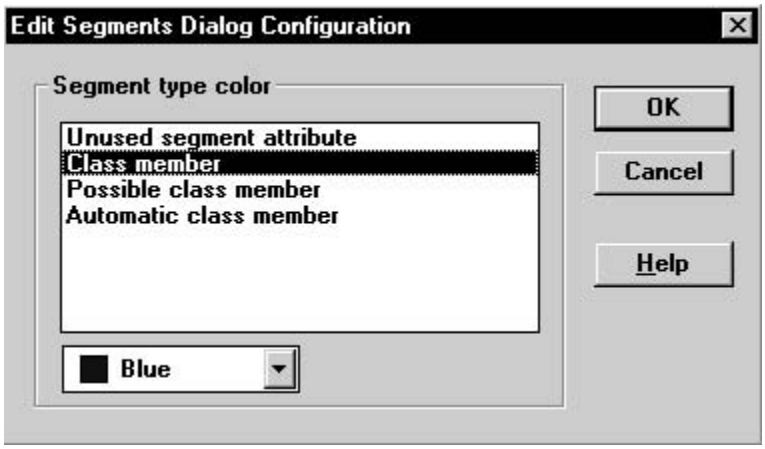
保守／廃止

6.7.23　Segment Colors...

“Segment Colors…”**コマンドは，**Editor Segments **ダイアログに表示されるセグメントの色を変更できる** Edit Segments Dialog Configuration **ダイアログを表示します。**

Edit Segments Dialog Configuration **ダイアログを図** 6－31 **に示します。**

図 6－31　Edit Segments Dialog Configuration **ダイアログ**



わかりやすく整理されたセグメント・リストを表示するために，Editor Segments **ダイアログは，異なるセグメント属性は違った色で表示します。それぞれ次の機能を表します。**

Edit Segments Dialog Configuration **ダイアログ**

Unused segment attribute
現在使用されていないセグメントの属性です。このような属性は，属性をセグメントに割り当てた後に，セグメントの名前を変更するか削除した場合に発生することがあります。

Class member
選択したクラスに属しているセグメント。

Possible class member
選択したクラスに追加できるセグメント。

Automatic class member
現在プロジェクトの一部になっていないセグメントの属性です。このような属性は，属性をセグメントに割り当てた後に，セグメントの名前を変更するか削除した場合に発生することがあります。

OK
ダイアログをクローズし，設定を受け入れます。

Cancel
ダイアログをクローズし，変更を無視します。

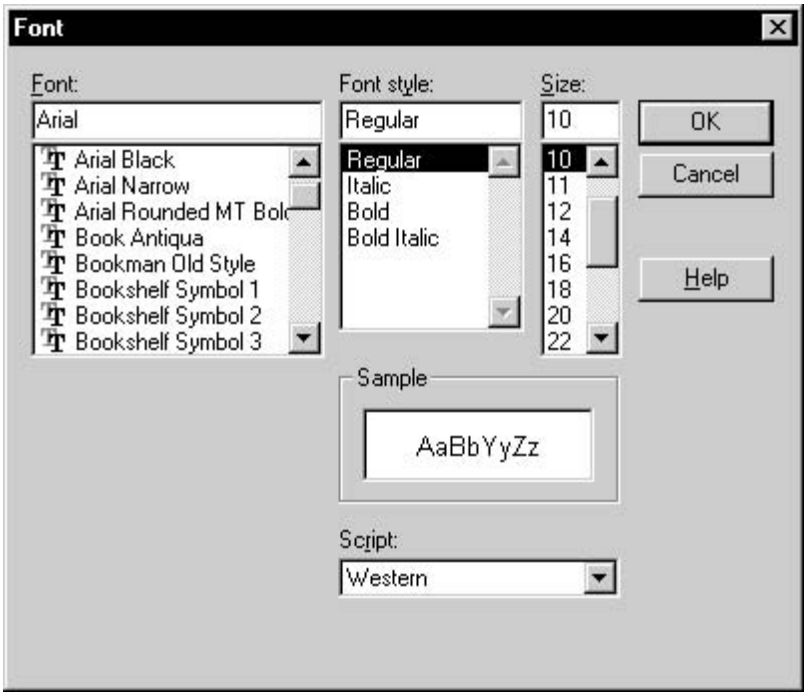
保守／廃止

6.7.24　Font...

“Font…”コマンドは，エディタ・ウインドウのテキストの字体とフォント属性を変更または表示するための Font ダイアログを表示します。

Font ダイアログを図 6－32 に示します。

図 6－32　Font ダイアログ



Font ダイアログの構成要素

Font

ユーザの Windows コンフィギュレーションで使用できるフォントからフォント・タイプを選択します。

Font style

標準，ボールド，イタリック，ボールド・イタリックの中から選択します。

Size

必要なフォントのサイズを選択します。サイズ・ボックスに表示される選択は，Windows が提供するものであり，使用可能なすべての選択を表示しているとは限らないことに注意してください。この場合，必要なポイント・サイズを直接入力し，リスト・ボックスは迂回してください。

Sample

サンプル・ボックスの中の文字列は，選択したすべての属性を持つ選択字体を表示します。ユーザ・システム上で使用可能な任意の ANSI フォント・タイプ名を選択することもできます。実際に使用されるフォントは，Windows に依存し，ユーザの選択と違っている場合があります。多くのプリンタ・フォントと記号フォントは ANSI フォントではないので，選択することはできません。

Script
書体の種類を選択します。

OK
ダイアログをクローズし，フォント設定を適用します。

Cancel
ダイアログをクローズし，操作をキャンセルします。前のフォント設定がそのまま有効です。

6.8　Window メニュー

Window **メニューには，**WB77016 **のウインドウを表示したり，手前に表示したりする標準の** MDI **関数とコマンドがあります。すでにオープンしているウインドウにウインドウ表示コマンドを適用すると，オープンしているウインドウが手前に表示され，コマンドにはチェック・マーク（✓）がつきます。**

6.8.1　Tile

“Tile”**コマンドは，**WB77016 **のメイン・ウインドウに収まるように，オープンしているウインドウを縮小します。**

6.8.2　Cascade

“Cascade”**コマンドは，それぞれのタイトル・バーが見えるようにウインドウを重ねて表示します。**

6.8.3　Arrange Icons

“Arrange Icons”**コマンドは，**WB77016 **のすべてのアイコンを** WB77016 **メイン・ウインドウの最下行に配置します。**

6.8.4　Close All

“Close All”**コマンドは，オープンしているすべてのウインドウをクローズします。エディタ・ウインドウで変更を保存し，それぞれプロジェクトのクローズを確認するように促されます。**

6.8.5　Message

“Message”**コマンドは，**Message **ウインドウをオープンするか，または手前に表示します。**

6.8.6　Project

“Project”**コマンドは，**Project **ウインドウをオープンするか，または手前に表示します。このコマンドは，プロジェクトが選択されていないと淡色表示になります。**

6.8.7　Open Windows List

WB77016 のウインドウは，重ね合せたりお互いを隠すことができる一方で，マウスでそれらをクリックしてアクティブにすることは難しくなります。そこで，すべての Window メニュー項目の後にあるリストには，オープンしているウインドウの番号付リストが入っています。ユーザがリストからウインドウを選択すると，あらたに選択されたウインドウがほかのウインドウの手前に表示されてアクティブ・ウインドウになります。アクティブ・ウインドウは，チェック・マーク（✓）で示されます。

リスト中のウインドウの順序は，オープンされた順です。ウインドウをクローズすると，リストの番号はつけ直されます。このリストには最初から 9 番目までのオープンしているウインドウが表示されます。9 個を越えるウインドウがオープンしている場合，リストには“More Windows…”コマンドが入っています。このコマンドは，リストにすでにあるものを含めてオープンしているすべてのウインドウを表示するダイアログを表示します。

6.9　Help メニュー

Help メニューのコマンドを使用すると，WB77016 のオンライン・ヘルプ，μPD77016 アセンブラ言語，μPD77016 に関するヘルプ情報にアクセスすることができます。

6.9.1　Index

“Index”コマンドは，WB77016 についてのすべての主要なヘルプ・トピックのリストを表示します。ヘルプ・トピックは次のとおりです。

- **コマンド**:　WB77016 のメニュー・コマンドを記述するトピック。
- **ファイル**:　WB77016 のファイル・タイプに関する情報。
- **ウインドウ**:　WB77016 のドキュメント・ウインドウ。
- **レファレンス**:　アセンブル・エラー，リンク・エラー，リンカ・トピックなどに関するほかの情報。
- **キーボード**:　キーボード・ショートカットとキーボード・インターフェイス。
- **ダイアログ索引**:　WB77016 のダイアログのアルファベット順リスト。

6.9.2　Topic Search

“Topic Search”コマンドは，カーソル位置で言語要素についての言語ヘルプを表示します。“Topic Search”は，ニモニック・アセンブラ・ディレクティブ，ニモニック・アセンブラ命令，ニモニック・アセンブラ文に対して有効です。

6.9.3　Using Help

“Using Help”コマンドは，Windows ヘルプ使用に関する簡単なチュートリアルやその他の情報を表示します。

保守／廃止

6.9.4　Assembler

“Assembler”**コマンドは，**WB77016 **アセンブラに関するオンライン情報を表示します。このヘルプ項目の主要なトピックは次のとおりです。**

・アセンブラ言語構成要素
定数
コントロール
ディレクティブ
式
キーワード
マクロ
数字
オペランド
演算子
文
シンボル
アセンブラ命令
予約語
用語集

6.9.5　On-Line Manual

このコマンドは，μPD77016 **ファミリの機能仕様に関する情報を表示します。このヘルプ項目の主要なトピックは次のとおりです。**

・内部ブロック
・レジスタ・セット
・アドレシング・モード
・データ・フォーマット
・命令セット
・割り込み
・組み込み周辺機器
・メモリ・インターフェイス
・AC/DC **ターゲット仕様**

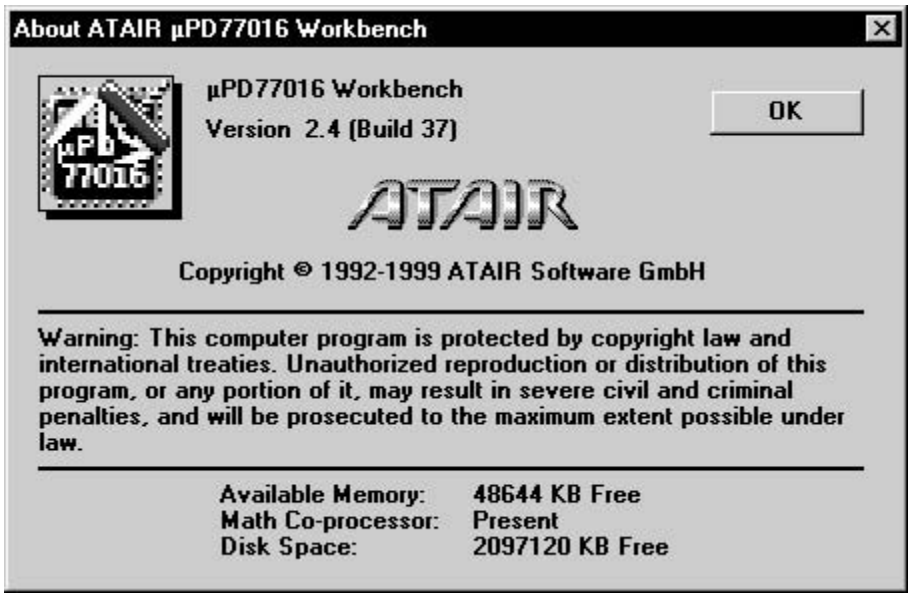
注意　仕様情報は変更されている場合がありますので，ご設計の際には最新のデータシートでご確認ください。

6.9.6　About...

このコマンドは，WB77016 **に関するいくつかの情報を表示する** About **ダイアログを表示します。この中には，アプリケーション名とアプリケーション・アイコン，プログラム・バージョン番号，著作権情報，プログラム作成者の名前などがあります。線で区切られた下の領域には，ユーザ・システムや現在のシステム・リソースに関する情報があります。**

About **ダイアログを図** 6－33 **に表示します。**

図 6－33　About **ダイアログ**



［メ　　モ］

★

付録 A　リンカ

A.1 概　要

リンカは，オブジェクト・モジュール・ファイルとライブラリ・ファイルの内容を 1 つのロード・モジュール・ファイルにまとめます。このために，リンカはオブジェクト間のシンボルの参照を解決し，セグメント・クラスのメンバを統合して，リロケータブル・セグメントとセグメント・クラスを割り当てます。オプションとして，内蔵オブジェクト・コンバータがさまざまな種類の HEX ファイルを生成します。

リンカには，次のような特徴があります。

・内蔵ロケータが，ロード・モジュール・ファイル内のリロケータブル・セクションの開始アドレスを定義します。コードとデータが書き込まれる空きメモリ領域が，専用のモデル定義ファイルのモデルに対して定義されます。実際のシステム・モデルは Options メニューの “Processor Model...”コマンドで選択できます。

表 A－1 に示すように，セクションは再配置の実行中に宣言に従って配置されます。

表 A－1　メモリ・セグメントの配置

ソース・ファイル内の宣言	ロード・モジュール内の配置
imseg	内部命令 ROM または RAM
imseg external	外部命令 ROM または RAM
xromseg	内部データ XROM
xromseg external	外部データ XROM
xramseg	内部データ XRAM
xramseg external	外部データ XRAM
yromseg	内部データ YROM
yromseg external	外部データ YROM
yramseg	内部データ YRAM
yramseg external	外部データ YRAM

・リンカはブート情報を追加生成できます。
・データ RAM を初期化するために，ROM 化可能なコードを生成できます。
・オプションとして，リンカが次の 3 つの部分（セクション，ローカル・シンボル，パブリック・シンボル）に分かれるマップ・ファイルを生成します。

マップ・ファイルの内容は Linker Settings ダイアログの設定によって変わります。リンカが出力するファイルの詳細については，第 3 章　ファイル・フォーマットを参照してください。

保守／廃止

A.2　セグメント

セグメントは対象メモリの連続した領域を表す分割できない単位です。セグメントはユーザがソース・プログラムを使って定義するか，リンクを行う前に直接定義されます。各セグメントにはセグメント名，セグメント長，対象メモリの位置，クラスの構成を指定する属性が付けられます。

セグメントの属性には次のようなものがあります。

・クラス構成
・セグメントが属するモジュール
・セグメント名
・セグメントの開始アドレス
・セグメント長
・メモリの配置属性

imem，xmem，ymem，ram，rom

内部または外部

絶対，再配置，配置済み

A.2.1　セグメントの配置

配置は別々のアセンブリ結果を 1 つの絶対オブジェクト・ファイルにまとめ，それに従ってセグメント間のアドレス参照を解決します。それぞれの指定セグメントやセグメントのクラスは決められたアドレス範囲で割り当てられます。絶対セグメント（WB77016 のアセンブラが生成）とユーザが配置したセグメントがセグメントの重複を避けるために最初に置かれます。配置プロセスは Options メニューの“Edit Segments...”コマンドで制御できます。

・Absolute segments

絶対セグメント（WB77016 アセンブラが生成）とユーザが配置したセグメントの重複を避けるために最初に置かれます。

・Relocateable segments

リロケータブル・セグメントは，ほかのセグメントとは異なり対象メモリのどこにでも置くことができます。リロケータブル・セグメントのアドレス参照はシンボルで表されるため，配置プロセスで絶対アドレス参照に置き換えることができます。

・Overlay segments

セグメント・オーバレイ・クラスは重複して割り当てられるセグメントをユーザが指定できます。オーバレイ・セグメントはメモリ内の同じ開始アドレスに配置されます。同じオーバレイ・クラスのメンバになることになっているセグメント・グループは重複から除外できます。

保守／廃止

A.3　セグメント・クラス

クラスは共通の論理属性を持つセグメントの集まりです。クラス構成はセグメントの目的や目的とする使い方を表します。セグメントは同時に複数のクラスのメンバになることができます。クラス名はリンカが処理する（初期化など）セグメントのリストの参照に都合のよいハンドル名を指定します。

WB77016 のリンカは次に示すセグメント・クラスを扱います。

- Boot class: μPD77016 ファミリのブート機能に対応します。
- INI class: このデータ RAM 初期化クラスはユーザ定義のセグメント名を付けます。
- INID class: この RAM 初期化クラスはアセンブラ・ソース・ファイルに含まれるディフォールトのセグメント名を使います。
- Overlay class: 所定のオーバレイ・クラスのメンバであるセグメントのオーバレイ割り当てが可能です。
- Preload module class: プリロード・モジュール・クラスのセグメントはμPD77116 のキャッシュ可能メモリに割り当てられます。
- DMA class: DMA セグメント・クラスは特に指定しない場合は別々の DMA メモリ・バンクに割り当てられる内部データ・セグメントのグループです。

A.3.1　ブート・クラス

リンカはブート・クラスを使って Self Boot モードと Host Boot モードをサポートします。ブート・クラスはインストラクション・メモリ・セグメントの集まりで，ブート・シーケンスの完了後に，実行または再書き込みされるプログラム・コードを持ちます。ユーザが設定するブート・クラスの属性には所定のブート・モードを実行するために必要なすべての設定があります。

(1) ブート・クラスのタイプ

ブート・モードに対するブート・クラスのタイプは，次のブート・クラス属性を設定すると作成されます。

- Boot source:

Self Boot モードか Host Boot モードかを定義します。

Self Boot モードに関しては，自動的に生成されるブート・データ・セグメントが割り当てられるデータ・メモリのタイプを特定する必要があります。特に指定しない場合は，ブート・パラメータを含むデータ・セグメントが割り当てられます。

- Starting format:

Reset モードか Reboot Boot モードかを定義します。Reboot ブート・クラスに関しては，ブート・データ・フラグメント数の値を 1 以上に指定できます。複数のブート・データ・フラグメントを使う必要があるのは，複数のインストラクション・メモリ領域があるプロセッサ（μPD77110 など）の場合と，ブート・クラスが 1 つの利用可能なインストラクション・メモリ領域に完全に収まらない場合のみです。ブート・データ・フラグメント数の値を 1 以上に設定すると，ブート・クラスに書き込まれたブート・データが指定した数のフラグメントに分割され，使用できないインストラクション・メモリ領域に擬似データが生成されるのを避けます。

・Data format:

Self Boot モードのときに転送フォーマットがバイト・データ転送かワード・データ転送かを定義します。

これらのブート・クラス属性のほかに，ユーザがブート・クラスの名前と種類を指定できます。Self Boot ブート・クラスの場合は，ブート・クラスの次にブート・データ・セグメントを指定します。

次に示すように，リンカはいろいろな種類のブート・クラスに対応します。

・Reset self-boot:

リンカはブート・パラメータを含むセグメントをアドレス 0x4000:Y（μPD77116 の場合は 0x8000:Y）に追加し，所定の X データ・メモリまたは Y データ・メモリの所定のセグメント（該当するブート・クラスのメンバ）をコピーします。

・Self reboot:

リンカはブート・パラメータを含むパブリック・シンボルをエクスポートし，所定のデータ・メモリの所定のセグメント（該当するブート・クラスのメンバ）をコピーします。

ブート・データ・フラグメント数の値を 1 以上に設定すると，それぞれのブート・データ・フラグメントに別々のデータ・セグメントが作成されます。ブート・データ・セグメント名は次の法則に従って付けられます。

BootClassName_FragmentNumber

BootClassName	それぞれのリブート・ブート・クラスの名前
FragmentNumber	それぞれのリブート・ブート・クラス・フラグメントを表す整数

・Host reboot:

リンカはブート・パラメータを含むパブリック・シンボルをエクスポートし，所定の HEX ファイルの所定のセグメント（該当するブート・クラスのメンバ）をコピーします。ブート・データ・フラグメント数の値を 1 以上に設定すると，それぞれのブート・データ・フラグメントに別々の HEX ファイルが作成されます。HEX ファイル名は次の法則に従って付けられます。

HexFileName_FragmentNumber.HEX

HexFileName	ユーザが指定した HEX ファイル名
FragmentNumber	それぞれのリブート・ブート・クラス・フラグメントを表す整数

保守／廃止

(2) ブート・クラスで生成されるパブリック・シンボル

表A－2に示すパブリック・シンボルはリンカがセルフ・リブート・ブート・クラスに対して生成します。

表A－3に示すパブリック・シンボルはホスト・リブート・ブート・クラスに対して生成されます。

表A－2　パブリック・シンボル

パブリック・シンボル	Xメモリ・リブート	Yメモリ・リブート
*classname*_**r7l**	リブートされる命令ステップ数	リブートされる命令ステップ数
*classname*_**dp3**	opコードを格納するXメモリの開始アドレス	ロードするインストラクション・メモリの開始アドレス
*classname*_**dp7**	ロードするインストラクション・メモリの開始アドレス	opコードを格納するYメモリの開始アドレス

表A－3　ホスト・リブート・ブート・クラスに対するパブリック・シンボル

パブリック・シンボル	ホスト・リブート
*classname*_**r7l**	リブートされる命令ステップ数
*classname*_**dp3**	ロードするインストラクション・メモリの開始アドレス

1つ以上のブート・データ・フラグメントがあるリブート・ブート・クラスに対して，次の法則に従ってパブリック・シンボルが作成されます。

BootClassName_FragmentNumber_Register

BootClassName	リブート・ブート・クラスの名前
FragmentNumber	それぞれのリブート・ブート・クラスのブート・データ・フラグメントを表す整数
Register	対応するブート・パラメータを持つレジスタの名前（詳細は表A－2　パブリック・シンボルを参照してください）

保守／廃止

(3) ブート・モード

表 A−4 にブート・モードの概要を示します。ブートのプロセスをハードウエアで実行する方法の詳細については，μPD7701x ファミリ　ユーザーズ・マニュアル　アーキテクチャ編，または μPD77111 ファミリ　ユーザーズ・マニュアル　アーキテクチャ編を参照してください。

表 A−4　ブート・モード

	Self Boot モード	Host Boot モード
Reset Boot モード Reset Boot モードとは DSP ハードウエアをリセットすることです。ハードウエアがリセットされると，インストラクション・ポインタは 0x0（リセット・ブート・エントリ・ポイント）に設定されます。リセット・ブート・ルーチンが実行されると，インストラクション・ポイントは 0x200（リセット・ベクトル）に設定されます。	プログラム・コード・データはデータからインストラクション・メモリに転送されます。データとインストラクション・メモリ間のデータ転送はバイト形式またはワード形式で実行できます。ブート・パラメータはアドレス 0x4000 の Y データ・メモリに設定されます。	プログラム・コード・データはホスト・インタフェースを通してインストラクション・メモリへ転送されます。プログラム・コード・データの前にブート・パラメータがホスト・インタフェースで読み取られます。
Reboot Boot モード Reboot モードとはアプリケーション・プログラム内からブート・サービス・サブルーチンが呼び出されることです。ブート・サービス・サブルーチンのエントリ・ポイントは DSP のブート ROM 領域に置かれます。	プログラム・コード・データはデータからインストラクション・メモリへ転送されます。データとインストラクション・メモリ間のデータ転送はバイト形式またはワード形式で実行できます。ブート・パラメータはレジスタ:R7L，DP3，DP7 に設定されます（アプリケーション・プログラム内から）。	プログラム・コード・データはホスト・インタフェースを通してインストラクション・メモリへ転送されます。ブート・パラメータはレジスタ:R7L，DP3 に設定されます（アプリケーション・プログラム内から）。

(4) ブート・クラスの割り当て

リンカは次の機能に従ってブート・クラスを割り当てます。

・リセット・ブート・クラスはリセット・ベクトル（0x200）のアドレスを持つインストラクション・メモリ領域だけに割り当てられます。

・リブート・ブート・クラスはリセット・ベクトルのアドレスがあるかどうかにかかわらず，どのインストラクション・メモリ領域にでも割り当てることができます。

・リンカはそれぞれのブート・クラス（リセットまたはリブート）のために確保していたインストラクション・メモリにブート・クラス全体を空白ができないように割り当てます。

(5) インストラクション・メモリの割り当て順序

空白がない，またはできるかぎり空白が少ないブート・クラスの割り当てを行うために，インストラクション・メモリ・セグメントの割り当て順序は次のように実行されます。

1．どのクラス構成にも属さない絶対セグメント
2．割り当てパラメータ'Allocate on absolute address ####'を持つプリロード・モジュール・クラス
3．割り当てパラメータ'Allocate inside of bank #'を持つプリロード・モジュール・クラス
4．割り当てパラメータ'Allocate on block size'を持つプリロード・モジュール・クラス
5．リセット・ブート・ブート・クラス
6．リブート・ブート・クラス（クラスのサイズ順，大きいクラスの次に小さいクラス）
7．属性が整列またはリロケータブルな残りのすべてのセグメント

(6) 正しいインストラクション・メモリ領域の選択

ブート・クラスを割り当てる前に，すべてのブート・クラスのメンバを確定します。ブート・クラスのメンバは次のようなものです。

・Edit Segments ダイアログで明確に追加されたブート・クラス・メンバ。
・所定のライブラリから追加されたブート・クラス・メンバ。これは Boot Class ダイアログで選択されます。

リセット・ブート・ブート・クラスに関しては，リセット・ベクトル（0x200）のアドレスを持つインストラクション・メモリ領域にクラス全体が割り当てられます。

リブート・ブート・クラスに関しては，（確保されたインストラクション・メモリに）空白ができないように，ブート・クラス全体が収まるサイズのインストラクション・メモリ領域にクラス全体が割り当てられます。リブート・ブート・クラスが 1 つのインストラクション・メモリに連続して収まらない場合は，ブート・データをブート・データ・フラグメントに分割する必要があります。こうすることで複数のインストラクション・メモリ領域を持つプロセッサ（μPD77110 など）に対応できます。

(7) ブート・クラス内の空白の回避

ブート・クラスに少なくとも 1 つの絶対セグメントがある場合，同じブート・クラスで利用できる残りのリロケータブル・セグメントは，すでに割り当てられている絶対セグメントの直後か少なくとも同じインストラクション・メモリ領域に割り当てられます。こうすることで，できるかぎりブート・クラス内に空白ができないようにします。

保守／廃止

(8) 異なるブート・クラスの自動オーバレイ

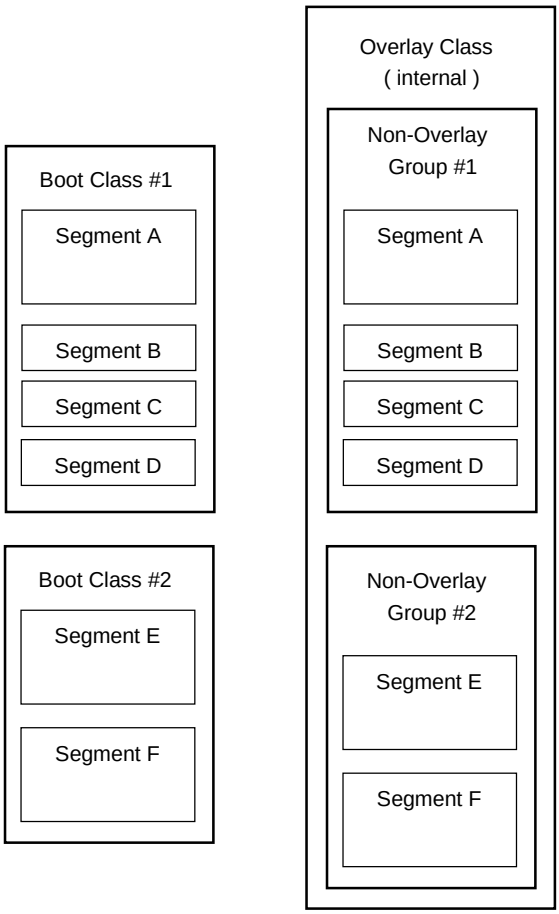
ブート・クラスは，すべてのブート・クラスのメンバがあるオーバレイ・クラス（6.7.9　Overlay Class ダイアログを参照）のメンバであるかのように，さらに，ある特定のブート・クラスのメンバが非オーバレイ・グループ（1 つのブート・クラスに対して 1 つの非オーバレイ・グループ）のメンバであるかのように処理されます。このオーバレイ・クラスと非オーバレイ・グループは，どのような方法でも表示されませんが，リンカが内部で任意の利用可能な異なるブート・クラスをオーバレイで割り当てるのに使われます。ある特定のブート・クラスのセグメントは当然非オーバレイ・グループのメンバとして扱われるため，オーバレイで割り当てることはできません。

リンカはこのような割り当てを適用して，異なるブート・クラスが同じインストラクション・メモリを使えるようにします。ブート・クラスのメンバのセグメント属性でリンカがセグメントを同じインストラクション・メモリに割り当てるようにすると，異なるブート・クラスだけを割り当てることができます。

1 つのブート・クラスに対して 1 つのオーバレイ・クラスまたは非オーバレイ・グループを使用すると，オーバレイしたブート・クラスの割り当てになります。

図 A−1 は 2 つのブート・クラスの関係と，中で使われているオーバレイ・クラスと非オーバレイ・グループを示しています。

図 A−1　オーバレイしたブート・クラスの割り当て



保守／廃止

(9) ブート・データのヘッダ・フォーマット

リンカはセルフ・ブート機能とホスト・ブート機能をサポートします。ブート手順を実行する方法の詳細については，μPD7701x ファミリ　ユーザーズ・マニュアル　アーキテクチャ編，または μPD77111 ファミリ　ユーザーズ・マニュアル　アーキテクチャ編を参照してください。

(10) セルフ・リセット・ブートのブート・ヘッダ

次の設定のブート・クラスのブート・ヘッダは表 A−5 のように生成されます。

ブート・ソース　　：Self-boot
開始フォーマット　　：Reset boot
データ・フォーマット　：Word

セルフ・リセット・ブートのブート・ヘッダ・フォーマット（ワード・フォーマット）を表 A−5 に示します。

表 A−5　セルフ・リセット・ブートのブート・ヘッダ・フォーマット（ワード・フォーマット）

アドレス	メモリの値
0x4000:Y	16/8 ビット，X/Y
0x4001:Y	DWTR に設定された値
0x4002:Y	IWTR に設定された値
0x4003:Y	読み取るプログラムを格納するデータ・メモリの開始アドレス
0x4004:Y	プログラムのステップ数

次の設定のブート・クラスのブート・ヘッダは次のように生成されます。

ブート・ソース　　：Self-boot
開始フォーマット　　：Reset boot
データ・フォーマット　：Byte

セルフ・リセット・ブートのブート・ヘッダ・フォーマット（バイト・フォーマット）を表 A−6 に示します。

保守／廃止

表A－6　セルフ・リセット・ブートのブート・ヘッダ・フォーマット（バイト・フォーマット）

アドレス	メモリの値
0x4000:Y	16/8 ビット，X/Y
0x4001:Y	－
0x4002:Y	DWTR に設定された値(下位バイト)
0x4003:Y	DWTR に設定された値(上位バイト)
0x4004:Y	IWTR に設定された値(下位バイト)
0x4005:Y	IWTR に設定された値(上位バイト)
0x4006:Y	読み込まれるプログラムを格納するデータ・メモリの開始アドレス(下位バイト)
0x4007:Y	読み込まれるプログラムを格納するデータ・メモリの開始アドレス(上位バイト)
0x4008:Y	プログラムのステップ数(下位バイト)
0x4009:Y	プログラムのステップ数(上位バイト)

表A－6 に示したブート・ヘッダの値は，すべてブート・クラス・パラメータ（開始アドレス，16/8 ビットに対する設定，X/Y など）と使用したモデル・ファイル（DWTR，IWTR）で確定します。

（11）ホスト・リセット・ブートのホスト・ブート・パラメータ

次の設定のブート・クラスのホスト・ブート・パラメータは表A－7 のように生成されます。

ブート・ソース:　Host-boot
開始フォーマット:　Reset boot
データ・フォーマット:　Byte

表A－7　ホスト・ブート・パラメータのフォーマット

ワード	説　明
1 ワード目	HST に設定される値
2 ワード目	IWTR に設定される値
3 ワード目	ロードするインストラクション・メモリの開始アドレス
4 ワード目	プログラムのステップ数

次の設定のブート・クラスのホスト・ブート・パラメータは表 A－8 のように生成されます（対象のプロセッサがμPD77116 の場合）。

ブート・ソース:　Host-boot
開始フォーマット:　Reset boot
データフォーマット:　Byte

表A−8　ホスト・ブート・パラメータのフォーマットII

バイト	説　明
1バイト目	HST（下位バイト）に設定される値
2バイト目	HST（上位バイト）に設定される値
3バイト目	IWTR（下位バイト）に設定される値
4バイト目	IWTR（上位バイト）に設定される値
5バイト目	ロードするインストラクション・メモリの開始アドレス（下位バイト）
6バイト目	ロードするインストラクション・メモリの開始アドレス（上位バイト）
7バイト目	プログラムのステップ数（下位バイト）
8バイト目	プログラムのステップ数（上位バイト）

表A−8に示したホスト・ブート・パラメータの値はすべてブート・クラス・パラメータ（開始アドレス，16/8ビットに対する設定など）と使用したモデル・ファイル（IWTR）で確定します。

A.3.2　データDMAセグメント・クラス

データDMAクラスによりセグメント単位でデータ・セグメントの内部データ・メモリと外部DMAデータ・メモリからのDMA転送が可能になります。内部Xデータ・メモリまたはyデータ・メモリと外部X-DMAデータ・メモリまたはY-DMAデータ・メモリ間のセグメント単位でのデータDMA転送を行うために，データDMAクラスには対になるセグメント（ペア・セグメント）の機能があります。ペア・セグメントは必ずDMAデータ・メモリに割り当てられ，ここには内部データ・メモリに割り当てられたセグメント内容のコピーがあります。このようにしてプログラムはセグメント単位で内部データ・メモリのセグメントと外部DMAデータ・メモリのセグメント間のデータDMA転送を実行できます。

データDMAクラスにはDMA転送機能はありません。何らかのデータDMA転送を行うにはユーザが必要なプログラム・コードを指定する必要があります。

(1) データDMAクラスのタイプ

割り当てのオプションが複数あるDMAクラスのタイプは1つしかありません。DMAクラスは特性上メモリ・バンクを処理します。データDMA転送のために内部データ・メモリがメモリ・バンクに分割されるため，それぞれのDMAクラスは別々のメモリ・バンクに割り当てられます。違うDMAクラスの割り当てオプションを使うと，リンカはクラス全体のメモリ・バンク単位での割り当てを次のオプションで行うことができます。

Allocate in separate bank:

これはディフォールトのクラス属性です。クラスはDMAアクセス可能な内部メモリの別々のバンクに割り当てられます。どのDMAクラスも同じ内部メモリ・バンク内に割り当てられません。ただし非DMAクラスまたは共通セグメントは，この割り当てオプションを使って選択したバンクに割り当てることができます。

Inside of bank #:

DMAクラスは指定されたバンクに置かれます。同じ割り当てパラメータとバンク番号が指定されたその他のDMAクラスも同じ内部メモリ・バンクに割り当てることができます。さらに非DMAクラスまたは共通セグメントは，この割り当てオプションを使って選択したバンクに割り当てることができます。

Absolute address:
DMA クラスは指定された絶対アドレスに置かれます。

(2) データ DMA クラスで生成されるパブリック・シンボル

データ DMA クラスに追加される各内部データ・メモリ・セグメントに対して，対応するペア・セグメントが外部 DMA データ・メモリに設定されます。何らかの DMA 転送を行うには，ユーザはこの 2 つのデータ・セグメント間で DMA 転送を行う必要があります。DMA コントローラにメモリ開始アドレスとメモリ・サイズの正しい値を書き込むために，リンカは表 A－9 に示すペア・セグメントに対してパブリック・シンボルを設定します。

表 A－9　ペア・セグメントに対するパブリック・シンボル

内部データ・メモリ・セグメント		外部 DMA データ・メモリ・セグメント		
モジュール	セグメント	モジュール	セグメント	パブリック・シンボル
MOD1	SEG1	NEW	DMA$MOD1$SEG1	DMA$MOD1$SEG1

ペア・セグメント名は次の法則に従って付けられます。

"**DMA**" + "$" + <Module> + "**$**" +

<Module> 内部データ・セグメントのモジュール名
 内部データ・セグメントのセグメント名

ペア・セグメントは必ず'_NEW_'というモジュール名で作成されます。

(3) データ DMA クラスの割り当て

データ DMA クラスはモジュール単位で割り当てられます。つまり，セグメント（クラスのメンバ）はそのクラス構成に注目して割り当てられます。

データ DMA クラスの割り当て順序は次のとおりです。

1．割り当て属性'Absolute address'を持つすべての DMA クラス
2．割り当て属性'Inside of bank #'を持つすべての DMA クラス
3．割り当て属性'Allocate in separate bank'を持つすべての DMA クラス

保守／廃止

A.3.3　データ RAM 初期化クラス

データ・メモリの初期化に対応するため，リンカにはデータ RAM 初期化クラスがあります。データ RAM 初期化クラスはクラスのタイプ別に明示的または黙示的に指定されたデータ RAM セグメントの集まりです。リンカはデータ ROM セグメントまたは RAM セグメントの初期化に必要なデータを持つ HEX ファイルを生成する必要があります。データ RAM セグメントそのものの初期化は，プログラムによってコードの対応する部分を使って明示的に行う必要があります。インストラクション・メモリを初期化するためのブート支援に相当する機能が対応するプロセッサに実装されていないため，ユーザがデータ RAM を初期化するためのコードを指定します。

(1) クラス・タイプ

データ RAM 初期化クラスには，INI クラスと INID クラスの 2 つがあります。

(a) INI クラス

明示的に追加されたクラスのメンバ（データ RAM セグメント）を持つデータ RAM 初期化クラスです。このクラス・タイプを使うとユーザはデータ RAM セグメントを明示的に指定することができ，これに対してリンカがデータ RAM 初期化データを書き込みます。

INI クラスの'data'と'zeros'に追加される各セグメントに対して次の初期化モードを選択できます。

詳しい説明は表 A－10 を参照してください。

表 A－10　初期化モード

初期化モード	説　明
data	セグメントはリンク時に指定された内容で初期化されます。初期化データを作成するときにできるだけ多くのメモリを節約するために，リンカがセグメント内容に対して次の分析を行います。 セグメント内容に連続する空文字のワードが見つかった場合は，リンカはこれに対する区分'zeros'というレコードを設定します。 セグメント内容に連続する未初期化ワードが見つかった場合は，リンカはこのワードを無視します。 **注意**　かなりの量のメモリを節約できると，リンカは区分'zeros'というデータ・レコードを設定します（ただし，通常の初期化モードでは'data'に設定）。
zero	セグメントはリンク時の内容に関わらず空文字のワードで初期化されます。

(b) INID クラス

黙示的に追加されたクラスのメンバ（データ RAM セグメント）を持つデータ RAM 初期化クラスです。このクラス・タイプは所定の法則に従って名前を付けられたデータ RAM セグメントをすべて自動的にクラスへ追加し，データ RAM 初期化データを書き込みます。INID クラスの数は 1 つに限られています。次に示した文字列で始まる名前のデータ RAM セグメントはすべて INID クラスに追加されます。

in_x_idata[*name*]

ex_x_idata[*name*]

in_y_idata[*name*]

ex_y_idata[*name*]

INID クラスに追加されるセグメント名の例は次のとおりです。INID クラスのセグメントに対する初期化モードは'data'に設定され，変更はできません。

in_x_idata
ex_y_idata_SinusTable

データ RAM 初期化データの書き込みを制御するために，それぞれのクラス・タイプ（INI と INID）に対して次のクラス・パラメータが指定されます（Self-initialization と Host- initialization）。

（c）Self-initialization

データ RAM 初期化データはデータ ROM セグメントとして X メモリまたは Y メモリに書き込まれます。クラス・タイプによって次のようなデータ ROM セグメントの命名規約が適用されます。

・INI Class

データ ROM セグメントの名前はクラス名と同じです。先頭のデータ・ワードはクラス名の前に下線を付けたパブリック・シンボルでアクセスできます。

クラス名	データ ROM セグメント名	データ ROM セグメントの 1 ワード目にアクセスするパブリック・シンボル
DataInit	DataInit	_DataInit

・INID Class:

このタイプにはクラスが 1 つしかないため，クラス名は次の表に示すように付けられます。

クラス名	データ ROM セグメント名	データ ROM セグメントの 1 ワード目にアクセスするパブリック・シンボル
INID_CLASS	INI_ROM	_INI_ROM

（d）Host-initialization

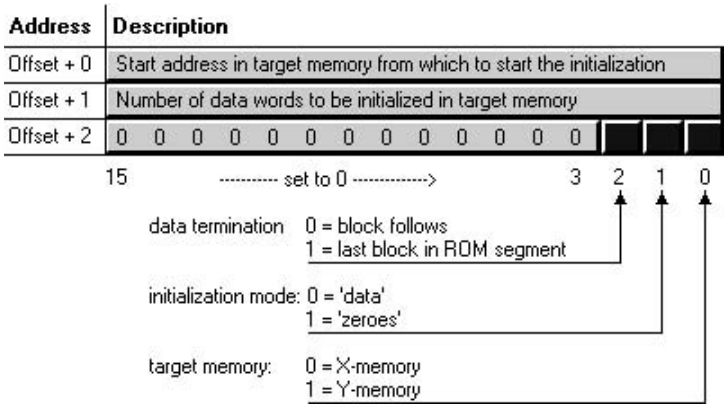
データ RAM 初期化データは HEX ファイルとして書き込まれます。使用するファイル名とアドレシング・フォーマットは，各 HEX ファイルに対してクラス・タイプに関係なく別々に設定できます。
HEX ファイルの内容はプロセッサのホスト・インタフェースを通して読み取ることができます。

保守／廃止

(2) データ・フォーマット

図 A－2 はデータ RAM 初期化データのフォーマットで，INI または INID クラスで設定されます。初期化データはレコードのシーケンスから成り，各レコードには次のようなヘッダが含まれます。

図 A－2　RAM 初期化データ・フォーマット



初期化データを読み取る（ユーザ）コードは特定のレコード番号やレコード・タイプを想定していません。データ値と初期化モードに従って初期化データのサイズを最適化するために，リンカがデータ・レコードのシーケンスを自動的に最適化します。つまり，リンカはたとえ初期化モードが“data”でも，自動的に連続する空文字のワードを検出し，該当するレコードを初期化データに挿入してメモリを節約します。

表 A－11 はレコードのシーケンスの例を示しています。

表 A－11　レコードのシーケンスの例

アドレス	説　明
Offset + 0	開始アドレス
Offset + 1	ワード数
Offset + 2	フラグ（初期化モード：“zeros”）
Offset + 3	開始アドレス
Offset + 4	ワード数
Offset + 5	フラグ（初期化モード：“data”）
Offset + 6	初期化データの 1 ワード目
Offset + 7	初期化データの 2 ワード目
Offset + 8	初期化データの 3 ワード目
Offset + 9	初期化データの 4 ワード目
Offset + 10	初期化データの 5 ワード目
Offset + 11	開始アドレス
Offset + 12	ワード数
Offset + 13	フラグ（初期化モード：“zeros”，最終ブロック）

(3) データ RAM 初期化クラスの割り当て

データ RAM 初期化クラスのメンバは，クラスの構成とは関係なく割り当てられます。すべてのクラスのメンバはディフォールトのセグメント割り当て機能で割り当てられます。

保守／廃止

A.3.4　オーバレイ・クラス

オーバレイ・クラスはセグメントを簡単にオーバレイで割り当てることができます。オーバレイ割り当てとは，2つ以上のセグメントをセグメントの内容に関係なく同じメモリ領域に割り当てることです。

所定のセグメントを非オーバレイ・グループに追加すると，そのセグメントをオーバレイから外すことができます（**図**6ー20　Segment Overlay Attributes ***ダイアログ*****を参照してください**）。

オーバレイ・クラスのセグメントは同じメモリ領域に割り当てられ，そのメモリ領域の内容は未定義になります。オーバレイで割り当てられるセグメントに特定の初期化データがある場合は，どのセグメントのデータがメモリ領域で利用されるのか保証されません。したがって，初期化されていないデータだけを持つセグメント（複数のセグメントから成るスタックをオーバレイする場合と同じように使うデータ・セグメントなど）に対してオーバレイ・クラスが有効に利用されます。

セグメントのオーバレイ割り当て機能は初期化されていないデータを持つセグメントしか考慮しないため，インストラクション・メモリのセグメントはオーバレイ・クラスに追加できません。

(1) 非オーバレイ・グループ

オーバレイ・クラスのメンバであるセグメントのオーバレイ割り当てを無効にするために，表示されたセグメントを非オーバレイ・グループに指定することができます。オーバレイ・クラスのメンバである各セグメントは非オーバレイ・グループのメンバとしても指定できます。オーバレイ・クラスごとのオーバレイ・クラス数と非オーバレイ・グループ数は制限されていません。

次の表はオーバレイ・クラスと非オーバレイ・グループの機能を利用する例を示しています。すべて初期化されていないデータを持つ5つのデータ・セグメントを基準にした例です。これらのセグメントはすべて同じサイズです。

このセグメント・サイズは実際のセグメント・サイズとは異なります。

データ・セグメント	サイズ
SegA	0x100
SegB	0x100
SegC	0x100
SegD	0x100
SegE	0x100

(2) オーバレイ・クラスを使わないセグメント割り当て（ディフォールトのセグメント割り当て）

セグメント'SegA'**と**'SegB'**のオーバレイ割り当てを行うために，指定したセグメントをオーバレイ・クラスに入れます。**

Overlay Class #1:　SegA, SegB

アドレス	そのアドレスに割り当てられるセグメント
0x0000	SegA, SegB
0x0100	SegC
0x0200	SegD
0x0300	SegE

セグメント'SegA'**と**'SegB'**は**'Overlay Class#1'**のメンバであるため，同じメモリにオーバレイで割り当てられます。**

残りのセグメントはオーバレイで割り当てられません（ディフォールトのセグメント割り当て）。

(3) 2 つのオーバレイ・クラスを使うセグメント割り当て

2 つのセグメント・グループのオーバレイ割り当て（case#1**）と，そのグループ間のオーバレイに影響しない割り当てを行うめに，該当するセグメントを 2 つの別々のオーバレイ・クラスに入れます。**

Overlay Class #1:　SegA, SegB

Overlay Class #2:　SegC, SegD, SegE

アドレス	そのアドレスに割り当てられるセグメント
0x0000	SegA, SegB
0x0100	SegC, SegD, SegE

セグメント'SegA'**と**'SegB'**は**'Overlay Class#1'**のメンバであるため，同じメモリにオーバレイで割り当てられます。**

セグメント'SegC'**，**'SegD'**，**'SegE'**は**'Overlay Class#2'**のメンバであるため，同じメモリにオーバレイで割り当てられます。**

保守／廃止

(4) 1 つのオーバレイ・クラスと 2 つの非オーバレイ・グループを使うセグメント割り当て

2 つのセグメント・グループのオーバレイ割り当て（case#2）と，あるグループ内のセグメントをオーバレイしない割り当てのために，1 つのオーバレイ・クラスと 2 つの非オーバレイ・グループを使います。

Overlay Class #1:　SegA, SegB, SegC, SegD, SegE
Non Overlay Group #A:　SegA, SegB
Non Overlay Group #B:　SegC, SegD, SegE

アドレス	そのアドレスに割り当てられるセグメント
0x0000	SegA, SegC
0x0100	SegB, SegD
0x0200	SegE

セグメント'SegA'と'SegB'は Non Overlay Group #A のため，オーバレイで割り当てられません。
セグメント'SegC'，'SegD'，'SegE'は Non Overlay Group #B のため，オーバレイで割り当てられません。
すべてのセグメントは少なくとも同じオーバレイ・クラスのため，2 つの非オーバレイ・グループはオーバレイで割り当てられます。

(5) オーバレイ・クラスの割り当て

オーバレイ割り当ては 2 つ以上のセグメントをセグメントの内容に関係なく同じメモリ領域に割り当てます。

- オーバレイで割り当てられたセグメントに初期化されていないデータがある場合は，メモリ領域のデータも初期化されません。
- オーバレイで割り当てられたセグメントに初期化された（定数など）がある場合は，メモリ領域の内容にオーバレイしたセグメント内容の任意の部分または一部分を入れることができます。

オーバレイ・クラスのセグメントはクラス別に割り当てられません。つまりオーバレイ・クラスのセグメントはディフォールトのセグメントと一まとまりで割り当てられません。このため，オーバレイ・クラスの構成は次のように処理されます。オーバレイ・クラスのあるセグメントを割り当てる場合，オーバレイ・クラスのその他のセグメントはすべて空メモリとして処理されます。このようにして，割り当てるセグメントが十分に収まる次の利用可能な空メモリが使われます。

保守／廃止

A.3.5　プリロード・モジュール・クラス

プリロード・モジュール・クラスのセグメントはμPD77116のキャッシュ可能メモリに割り当てられます。このグループ化されたセグメントにはこのグループのメンバでないセグメントに対する呼び出しやジャンプがほとんどありません。プリロード・モジュール内のセグメントは同時にキャッシュに常駐することになっているため，同じプリロード・モジュール内のアドレスへジャンプや呼び出しを行っても，どのコードもキャッシュ内へプリロードされません。それぞれのプリロード・モジュールを別々のバンクへプリロードできるキャッシュ可能メモリへ割り当てる必要があります。そうすると，その他のモジュールを実行する間に一部のモジュールをプリロードできます。プリロード・モジュール・クラスのメンバであるセグメントだけがキャッシュ可能な外部インストラクション・メモリに割り当てられます。その他のセグメントは直接アクセス可能なメモリか内部インストラクション・メモリだけに割り当てられます。プリロード・モジュール・クラスのセグメントはEdit Segmentsダイアログで作成します。プリロード・モジュール・クラスのプロパティはPreload Module Classダイアログで設定します。

リンカは各プリロード・クラスに対して次のパブリック・シンボルを生成します。

パブリック・シンボル	説　明
classname_beg	プリロード・クラスの開始アドレスに対するパブリック・シンボル
classname_len	プリロード・クラス長に対するパブリック・シンボル

［メ　　モ］

保守／廃止

付録 B　エラーとワーニング

B.1　エラー・リスト凡例

リスト出力の凡例は次のとおりです。

<file>: **エラーが発生しているファイルの名前**

<ln>: **エラーがあると考えられる行番号**

B.1.1　注意事項

アセンブラとリンカのメッセージ A005，A006，A007，A008，L008，L009，L010 **の**<cause>**フィールドには次のメッセージのいずれかが表示されます。**

- the file can't be located
- all or part of the path is invalid
- the permitted number of open files was exceeded
- the file can't be accessed
- the disk is full

リンカのメッセージ L013 **の**<cause>**フィールドには次のメッセージのいずれかが表示されます。**

- memory '<type>' not present
- memory '<type>' '<start>':'<length>' not present
- not enough memory '<type>'
- memory occupied by segment '<name>' (module '<name>')

保守／廃止

B.2　アセンブラ

アセンブラのフェータル・エラー，エラー，ワーニングをメッセージ番号の昇順に説明します。

フェータル・エラーが発生した場合には，アセンブラはただちに停止します。

エラーが発生しても，リロケータブル・ファイルを作成せずにアセンブルを続行します。

B.2.1　エラーとワーニング

"Fatal error A001: User break."

Status ダイアログの"Abort"をクリックすると，アセンブルを中止します。

"Fatal error A002: Too many errors."

発生したエラーの数が制限を越えています。

"Fatal error A003: Too many warnings."

発生したワーニングの数が制限を越えています。

"Fatal error A004: Low memory problem."

メモリの配置に問題が発生しました。現在立ち上げているほかのアプリケーションを終了し，もう一度実行してください。

"Error A005: Can't open file '<name>'- <cause>."

"Error A006: Can't read file '<name>' - <cause>."

"Error A007: Can't write file '<name>' - <cause>."

"Error A008: <file> (<ln>): Can't open include file '<name>' - <cause>."

"Error A009: <file> (<ln>): Include file '<name>' already in use."

インクルード・ファイルが現在のコンパイル単位で繰り返し使用されています。

"Error A010: <file> (<ln>): Syntax error."

"Error A011: <file> (<ln>): Syntax error in 'if' statement."

"Error A012: <file> (<ln>): Label in source module header."

"Error A013: <file> (<ln>): ORG directive in source module header."

"Error A014: <file> (<ln>): DS area reservation directive not in data ram segment."

"Error A015: <file> (<ln>): DW area reservation directive not in data segment."

"Error A016: <file> (<ln>): DWL area reservation directive not in data segment."

"Error A017: <file> (<ln>): Assembly instruction not in instruction memory segment."

"Error A018: <file> (<ln>): Illegal use of 'EX' operator."

条件命令'if'でレジスタ拡張ビット（39 から 31）のテストに不正な演算子が使用されています。使用できるのは，対等を表す演算子（'==', '!=', 'EQ', 'NE'）だけです。

"Error A019: <file> (<ln>): Missing 'END' directive."

"Error A020: <file> (<ln>): Unknown control or no control specified following '$'."

オンラインのアセンブラ制御に構文エラーが発生しました。制御の先頭文字に'$'が見つかりましたが，制御名が正しくありません。

"Error A021: <file> (<ln>): End of file in macro."

"Error A022: <file> (<ln>): Missing semicolon."

"Error A023: <file> (<ln>): Input following end statement."

"Error A024: <file> (<ln>): End of file in comment."

"Error A025: <file> (<ln>): Undefined macro."

"Error A026: <file> (<ln>): Syntax error in macro statement."

"Error A027: <file> (<ln>): Illegal instruction statement at REP."

"Error A028: <file> (<ln>): Illegal instruction statement at LOOP end."

"Error A029: <file> (<ln>): No valid use of data pointer register."

"Error A030: <file> (<ln>): Illegal use of return instruction."

"Error A031: <file> (<ln>): Duplicate register."

"Error A032: <file> (<ln>): No debug directive specified following '.'."

"Error A033: <file> (<ln>): Identifier was already defined by means of 'DEFINE' control."

"Error A034: <file> (<ln>): Illegal identifier after 'DEFINE' control."

"Error A035: <file> (<ln>): '#ELSE' without '#IF', '#IFDEF' or '#IFNDEF'."

"Error A036: <file> (<ln>): '#ENDIF' without '#IF', '#IFDEF' or '#IFNDEF'."

"Error A037: <file> (<ln>): Unterminated string."

文字列定数の終わりにダブルクォーテーションがありません。

"Error A038: <file> (<ln>): Parameter mismatch in macro call."

"Error A039: <file> (<ln>): Can't parallel load store x memory."

"Error A040: <file> (<ln>): Can't parallel load store y memory."

"Error A041: <file> (<ln>): Illegal memory identifier."

直接アドレシング・ロード／ストア命令で指定されたメモリが正しくありません。指定できるのは（'x'，'X'，'y'，'Y'**）だけです。**

"Error A042: <file> (<ln>): Read only register."

"Error A043: <file> (<ln>): 'ORG' value is less than current location address."

"Error A044: <file> (<ln>): Illegal address."

"Error A045: <file> (<ln>): Location address overlap."

"Error A046: <file> (<ln>): Location address overflow."

"Error A047: <file> (<ln>): No location counter ($) in header, assumed 0."

"Error A048: <file> (<ln>): New location less than current location counter value."

"Error A049: <file> (<ln>): Illegal segment name '<name>'."

"Error A050: <file> (<ln>): Public symbol is undefined '<name>'."

"Error A051: <file> (<ln>): Illegal symbol for public."

"Error A052: <file> (<ln>): Already declared public '<name>'."

"Error A053: <file> (<ln>): Multiple defined module name."

"Error A054: <file> (<ln>): 1 or 16 expected."

"Error A055: <file> (<ln>): 1 expected."

"Error A056: <file> (<ln>): Repeat out of range."

"Error A057: <file> (<ln>): Loop count out of range (0x1 to 0x7FFF)."

"Error A058: <file> (<ln>): Loop block must be 2 to 255 instructions wide."

"Error A059: <file> (<ln>): Loop not closed."

"Error A060: <file> (<ln>): No open loop for loop end (})."

"Error A061: <file> (<ln>): Symbol error."

"Error A062: <file> (<ln>): Illegal external symbol."

"Error A063: <file> (<ln>): Illegal segment name."

"Error A064: <file> (<ln>): Symbol already defined."

"Error A065: <file> (<ln>): Illegal symbol definition."

"Error A066: <file> (<ln>): Multiple declaration of label."

"Error A067: <file> (<ln>): 0 expected."

"Error A068: <file> (<ln>): Division by 0."

"Error A069: <file> (<ln>): Illegal expression."

"Error A070: <file> (<ln>): Illegal expression after '#IF' directive."

"Error A071: <file> (<ln>): Evaluation out of range."

"Error A072: <file> (<ln>): Shift value out of range."

"Error A073: <file> (<ln>): Illegal branch address."

"Error A074: <file> (<ln>): Character constant to long."

"Error A075: <file> (<ln>): Radix must be 2, 8, 10 or 16."

"Error A076: <file> <ln>: Segment <name> has different memory attributes - cannot be combined."

"Warning A077: <file> (<ln>): Maximum loop nesting level exceeded."

"Warning A078: <file> (<ln>): Extern symbol already defined."

"Error A079: Overlapped <name> memory area <address> in the current model."

"Error A080: <file> (<ln>): Illegal load/store instruction before 'ret' or 'reti'."

"Error A081: <file> (<ln>): '.efunc'-directive without matching '.func'-directive."

"Error A082: <file> <ln>: Nested macro call."

"Warning A083: <file> <ln>: Pagelength value out of range."

"Warning A084: <file> <ln>: Pagewidth value out of range."

"Error A085: <file> <ln>: Segment <name> has different positioning attributes - cannot be combined."

"Warning A086: <file> <ln>: Unknown storage class <file> in debug directive."

"Error A087: <file> <ln>: Illegal instruction at loop start."

"Error A088: <file> <ln>: Illegal instruction for processor type."

保守／廃止

B.3　リンカ

リンカのフェータル・エラー，エラー，ワーニングをメッセージ番号の昇順に説明します。

B.3.1　エラーとワーニング

"Fatal error L001: User break."

Status ダイアログの"Abort"をクリックすると，リンクを中止します。

"Fatal error L002: Too many errors."

発生したエラーの数が制限を越えています。

"Fatal error L003: Too many warnings."

発生したワーニングの数が制限を越えています。

"Fatal error L004: Low memory problem."

メモリの配置に問題が発生しました。現在立ち上げているほかのアプリケーションを終了し，もう一度実行してください。

"Fatal error L005: Load module file '<file name>'must be the only file in the project."

プロジェクトにロード・モジュール・ファイル（LNK ファイル）以外のファイルが含まれています。プロジェクトにロード・モジュール・ファイル（LNK ファイル）が含まれている場合は，そのファイルだけがプロジェクト・ファイルです。この特徴を利用してロード・モジュール・ファイルを HEX ファイルに変換できます。

"Fatal error L006: Can't load file '<name>' - unknown format."

指定したファイルは有効な REL ファイル，LNK ファイル，LIB ファイルではありません。

"Error L007: Can't open file '<name>' - <cause>."

"Error L008: Can't write file '<name>' - <cause>."

"Error L009: Can't read file '<name>' - <cause>."

"Error L010: Module '<name>' has different processor type."

リンクするオブジェクト・ファイルのプロセッサ・モデルが現在のリンカ・プロセッサ・モデルと対応していません。.

"Error L011: Public symbol '<name>' defined in both modules '<name 1>' and '<name 2>'."

"Error L012: Can't resolve external symbol '<name>' (module '<name>')."

外部定義されているシンボルがモジュールで参照されていますが，プロジェクトに含まれるオブジェクト・ファイルやライブラリでは定義されていません。

"Error L013: Segment '<name>' (module '<name>') can't be allocated - <cause>"

現在のシステム・モデルにセグメントを割り当てる空き領域がありません。

DMA バンクに十分な空きメモリがありません。

十分な空き DMA バンクがありません。

セグメントを DMA アクセス可能領域に割り当てることができません。

"Error L014: Overlapped memory area <memory type> <start>:<end> in current model"

モデル定義ファイルではメモリ領域を重複してはいけません。

"Warning L015: Overlapped area <start>:<size> between segments '<segment name>' and '<segment name>'."

"Warning L016: Call of function '<name>' from '<name>' exceeds limit 15 nested calls."

このワーニングは Linker Settings ダイアログの C code analysis チェック・ボックスをオンにしたときに出力されます。

"Warning L017: Register call on address '<address>' in segment '<name>' (module '<name>')."

レジスタの呼び出しが C **関数の静的スタック・フレームの重複を無効にしています。このワーニングは** Linker Settings **ダイアログの** C static stack overlay **チェック・ボックスをオンにしたときに出力されます。**

"Warning L018: Function '<callee>' is recursive called from function '<caller>'."

"caller"**関数の連続する呼び出しが検出されました。したがって，ローカル変数は静的スタックではなく動的スタックを使用してください。このワーニングは** Linker Settings **ダイアログの** C static stack overlay **チェック・ボックスをオンにしたときにのみ出力されます。**

"Warning L019: Can't find 'main_code' segment."

プログラムの呼び出し系統を分析しても開始アドレスが見つかりません。このワーニングは Linker Settings **ダイアログの** C code analysis **チェック・ボックスと** C static stack overlay **チェック・ボックスの両方またはどちらかをオンにしたときにのみ出力されます。**

"Warning L020: Internal segment '< name>' (module '<name>') allocated in external memory."

"Warning L021: External segment '<name>' (module '<name>') allocated in internal memory."

（外部に対する）内部命令セグメントの割り当てに問題が発生し，リンカがこのセグメントを（内部メモリに対する）外部メモリに割り当てたときにワーニング L020 **と** L021 **を出力します。この形式で割り当てるのは命令メモリ・セグメントだけです。**

"Warning L022: Segment　'< name>' (module '<name>') was allocated in bank '<banknumber>'."

セグメントを指定以外のバンクに配置すると，このワーニングを出力します。

"Warning L023: Jump/Call in segment '< name>' (module '<name>'), address '<address>' to segment '< name>' (module '<name>'), address '<address>'."

モジュール間でジャンプや呼び出しを行うと，このワーニングを出力します。

"Warning L024: Register Jump/Call in segment '<name> ', address '<address> '."

"Warning L025: Preload module class '<name>' could not be allocated in the bank <address>."

"Error L026: Could not allocate all preloaded segments in the class '<name>' into one bank."

"Error L027: Preload module class '<name>' could not be allocated on the address <address>."

"Error L028: Device limitation on instruction in module <name>, segment <name>, address <address>: <name>"

"Error L029: Can't find segment <name> in module <name>."

"Error L030: Syntax error in segment order file <name>, line <line-nr>."

"Error L031: Failed to allocate DMA class '<name> - Not enough free DMA banks."

"Warning L032: DMA class <name> uses more than one DMA bank."

"Error L033: Can't allocate DMA class <name> - There is no DMA accessible memory in the current model."

"Warning L034: Segment <name> (module <name>) specified in segment order file does not exist. Segment ignored."

"Warning L035: Non-relocatable segment <name> (module <name>) specified in segment order file can't be ordered. Segment ignored."

"Error L036: Too many segments to be ordered specified in segment order file <name>, line <line-nr>."

"Warning L037: Segment <name> (module <name>) specified in segment order file <name>, line <address>, already specified for ordering. Segment ignored."

"Warning L038: Segment <name> (module <name>) of DMA class <name> uses more than one DMA bank."

"Error L039: Can't allocate class <name>. Segment <name> (module <name>) declared with conflicting memory type."

"Warning L040: Class <name> does not contain any segments."

"Error L041: Can't allocate class <name>. Segment <name> (module <name>) declared with conflicting segment

attributes."

"Error L042: Segment <name> (module <name>) of DMA class <name> could not be allocated inside DMA accessible memory area."

"Error L043: Can't allocate class <name>. Segment <name> (module <name>) declared for absolute allocation"

"Error L044: Can't allocate class <name>. Segment <name> (module <name>) declared for allocation in external memory."

"Error L045: Can't allocate class <name>. Segment <name> (module <name>) declared for allocation in ROM."

"Error L046: Can't allocate class <name>. Segment <name> (module <name>) not explicitly declared for allocation in RAM."

"Error L047: Can't allocate class <name>. Segment <name> (module <name>) is not of type instruction memory."

"Error L048: Can't allocate class <name>. Module '__NEW__' not found."

"Error L049: Can't allocate class <name>. Segment <name> (module <name>) declared for aligned allocation"

"Error L050: Can't allocate overlay class <name> with segment <name> (module <name>). Class members are not in the same DMA class."

"Error L051: Reserved module name '__NEW__' found in REL-files."

"Error L052: Boot class <name> starts on address below 0x200. Can't create HEX-file for host-reset-boot."

"Error L053: Can't allocate class <name>. Segment <name> (module <name>) is of type instruction memory."

"Error L054: Can't allocate class <name>. Segment <name> (module <name>) is specified more than once."

"Warning L055: Segment attributes for segment <name> (module <name>) ignored. Segment does not exist."

"Warning L056: Segment <name> (module <name>) is allocated (combined) relocatable."

"Warning L057: Preload module class <name> could not be aligned on cache block boundary."

"Warning L058: Preload module class <name> exceeds cache bank size."

"Error L059: Can't allocate Preload module class <name> - There is no cacheable memory in the current model."

"Error L060: Reset boot class <name>could not be (completely) allocated inside instruction memory area with reset vector."

このエラーはすべてのリセット・ブート・クラスのセグメントがリセット・ベクトル（0x200）のアドレスを持つ命令メモリ領域に割り当てられなかったときに出力します。リセット・ブート・クラス全体を必ずリセット・ベクトル（0x200）のアドレスを持つ命令メモリ領域に割り当ててください。リセット・ブート・クラス全体が命令メモリ領域に収まらない場合は，リセット・ブート・クラスをリセット・ブート・クラスとリブート・ブート・クラスに分割してください。リセット・ブート・クラスのセグメント数はリセット・ベクトルを持つ命令メモリ内に割り当てられるようにできるだけ少なくしてください。リブート・ブート・クラスは明示的にリブートしてください。

ワーニング L061: **将来的に利用するためのコードです。**

モジュールのライブラリ名の有用性によっては，２つの異なるワーニング L062 **が表示されます。**

"Warning L062: Instruction memory segment <name> (module <name>) is not a member of any boot class."

"Warning L062: Instruction memory segment <name> (module <name>, library <name>) is not a member of any boot class."

このワーニングは指定したセグメントが利用可能なブート・クラス（リセット・ブート・またはリブート・ブート）のメンバでないために，プロセッサのリセットまたはブート後は利用できなくなる場合に出力されます。指定したセグメントがライブラリからリンクされると，ライブラリのファイル名もワーニングの中に出力されます。

このワーニングが出力されたらコードは正しくリンクされますが，ワーニングの原因となったセグメントはプロセッサのリセット・ブートと次に続くかもしれないリブート・ブート後は利用できなくなります。

ワーニング L063: **将来的に利用するためのコードです。**

"Error L064: Can't allocate segment <name> (module <name>) in address range <start address>-<end address>."

このエラーは一定のアドレス範囲にセグメントを割り当てることができない場合に出力されます。ブート・クラスのメンバであるセグメントをブート・クラス全体が収まるサイズの命令メモリのアドレス内に割り当てる必要がある場合は，リンカが自動的に一定のアドレス範囲を選択します。

"Error L065: Reset boot class <name>does not contain address of reset vector (0x200)."

このエラーはリセット・ブート・クラスのメンバであるどのセグメントも命令メモリのアドレス 0x200 **（リセット・ベクトルのアドレス）に割り当てられていない場合に出力されます。リセット・ブート・クラスには，アドレス** 0x200 **に割り当てる命令を必ず** 1 **つは入れてください。**

"Error L066: Forbidden class name for class <name>"

このエラーはクラスに指定した名前の識別子が正しくない場合や，その名前がほかに使われている場合に出力されます。

"Warning L067: Boot class <name> does not need <number> fragments. <number> fragment(s) are empty"

このワーニングは命令メモリのフラグメント数の設定値が高すぎる場合に出力されます。データがないブート・データ・フラグメントがある場合は，空のブート・データ・フラグメントの処理やブートをしないために，命令メモリのフラグメント数を減らしてください。

"Error L068: Boot class <name> does not fit into <number> instruction memory fragment(s)"

このエラーはブート・クラスのセグメントが指定した数のインストラクション・メモリのフラグメントに収まらないときに出力します。指定したプロセッサに複数のインストラクション・メモリ領域がある場合は（μ PD77110 **など），インストラクション・メモリのフラグメント数が多くなり，リンカがブート・クラスを複数のフラグメントに分割します。インストラクション・メモリのフラグメント数が増えた場合，追加作成したブート・データ・フラグメントも明示的にブートしてください。**

"Error L069: Identical output filenames. Classes <name> and <name> use the same HEX-filename: <name>"

このエラーは異なるクラスに同じ HEX **ファイル名が指定されたときに出力されます。**

"Error L070: <name> - HEX-filename used by class <name> too long for operating system"

このエラーは指定した HEX **ファイル名が長すぎるためにオペレーティング・システムが保存できない場合に出力されます。**

"Warning L071: Boot data for class <name> is empty (no segments allocated in RAM)"

このワーニングはブート・クラスにブートされる（RAM **に割り当てられた）セグメントがないため，ブート・クラスにブート・データが作成されなかったときに出力されます。**

保守／廃止

★ # 付録 C　モデル

C.1 概　　要

μPD77016 ファミリのプロセッサ用プログラムの開発はモデル・ファイルが基本になります。このモデルは目的の環境に関する基本情報を格納するための特別なシンタックスを使うファイルです（拡張子". model"で見分けられます）。目的の環境の情報は，外部メモリ領域のほかにプロセッサの属性を含むタイプなどで構成されます。

WB77016 のパッケージに含まれるモデル・エディタを使うと，手入力でモデル定義ファイルを編集せずに新しいモデルを作成したり既存のモデルを変更することができます。

モデル・エディタは主に次の 2 つの部分から成ります。

・Model File Wizard

WB77016 内からモデル・ファイルの作成，複製，編集ができます。Model Wizard は Select Model ダイアログの"Model Wizard"ボタンをクリックすると起動します

・シェルの拡張機能

登録されたタイプのその他のファイルと同じように，モデル・ファイルの作成，移動，コピー，ショートカット，名前の変更，編集ができます。これは Windows エクスプローラからか直接デスクトップ上で操作できます。

C.2 モデルとディバグ

モデルには目的の環境の基本情報が含まれます。目的の環境の情報には次の項目があります。

・一意のモデル名
・プロセッサのタイプ
・クロックのスピード
・X 外部データ・メモリおよび Y 外部データ・メモリのメモリ・ブロック（任意）
・外部インストラクション・メモリのメモリ・ブロック（任意）
・モデルがロードされたり処理がリセットされるたびに HSM77016 や ID77016 が実行する式（任意）
・モデルの記述（任意）
・ID77016 にはさらにプログラムが実行される環境とチップに関係する情報が必要です。この追加情報はいわゆる“ ディバッギ ”にまとめられます。その内容は次のとおりです。
・一意のディバッギ名
・モデルの基本となる以下の項目
　・リンク名（リンクは目的のハードウエアとの通信手段です。利用できるリンク名を示した一覧が目的のハードウエアのドライバで表示されます）
　・追加のバイナリ情報（目的のハードウエアでホストがブートできるかどうかといった情報）
　・ディバッギをリセットするたびに実行される初期化の式（任意）
　・ディバッギは既存のモデルを基本としているため，基本のモデルのファイル内にあります。

C.2.1　モデル・ファイル

モデル・ファイルには，まず 1 つのモデルとモデルの基本となるチップの記述，任意にこのモデルの基本となるディバッギが含まれます。登録されたほかの種類のファイルと同じように，エクスプローラからか直接デスクトップ上で作成，移動，コピー，名前の変更，編集ができます。シェルの拡張機能の正しい使い方は，C.4　**シェルの拡張機能**を参照してください。

C.2.2　チップ記述ファイル

チップ記述ファイルには利用できるすべてのチップの記述が含まれます。このファイルは初期設定では WB77016 インストール先と同じディレクトリにあります。

それぞれのモデル・ファイルには追加でモデルの基本となるチップの独自の記述が入ります。そのため，スクラッチから作成するモデルが基本にできるのは，チップ記述ファイルに記述された 1 つのチップだけです。複製または編集されるモデルは，チップ記述ファイルかモデル・ファイルに記述されたチップを選ぶことができます。

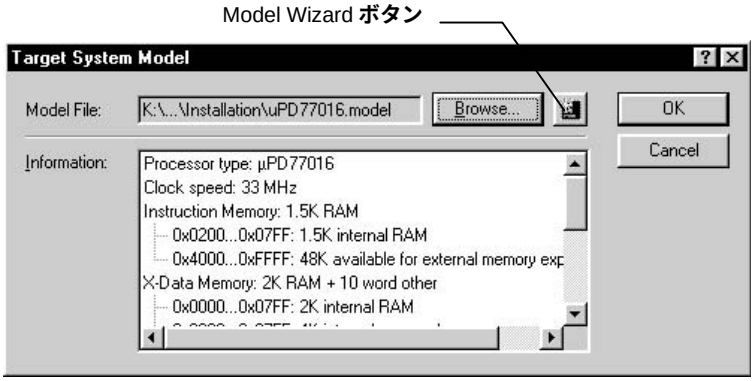
複製または編集されているモデル・ファイルとチップ記述ファイルの両方に記述されたチップを選ぶ場合は，その記述が有効な間はモデル・ファイル内のチップの方が選ばれます。両方のチップの記述が有効な場合，ユーザはどちらかを選択するよう要求されます。

C.3　Model Wizard

Model Wizard は WB77016 の Select Model ダイアログから起動します。

図 C－1 は Target System Model ダイアログのウィザードを起動するボタンを示しています。

図 C－1　Model Wizard ボタン



Model Wizard **ボタンのコマンド**

New...

Model Wizard **を起動して，チップ記述ファイル内の任意の** DSP **を基本とするまったく新しいモデルを作成します。ユーザがすべてのモデルのプロパティを指定します。**Model Wizard **が完了するときにファイル選択のダイアログが表示されますので，ここでモデル名と格納先を指定します。**

Clone...

Model Wizard **を起動して，現在アクティブなモデルを基本にした新しいモデルを作成します。複製されたモデルは，ユーザが属性を変更しないかぎり基本になったモデルの属性をすべて引き継ぎます。**Model Wizard **が完了するときにファイル選択のダイアログが表示されますので，ここでモデル名と格納先を指定します。**

Edit...

Model Wizard **を起動して，現在アクティブなモデルを変更します。新しいモデルは作成されません。現在のモデルが変更された内容に置き換えられます。モデルの複製と同じように，属性（モデル名は除く）を** 1 **つずつ変更したり削除することができ，新しい属性も追加できます。**

Rename...

アクティブなモデル定義ファイルで現在指定されているモデル名を変更するダイアログが表示されます。このコマンドが使えるのは，ディバグ処理を元のモデル定義ファイルで実行する場合のみです。

Delete

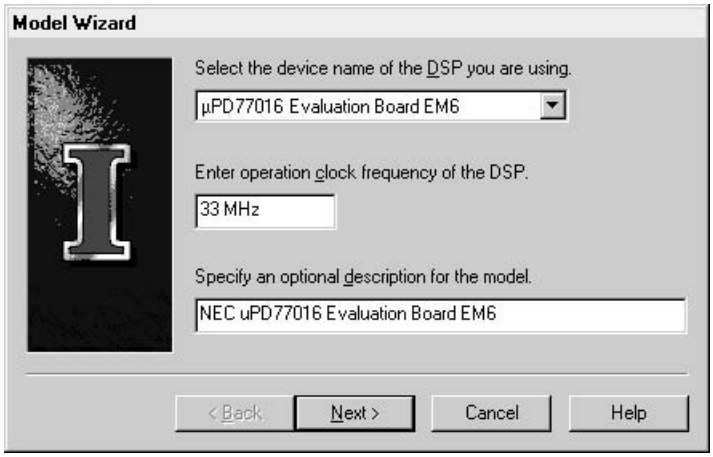
アクティブなモデル定義ファイルで現在指定されているモデルを削除します。このコマンドが使えるのは，ディバグ処理を元のモデル定義ファイルで実行する場合のみです。モデル定義ファイルから削除したモデルは復元できません。

C.3.1　Model Wizard **の** I **ページ（**DSP **の選択）**

このページはモデルの基本属性に，そのモデルの基本となるプロセッサのタイプとクロック周波数を指定できます。

DSP **の選択ページを図** C－2 **に示します。**

図 C－2　DSP **の選択ページ**



DSP の選択ページの構成

Select the device name of the DSP you are using

表示されたμ PD77016 ファミリの一覧から目的の DSP を選択します。複製または編集したモデルの場合は，このフィールドに現在アクティブなモデルの基本となったプロセッサが表示されます。新しいモデルの場合はこのフィールドに何も表示されませんが，新しいモデルに対する目的の DSP として，一覧から任意のプロセッサを選ぶことができます。プロセッサ・タイプの一覧には，現在アクティブなモデル定義ファイルで指定されたタイプだけが表示されます。

Enter operation clock frequency of the DSP

指定したターゲット DSP のクロック周波数を入力します。使用できる周波数の単位は，Hz，kHz，MHz，GHz です。

Specify an optional description for the model

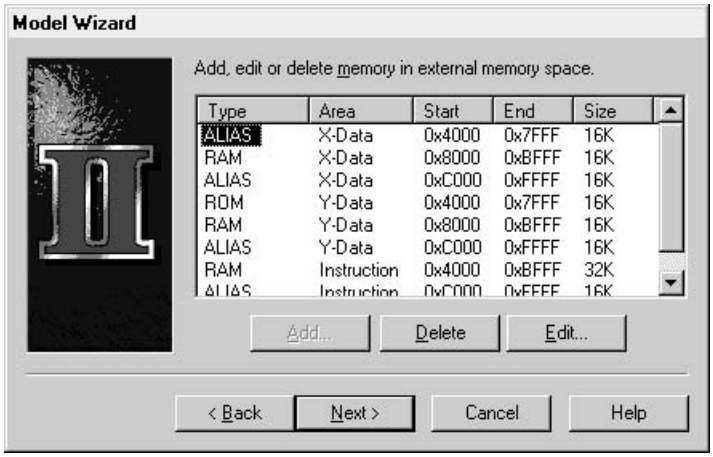
モデルに対する備考や補足説明を入力するためのフィールドです。

C.3.2　Model Wizard の II ページ（外部メモリの設定）

このページは現在アクティブなモデルに設定されている外部インストラクション・メモリ・ブロック，X データ・メモリ・ブロック，Y データ・メモリ・ブロックおよび空きメモリ・ブロックを表示します。モデルのメモリ・ブロックはメモリ・ブロックの追加や削除や編集で変更できます。プロセッサのタイプが変更されると，自動的に現在のメモリ領域が適用されます。問題が発生した場合は，競合しているメモリ領域を除去すればウィザードを続行できます。

外部メモリの設定ページの内容は図 C－3 のとおりです。

図 C－3　外部メモリの設定ページ



外部メモリの設定ページの構成

Memory block list

この一覧はメモリのタイプ，メモリ空間，開始および終了アドレス，ブロック・サイズなどのメモリ・ブロックの情報を表示します。

Add...

このボタンをクリックすると，メモリ・ブロックを追加して，ブロックのアドレス，タイプの属性，メモリへのアクセスの待機サイクルなどの属性を設定できます。空きメモリ・ブロックが少なくとも 1 つはある場合のみ，このボタンは有効です。

Delete

選択されているメモリ・ブロックを一覧から削除できます。空きメモリ・ブロックを選択すると，このボタンは無効です。

Edit

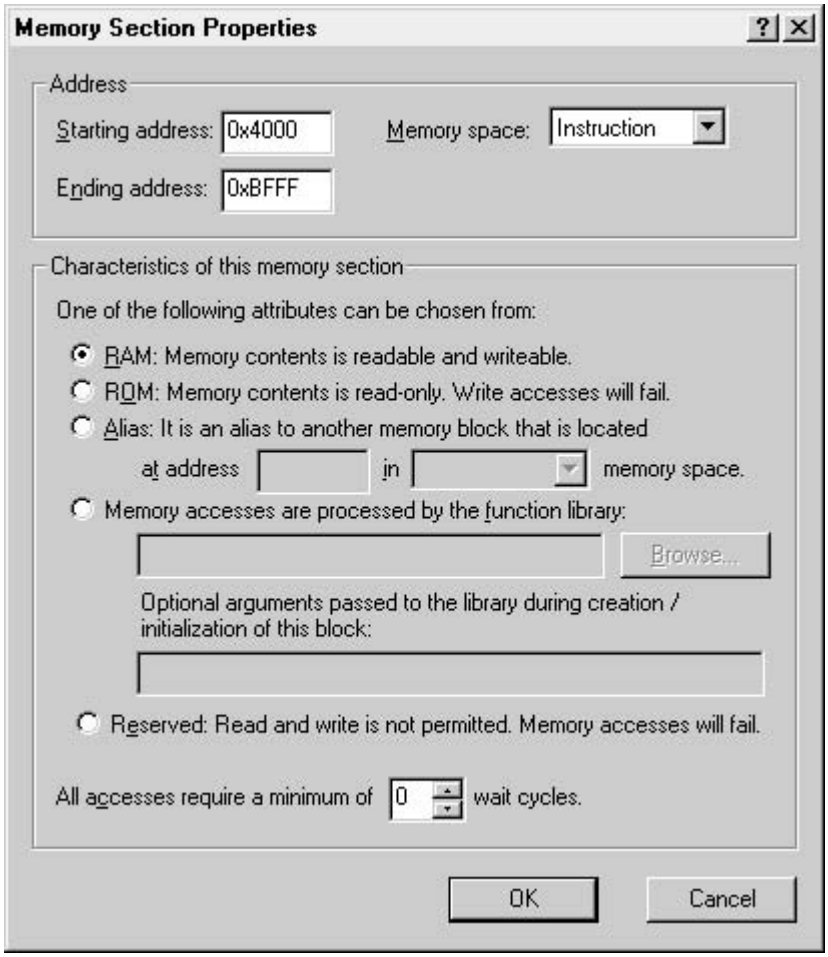
選択したメモリ・ブロックを編集して，メモリ・ブロックの属性を変更できます。メモリ・ブロックを 1 つ選択した場合のみ，このボタンは有効です。

C.3.3　Memory Section Properties ダイアログ

このダイアログを使うと，選択した外部インストラクション・メモリまたはデータ・メモリのブロックの属性を編集したり変更できます。

Memory Section Properties **ダイアログを図** C－4 **に示します。**

図 C－4　Memory Section Properties **ダイアログ**



Memory Section Properties **ダイアログの構成**

Starting address
メモリ・ブロックの開始アドレスを入力します。

Ending address
メモリ・ブロックの終了アドレスを入力します。

Memory space
目的のメモリ空間をドロップダウン・リストから選択します。ここには選択されているプロセッサのタイプに合ったメモリ空間が表示されます。

RAM
クリックすると，RAM **にメモリ・ブロックが設定されます。**

ROM
クリックすると，ROM **にメモリ・ブロックが設定されます。**

Alias
クリックすると，指定したアドレスに配置されたメモリ・ブロックに対する別名でメモリ・ブロックが設定されます。別名で定義するメモリ・ブロックの開始アドレスを入力し，そのブロックを配置するメモリ空間を選択します。一覧に表示されるメモリ空間は，編集しているメモリ・ブロックに対して指定されたメモリ空間によって変わります。

Memory accesses are processed by the function library
クリックすると，指定したメモリ・ブロックが外部ライブラリで処理されます。指定したメモリ・ブロックへのメモリ・アクセスに使う外部ライブラリの名前とパスを入力するか，"Browse..."**ボタンをクリックして外部ライブラリを閲覧します。ライブラリを表示させたら，指定した外部ライブラリに対して処理されるコマンド・ラインの引数を指定します。**

Reserved
クリックすると，指定したメモリ・ブロックを予約されたものとして区別します。予約したメモリ・ブロックにアクセスすると，エラー・メッセージが表示されます。

Wait cycles
指定したメモリ・ブロックに必要な待機サイクルの回数を入力します。スピン・ボックスをクリックして待機サイクルの値（0〜255**）を設定します。**

OK
ダイアログを閉じて変更内容を適用します。

Cancel
変更内容を適用せずにダイアログを閉じます。

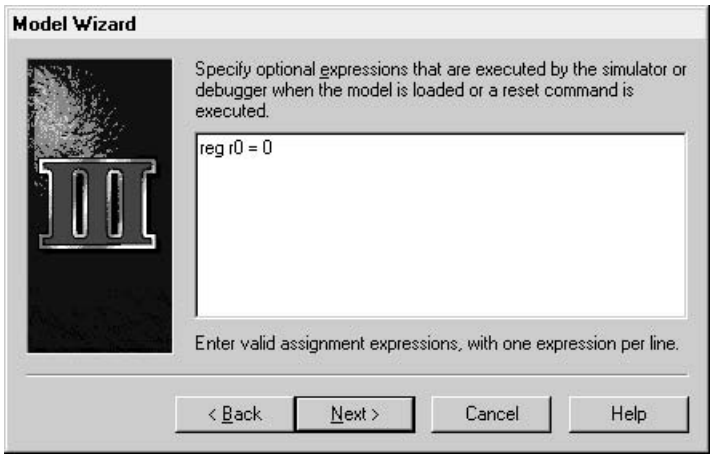
保守／廃止

C.3.4　Model Wizard の III ページ（スタートアップ処理の式）

このページを使って目的のプロセッサをリセットしたときに実行する割り当て式を指定します。式の編集フィールドの改行は Ctrl + Enter キーを押します。

スタートアップ処理の式ページを図 C－5 に示します。

図 C－5　スタートアップ処理の式ページ

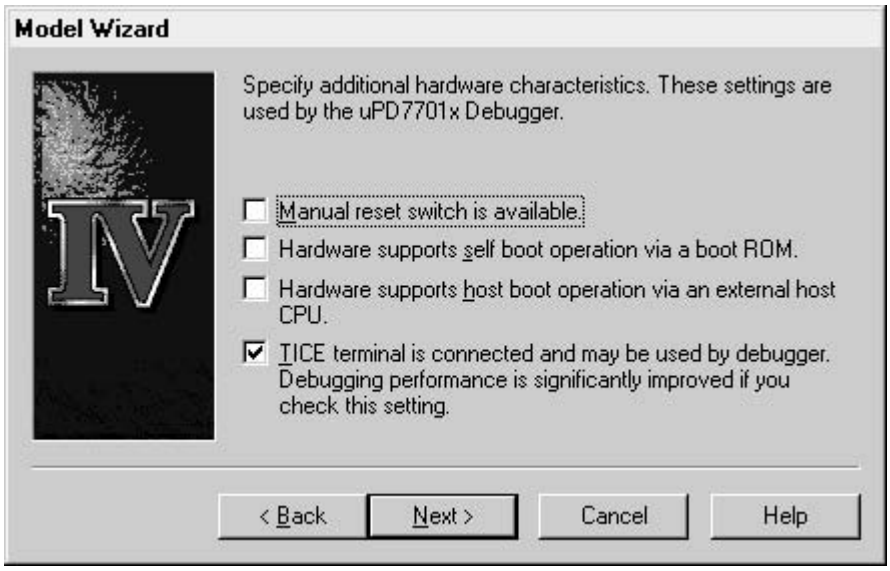


C.3.5　Model Wizard の IV ページ（ディバッガ装置の設定）

このページは ID77016 に接続したターゲット・システムに関する設定を指定します。

ディバッガ装置の設定ページを図 C－6 に示します。

図 C－6　ディバッガ装置の設定ページ



ディバッガ装置の設定ページの構成

Manual reset switch is available

ID77016 **に接続した装置のボードを手入力でリセットする場合は，このチェック・ボックスをオンにします。この設定によって** Select Start-Up Options **ダイアログに表示される選択肢が変わります。**Select Start-Up Options **ダイアログは，**ID77016 **の起動時や目的のシステムのプロパティがディバグ処理中に変更されたときに表示されます。このチェック・ボックスをオンにすると，**Select Start-Up Options **ダイアログの**"Wait for externally applied reset signal"**が有効になります。**

Hardware supports self boot operation via a boot ROM

ID77016 **に接続した装置のボードにブート** ROM **がある場合は，このチェック・ボックスをオンにします。この設定によって** Select Start-Up Options **ダイアログに表示される選択肢が変わります。**Select Start-Up Options **ダイアログは，**ID77016 **の起動時や目的のシステムのプロパティがディバグ処理中に変更されたときに表示されます。このチェック・ボックスをオンにすると，**Select Start-Up Options **ダイアログの**"Boot from external memory or host processor"**が有効になります。**

Hardware supports self boot operation via an external host CPU

ID77016 **に接続した装置のボードを外部メモリかホスト・プロセッサからブートできる場合は，このチェック・ボックスをオンにします。この設定によって** Select Start-Up Options **ダイアログに表示される選択肢が変わります。**Select Start-Up Options **ダイアログは，**ID77016 **の起動時や目的のシステムのプロパティがディバグ処理中に変更されたときに表示されます。このチェック・ボックスをオンにすると，**Select Start-Up Options **ダイアログの**"Boot from external memory or host processor"**が有効になります。**

TICE terminal is connected

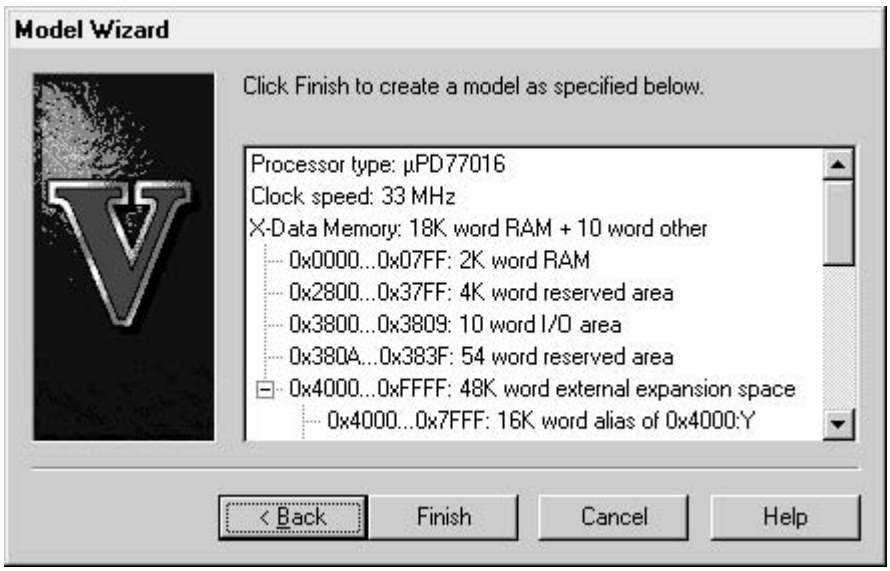
JTAG **ポートの** TICE(in-circuit-emulation)**ピンが** JTAG **ラインに接続されている場合は，このチェック・ボックスをオンにします。**ID77016 **に** TICE **ピンを使うと，ディバグ処理の性能が向上します。**

保守／廃止

C.3.6　Model Wizard の V ページ（モデルの概略）

このページは Model Wizard で作成したモデルやシェルの拡張機能のプロパティ・ダイアログで編集したモデルの概略を表示します。この内容には，そのモデルの基本となったプロセッサのタイプ，プロセッサのクロック周波数，インストラクション・メモリやデータ・メモリとその開始アドレス，ブロックのサイズとメモリの種類があります。表示されている内容に変更がない場合は，"Finish"ボタンをクリックします。新しいモデルや変更されたモデルが現在有効なモデル・ファイルに設定されて使える状態になります。モデルの概略ページを図 C－7 に示します。

図 C－7　モデルの概略ページ

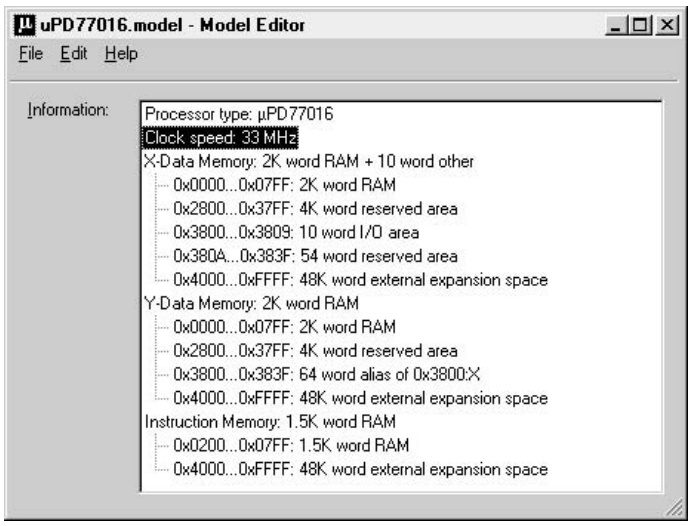


保守／廃止

C.4　シェルの拡張機能

WB77016 のアプリケーションを起動せずにモデルの変更をすることができます。Windows エクスプローラで表示したモデルの中から目的のモデルを右クリックし，ショートカット・メニューの"Open"コマンドをクリックすると，図 C－8 に示す Model Editor のメイン・ウインドウが表示されます。このウインドウでここまでに説明したすべてのモデル・ファイルやモデル定義ファイルを操作できます。

図 C－8　Model Editor のメイン・ウインドウ



保守／廃止

付録 D　WB77016 のキー

D.1　コマンド･キー

WB77016 のメニュー･コマンドは，Alt キーとメニュー･コマンドの下線付き文字を同時に押すと利用できます。コマンドの中には同じメニュー･コマンドの右にキーボード･ショートカットが表示されているものもあります。

・共通の方向キー

キー	移動先
←	1 文字左へ移動する。
↑	1 文字右へ移動する。
Ctrl＋←	1 単語左へ移動する。
Ctrl＋→	1 単語右へ移動する。
↑	1 行上へ移動する。
↓	1 行下へ移動する。
PageUp	1 画面上へ移動する。
PageDown	1 画面下へ移動する。
Home	行頭へ移動する。
Ctrl+Home	文書の先頭へ移動する。
End	行末へ移動する。
Ctrl+PageUp	ファイルの先頭へ移動する。
Ctrl+PageDown	ファイルの末尾へ移動する。

・ブロックの選択範囲

キー	選択範囲
Shift＋←	カーソルの左側の 1 文字。
Shift＋→	カーソルの右側の 1 文字。
Shift＋Ctrl＋←	カーソルの左側の 1 単語。
Shift＋Ctrl＋→	カーソルの右側の 1 単語。
Shift＋Home	行の先頭まで。
Shift＋End	行の末尾まで。
Shift＋Ctrl＋PageUp	ファイルの先頭まで。
Shift＋Ctrl＋PageDown	ファイルの末尾まで。
Shift＋→	次の行の同じ列まで。
Shift＋←	前の行の同じ列まで。
Shift＋PageDown	1 画面下まで。
Shift＋PageUp	1 画面上まで。

保守／廃止

・挿入および削除コマンド

キー	目的の操作
Del	カーソルの右側の 1 文字を削除する。
Backspace	カーソルの左側の文字を削除する。
Ctrl＋Y	カーソルがある位置の行を削除する。
Ctrl＋E	行末まで削除する。
Ctrl＋Enter	カーソルがある位置で行を挿入する。
Ins	上書きモードと挿入モードを切り換える。
Ctrl＋N	ファイル名を挿入する。
Ctrl＋T	時刻を入力する。
Ctrl＋D	日付けを入力する。

・エディタ・ウインドウで使うキー

キー	目的の操作
Shift+＋Del	選択したデータを切り取り，クリップボードにコピーする。
Ctrl＋Ins	選択したデータをクリップボードにコピーする。
Shift＋Ins	クリップボードのデータを貼り付ける。
Del	選択したデータを削除する。
F5	次の検索結果を表示する。
F6	前の検索結果を表示する。
Shift＋F5	次のエラーを表示する。
Shift＋F6	前のエラーを表示する。
Alt＋F5	カーソルを指定した行番号へ移動する。
Ctrl＋V	表示モードに切り換える。
Ctrl＋L	小文字に変換する。
Ctrl＋U	大文字に変換する。
Ctrl＋Shift＋（1～10 までの番号）	位置マーカーを設定する。
Ctrl＋（1～10 までの番号）	位置マーカーを検索する。
Ctrl＋F	一致する内容を検索する。

・メッセージ・ウインドウで使うキー

キー	目的の操作
Shift＋F5	次のエラーを表示する。
Shift＋F6	前のエラーを表示する。

・プロジェクト・ウインドウで使うキー

キー	目的の操作
Ins	ファイルをプロジェクトに追加する。
Del	選択したファイルをプロジェクトから削除する。

・ヘルプ･コマンド

キー	目的の操作
Ins	ファイルをプロジェクトに追加する。
Del	選択したファイルをプロジェクトから削除する。

D.2　ショートカット･キー

WB77016 のメニュー･コマンドは，Alt キーとメニュー･コマンドの下線付き文字を同時に押すと利用できます。コマンドの中には同じメニュー･コマンドの右にキーボード･ショートカットが表示されているものもあります。

Editor Settings ダイアログではクラシック･ワークベンチ･ショートカット（Classic）と Microsoft 互換ショートカット（Microsoft）のどちらを使うかを選びます。Classic Shortcuts チェック･ボックスをオンにすると，WB77016 オリジナルのショートカット･キーを使えます。Microsoft Compatible Shortcuts のメニュー･コマンドを選ぶと，Microsoft のショートカット･キーを使えます。

・File メニューのコマンド

Classic	Microsoft	メニュー項目	目的の操作
-	Ctrl＋N	File | New	新しいファイルを作成する。
F3	Ctrl＋O	File | Open...	ファイルをオープンする。
F2	Ctrl＋S	File | Save	ファイルを上書き保存する。
Shift+F2	-	File | Save As...	ファイルに名前を付けて保存する。
-	Ctrl＋P	File | Print	ファイルを印刷する。
Ctrl+F4	-	File | Close	ファイルをクローズする。
Alt+F4	Alt＋F4	File | Exit	WB77016 に戻る。

・Edit メニューのコマンド

Classic	Microsoft	メニュー項目	目的の操作
Shift＋Del	Ctrl＋X	Edit | Cut	選択したテキストを切り取り，クリップボードにコピーする。
Ctrl＋Ins	Ctrl＋C	Edit | Copy	選択したテキストをクリップボードにコピーする。
Shift＋Ins	Ctrl＋V	Edit | Paste	テキストをクリップボードから貼り付ける。
Del	Del	Edit | Delete	選択したテキストをエディタで削除する。
Ctrl＋A	Ctrl＋A	Edit | Select All	プロジェクト・ウインドウのプロジェクト項目をすべて選択する。 エディタ・ウインドウのテキストを選択する。

• Search **メニューのコマンド**

Classic	Microsoft	メニュー項目	目的の操作
Alt＋F3	-	Search | Find	Find **ダイアログを表示し，エディタ・ウインドウで指定した文字列を検索する。**
F5	F3	Search | Next	**次の検索結果を表示する。**
F6	Shift＋F3	Search | Previous	**前の検索結果を表示する。**
-	Ctrl＋H	Search | Replace	**ダイアログを表示し，エディタ・ウインドウで指定した文字列を検索して置換する。**
Shift+F5	F4	Search | Next error	**次のエラーを表示する。**
Shift+F6	Shift＋F4	Search | Previous error	**前のエラーを表示する。**
Alt+F5	Ctrl＋G	Search | Goto line...	**カーソルを指定した行番号へ移動する。**

• Make **メニューのコマンド**

Classic	Microsoft	メニュー項目	目的の操作
F7	Ctrl＋F7	Make | Assemble	**現在開いているエディタ・ウインドウのファイルやプロジェクト・ウインドウで強調表示されたファイルをアセンブルする。**
Shift＋F7	Shift＋F7	Make | Link	**ファイルをアセンブルせずにリンクする。**
F8	F7	Make | Make	**プロジェクトを更新する。**
Shift＋F8	-	Make | Build all	**プロジェクト全体を再構築する。**
-	F5	Make | Simulate	**プログラム作成処理を開始する。**

• Project **メニューのコマンド**

Classic	Microsoft	メニュー項目	目的の操作
F9	-	Project | Open project	**プロジェクトをオープンする。**
Shift＋F9	-	Project | Close project	**プロジェクトをクローズする。**
Ins	-	Project | Add item...	**項目をプロジェクト･ファイルに追加する。**
Del	-	Project | Delete item	**項目をプロジェクト･ファイルから削除する。**

• Options **メニューのコマンド**

Classic	Microsoft	メニュー項目	目的の操作
Alt＋F7	Alt＋F7	Options | Edit segments	Edit Segments **ダイアログを開き，セグメントを適切な設定に変更する。**

• Window **メニューのコマンド**

Classic	Microsoft	メニュー項目	目的の操作
Shift+F10	-	Window | Tile	ウインドウを重ねずに並べて表示する。
F10	-	Window | Cascade	ウインドウを重ねて表示する。
Alt＋F10	-	Window | Arrange Icons	アイコンを整列する。
Ctrl＋M	Alt＋0	Window | Message	メッセージ・ウインドウをオープンする，または一番上に重ねて表示する。
Ctrl＋P	Alt＋2	Window | Project	プロジェクト・ウインドウを一番上に重ねて表示する。

• Help **メニューのコマンド**

Classic	Microsoft	メニュー項目	目的の操作
Shift＋F1	Shift＋F1	Help | Index	トピックを表示する。
Ctrl＋F1	Ctrl＋F1	Help | Topic Search	カーソル位置の語に対するヘルプを表示する。

［メ　　モ］

保守／廃止

★ 付録 E　μPD77016 ライブラリアン

E.1 概　　要

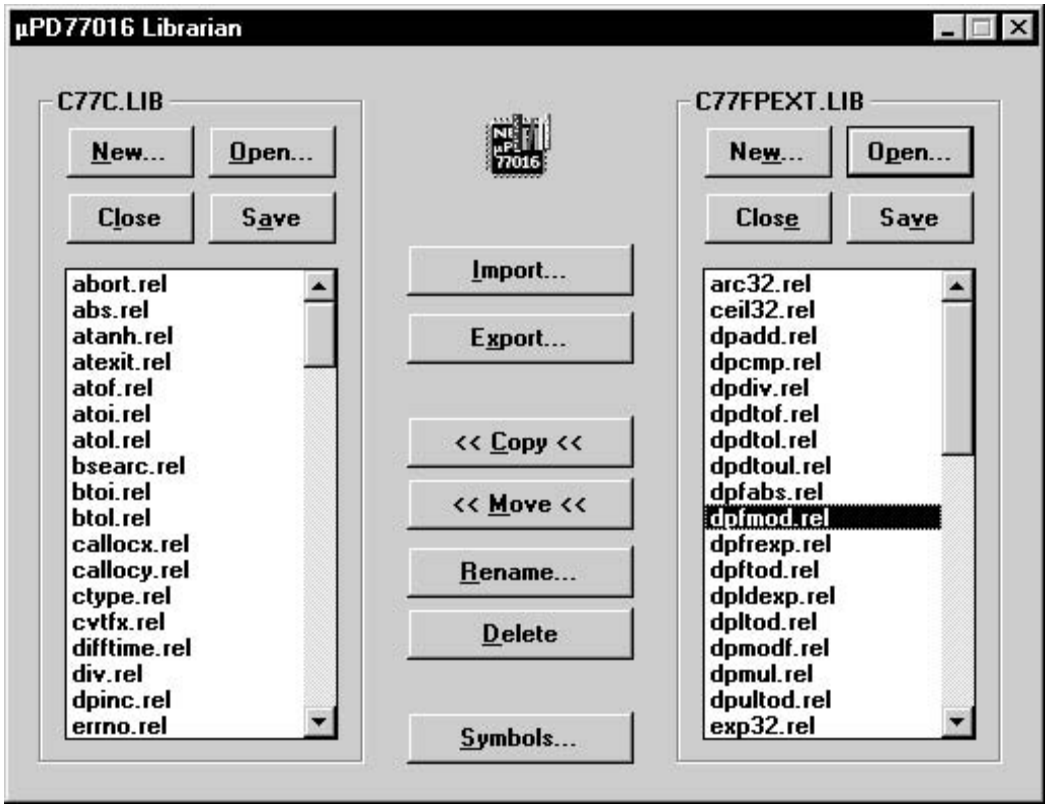
μPD77016 ライブラリアン（以下ライブラリアンといいます）は，μPD77016 ファミリ用プログラム開発のライブラリの作成，維持，一覧表示を行うツールです。このツールは既存のライブラリを利用したり，新しいライブラリの作成にも使えます。

ライブラリアンをすばやく操作できるように，メニューがない設計になっています。

ライブラリアンには 2 つのリソース・ブラウザがあり，ここからライブラリ・ファイル（拡張子“.LIB”）を開きます。2 つのブラウザがあるため，両方を同時に開き，モジュールを一方のライブラリからもう一方へすばやくコピーすることができます。

ライブラリアンを図 E−1 に示します。

図 E−1　ライブラリアン



保守／廃止

ライブラリアンの構成

Import

このボタンをクリックするとファイル選択のダイアログが表示され，既存のリロケータブル・アセンブラ・ファイル（拡張子 “ .REL ”）からモジュールをインポートできます。

Export

このボタンをクリックするとファイル選択のダイアログが表示され，モジュールをリロケータブル・アセンブラ・ファイルへエクスポートできます。

Copy

選択したモジュールを一方のライブラリ（モジュールが選択されているライブラリ）からもう一方のライブラリ（反対側のリソース・ブラウザで開かれているライブラリ）へコピーします。コピーの方向は Copy **ボタンの矢印で示されます。この操作を実行できるのは** 2 **つのライブラリが同時に開かれている場合のみです。**

Move

選択したモジュールを一方のライブラリからもう一方のリソース・ブラウザで開かれているライブラリへ移動します。移動の方向は Move **ボタンの矢印で示されます。この操作を実行できるのは** 2 **つのライブラリが同時に開かれている場合のみです。**
この操作を実行すると，選択したモジュールが元のライブラリから削除されます。

Rename

ダイアログで選択したモジュールの名前を変更できます。

Delete

選択したモジュールをライブラリから削除します。この場合，削除を確認するためのダイアログは表示されません。

Symbols

選択されているモジュールに含まれるすべてのグローバル・シンボルを一覧表示するダイアログが表示されます。

New

ファイル選択のダイアログが表示され，新しいライブラリ・ファイルを作成できます。

Open...

ファイルの選択のダイアログが表示され，既存のライブラリ・ファイルをオープンすることができます。

Close

ライブラリの変更を保存してライブラリ・ファイルをクローズします。

Save

選択されているライブラリが保存されます。

保守／廃止

★ # 付録 F　リリース・ノート

F.1　バージョン 2.4（今回のリリース製品）

(1) 変更

●データ RAM 初期化セグメントのクラス

INID クラスのディフォールト・メンバの名前付け法則を，同じアセンブラ・モジュールで INID クラスのメンバであるセグメントに対して別々のセグメント属性を持たせるように改善しました。名前付け法則を改善することで，既存のディフォールト・セグメントと完全に下位互換性を持つことができます。改善した名前付け法則についての詳細は,WB77016 のオンライン･ヘルプのトピック"Data RAM Initialization Class Types" を参照してください。

(2) バグ修正

SPR #223:Typing error in the assembler language help (INFNDEF control example)

説明：アセンブラのオンライン・ヘルプで"EXAMPLE"のトピック"IF，IFDEF，IFNDEF，ELSE，ENDIF，UNDEF Control"の部分が次のように表示されます。

```
#IFDEF (literalname)
#DEFINE (literalname) (0x10)
#ENDIF
```

これは INDEF コントロールに対する例のため，次のように表示しなくてはなりません。

```
#IFNDEF (literalname)
#DEFINE (literalname) (0x10)
#ENDIF
```

状況：この例の表示を修正しました。

SPR #220:Wrong topic title in the Model Editor on-line help.

説明：トピック"Memory Selection Properties Dialog"のトピック・タイトルは"Memory Section Properties Dialog"と表示しなくてはなりません。

状況：このヘルプのトピック・タイトルとすべての参照内容を正しく修正しました。

SPR #218:Workbench on-line help contents and spelling errors.

説明：トピック"Segment Allocation Order File Example"のスペルの誤り"reloaceatable"は"relocatable"と表示しなければなりません。トピック"How to..."のスペルの誤り"continuos"は"continuous"と表示しなければなりません。

目次の修正：μPD77110 のブート・モードの特性がヘルプの目次に追加されます。

状況：スペルの誤りを修正しました。ブート・モードの説明を改善しました。

SPR #215:Unpredictable linking results when initialization ROM segment exceeds 1000 words.

説明：INI **または** INID RAM **初期化セグメント・クラスで生成される** ROM **データ・セグメントのサイズが** 1,000(0x3E8)**ワードを越える場合，リンカが予測できない結果を生成することがあります（例** ROM **セグメントのヘッダ情報が正しくない可能性があります）。**

状況：この問題を修正しました。

●ライブラリに含まれる INID **クラスのディフォールト・メンバのセグメントがデータ** RAM **初期化クラスに属していません。**

説明：ライブラリに INID **クラスのディフォールト・メンバの名前付け法則に従って名前をつけたセグメントがある場合に，これらは既存の** INID **セグメント・クラスに属していません。**

状況：この問題を修正しました。

（3）重要な注意点

●印刷プレビュー

説明：印刷プレビューが正しく表示されません。

対応：標準システム・プリンタのドライバが正しくインストールされているかどうか確認してください。

●DYNALINK **エラー／モデル・エディタの** 16 **ビット部分（**MODELDLG.DLL**）**

説明： Windows **フォルダの中にある** System **フォルダの中の古い** MODELDLG.DLL **ファイルが，プロジェクトの構築やモデル・ファイルの操作に対して** DYNALINK **エラーというメッセージを表示することがあります。**

対応：MODELDLG.DLL **ファイルを使っている可能性のあるアプリケーションをすべて閉じ，このファイルの名前を変更するかバックアップをとって** Windows **フォルダの中にある** System **フォルダから削除してください。**System **フォルダから削除すると，**WB77016 **のアプリケーションでインストールした正しい** MODELDLG.DLL **ファイルが自動的に使われます。**

●プロジェクト・ファイルやモデル・ファイルのパス名が長い

説明：属性"store relative path"**が設定されているときには，プロジェクト・ファイルやモデル・ファイルを長いパス名で保存することはお勧めしません。このようなプロジェクトを**"Save As"**コマンドで保存したり，手操作で別のフォルダに移動すると，モデル・ファイルが見つかりません。**

対応：プロジェクト・ファイルやモデル・ファイルを長いパス名を持たないパスで保存してください。

●Windows **の** COMCTL32.DLL **ファイル**

説明：モデル・エディタを Workbench Select Model **ダイアログから起動すると，**Windows95 **のインストール作業で表示の問題が発生します。**

対応：この問題を解決するには，Windows95 **の** COMCTL32.DLL **ファイルを更新する必要があります。そのために，次の手順を実行してください。**

1. WB77016 **のアプリケーションをクローズします。**
2. Windows **エクスプローラから** WB77016 **配布ディスクのディスク** 4 **の中の** COM32UPD.EXE **ファイルをダブル・クリックします。**
3. **画面上の指示に従って操作します。**

保守／廃止

(4) 確認されている問題

●モデル・エディタ

説明：モデル・ファイルをモデル・エディタで編集すると正しく保存されず，モデル・ファイルまたは主要なチップ記述ファイル UPD7701X.INI **が読み取り専用に設定されます。**

対応：主要なチップ記述ファイルや目的のモデル・ファイルから読み取り専用のファイル属性を削除してから，モデルをモデル・エディタで変更します。

F.2 バージョン 2.4 ベータ 2

(1) バグ修正

●エラーの発生数が指定した数を越えると，インクルード・ファイルが削除されます。

問題：アセンブルやリンクが終了する前にインクルード・ファイルで発生したエラー数が指定した数（Options **メニューの**"Miscellaneous..."**コマンドで指定）を越えると，そのインクルード・ファイルが削除されます。**

状況：このバグを修正しました。

●ソース・ファイルを（再び）プロジェクト・ウインドウからオープンすると，ファイルの変更内容が消失します。

問題：プロジェクト・ウインドウから開いたソース・ファイルを変更したり，ソース・ファイルを再びプロジェクト・ウインドウから開くと，ソース・ファイルの変更内容が消失します。

状況：このバグを修正しました。

F.3 バージョン 2.4 M3

(1) バグ修正

SPR #224: Wrong symbol value if location counter is used with the EQU directive.

問題：位置カウンタ ($**) を絶対セグメントの中で** EQU **命令と合わせて使うと，リンク・ファイルとマップ・ファイルの中のシンボルが誤った値になります。ただし，シンボルを使って初期化したメモリの内容は正しい値です。**

状況：リンク・ファイルとマップ・ファイルのシンボル値は正しく出力されます。

F.4 バージョン 2.4M2

(1) 変更

●ブート機能のサポート

評価版の第 2 **版はブート機能のサポートの強化を目的としています。**

(2) バグ修正

SPR #222: The Librarian application does not show an application icon in the title bar.

問題： Windows95 **または** Windows98 **環境で**μ PD77016 **ライブラリアンをインストールしても，タイトル・バーにアプリケーション・アイコンが表示されません。**

状況： Windows95 **または** Windows98 **環境でライブラリアンのアプリケーション・アイコンがアプリケーション・タイトル・バーに表示されます。**

SPR #221: Wrong SET directive is referenced when symbol is used in an allocation directive.

問題：SET **命令を使って同じシンボルを繰り返し定義したり，セグメントの配置命令でシンボルを参照すると，**SET **命令の最後の結果ではなく最初の結果が参照されます。**

状況：SET **命令の最後の結果が参照されます。**

F.5　バージョン 2.4M1

(1) 変更

●ブート機能のサポート

この評価版はブート機能のサポートの強化を目的としています。

F.6　バージョン 2.4 ベータ版

(1) 変更

●ブート機能のサポート

WB77016 **のブート機能を強化し，連続しないインストラクション** RAM **領域を持つプロセッサ（例　**μPD77110**）に対応します。**

●マスク・ファイルのサポート

WB77016 **は**μPD77113，77114，NA77113 **用にマスク・ファイル・オプションの設定を追加しました。**

●ブート・クラスと RAM **初期化クラスの** HEX **ファイルの出力**

ブート・クラスと RAM **初期化セグメント・クラスで，バイト・フォーマットとワード・フォーマットの** HEX **ファイルを出力できます。**

●パブリック・シンボルの直列型宣言

"PUBLIC"**命令を使う以外に，「**symbol_name::**」のようにコロンを** 2 **つ付けてパブリック・シンボルを宣言できます。**

●ライブラリ・ファイルの扱い

プロジェクト・ウインドウに表示されたライブラリ・ファイルのエントリをダブルクリックして，μPD77016 **ライブラリアンの各ライブラリを開くことができます。**

●ライブラリ・ファイルのドラッグ・アンド・ドロップ

Windows **エクスプローラから** 2 **つまでのライブラリ・ファイルを**μ PD77016 **ライブラリアンにドラッグ・アンド・ドロップで移動できます。それぞれのライブラリは専用のリソース・ブラウザの枠内に表示されます。**

(2) バグ修正

SPR #195: Wrong on-line help may be launched by the Workbench application

問題： Windows **フォルダの中のワークベンチ初期設定ファイル** WB77016.INI **に** WB77016 **のオンライン・ヘルプ・ファイルの位置と名前を指定する登録があると，ワークベンチのアプリケーション・フォルダの中のヘルプ・ファイルではなく，**WB77016.INI **ファイルで指定されたヘルプ・ファイルを起動しようとします。**

保守／廃止

状況：　アプリケーション・フォルダのヘルプ・ファイルの方を起動します。ヘルプ・ファイルが見つからない場合は，ユーザがオンライン・ヘルプ・ファイルの位置を指定しない限り WB77016.INI **ファイルで指定されたヘルプ・ファイルを起動します。**

SPR #192: Workbench on-line help contents error

問題：トピック"Edit Segments Dialog"**，ヘッダ**"New class..."**にスペルの誤りがあります。公式**"except μPD7116DSP"**は正しくありません。**

状況：　関連する公式をオンライン・ヘルプから削除しました。

F.7　バージョン 2.32

（1）バグ修正

SPR #191: Stop instruction causes an assembler error message

問題：　μPD77111 **または**μPD77116 **モデルを選択したときに** STOP **命令を含むアセンブラ・ソース・ファイルをアセンブルすると，アセンブラ・エラー**A088"Illegal instruction for processor type"**が表示されます。**

状況：　μPD77016 **を基本とするモデルを選択したときに** STOP **命令を含むアセンブラ・ソース・ファイルをアセンブルすると，アセンブラ・エラー**A088"Illegal instruction for processor type"**が表示されます。**

SPR #190: Simultaneous overlay and DMA class memberships are not correctly resolved

問題：1 **つのプロジェクトに複数の** DMA **クラスと複数のオーバレイ・クラスがあり，ここで複数の** DMA **クラスのセグメントが対応するオーバレイ・クラスに割り当てられる場合に，リンカがオーバレイ・クラスの構成を正しく処理しません。この場合，リンカは誤って「オーバレイ・クラスのセグメントは同じ** DMA **クラスのメンバでない」というメッセージを表示します（エラー**L050)。

状況：エラーやワーニングは表示されず，セグメントは正しく割り当てられます。

SPR #189: Link file path and name is not correctly handed over to the 16-bit simulator

問題：　"Workbench Simulate"**コマンドを** 16 **ビット・シミュレータ（**SM77016**）に対して実行すると，リンク・ファイルのパスと名前が引用した（二重引用符で囲む）形式で送られ，これを** 16 **ビット・シミュレータで認識できません。この場合，シミュレータが指定されたリンク・ファイルが見つからないことを表示します。**

状況：　"Workbench Simulate"**コマンドを使うことで，モデル・ファイル名と同じようにリンク・ファイルのパスと名前は引用符がない形式で** 16 **ビット・シミュレータへ送られ，引用符で囲む形式で高速シミュレータへ送られます。ただし，**16 **ビット・シミュレータへ長いファイル名を送る場合は，関連する** 16 **ビット用アプリケーションの長さの制限に従います。**

SPR #188: Help button in the "DMA Class" dialog does not work

問題：　DMA Class **ダイアログの** Help **ボタンをクリックしてもヘルプが表示されません。**

状況：　DMA Class **ダイアログからヘルプを表示するように修正しました。**

SPR #187: Workbench on-line help formatting, typing and contents errors

問題：　**一部のオンライン・ヘルプの表現，スペル，目次に誤りがあります。**

状況：　**オンライン・ヘルプの誤りを修正しました。**

SPR #186: Debug information is not generated correctly

問題：　同じ名前のセグメントが異なるインクルード・ファイルに存在する場合に，これらのセグメントに対するディバグ情報が正しく生成されません。

状況：　ディバグ情報が正しく生成されるように修正しました。

●プリロード・モジュール・クラスのセグメントの割り当て

問題：キャッシュできない外部メモリのアドレスに絶対的に割り当てられたインストラクション・メモリのセグメントがプリロード・モジュール・クラスのメンバの場合，セグメントが指定したアドレスに正しく割り当てられません。

状況：　リンカ・エラーL013 "Segment<segmentname> (module<modulename>) can't be allocated - segment could not be allocated in cacheable area."**が表示されます。**

F.8　バージョン 2.32 ベータ 3

(1) 変更

●μPD77116 **のブート機能のサポート**

最新のハードウエア仕様に合わせて，μPD77116 **に対するブート機能のサポートを追加しました。**

F.9　バージョン 2.32 ベータ 2

(1) 変更

●DMA **方式の採用**

最新のハードウエア仕様に合わせて，μPD77116 **に** DMA **方式を採用しました。詳細については，**WB77016 **のオンライン・ヘルプのトピック**"Segment Classes"**を参照してください。**

●セグメントの割り当て順序ファイル

このファイルはプレーンな ASCII **テキスト形式で** DSP **メモリにおけるセグメントの割り当て順序を指定できます。詳細については，オンライン・ヘルプのトピック**"Segment Allocation Order Files"**を参照してください。**

● セグメントの扱い

-Edit Segments **ダイアログでセグメントの行をカラー表示し，セグメント・クラスへの割り当てと使用可能なセグメント属性を示します。**

-**現在使用していないセグメント属性が** Edit Segments **ダイアログに表示されます。これらのセグメントはボタンをクリックして削除することができます。**

-**セグメント名，レーベル名，クラス名の** 31 **文字までという長さ制限をなくしました。**

-**自動的にライブラリのセグメントを取り込むことができる新しいブート・クラスのオプションを利用できます。**

●ユーザ・インタフェースの改善

- Edit Segments ダイアログをすばやく表示するために，新しいツール・バーのボタンを追加しました。

-WB77016 のファイルの種類をドラッグ・アンド・ドロップで操作できます。

-Windows エクスプローラから目的のファイルをダブルクリックして，WB77016 のプロジェクトをオープンすることができます。

-印刷プレビューの機能を追加しました。

-アセンブラとリンカのシンボルの出力（PRN ファイルと MAP ファイル）を 1 つのウインドウで表示できます。

-プロジェクトに関連するすべてのファイルを自動保存するオプションを利用できます。このオプションを選ぶと，プロジェクトのアセンブルやリンクを実行する前に，すべてのプロジェクト・ファイルが自動的に保存されます。

-すべてのウインドウをクローズするオプションで，プロジェクトを閉じるときに自動的にすべてのウインドウを閉じることができます。

●エラーとワーニングのメッセージ

新しい特徴に関するアセンブラとリンカのエラーやワーニングのメッセージを追加しました。

●μPD77016 ライブラリアン

ライブラリアン・アプリケーションを変更して操作性を改善しました。

●モデル・エディタ

WB77016 （32 ビット Windows 環境のみ）にモデル・エディタの新しいバージョンを採用しました。操作性がより簡単になり，モデルの扱いが改善され，またμPD77116 の XDMA メモリおよび YDMA メモリに対応します。

(2) バグ修正

SPR #182: Segment attributes are lost when a project is moved.

問題： WB77016 の Edit Segments ダイアログで指定したセグメント属性は，アセンブラで生成したオブジェクト・ファイル（REL ファイル）がプロジェクトの移動後に有効な場合のみ復元できます。プロジェクトのすべてのファイルに対して"relative path"を設定してプロジェクトを別の位置に移動すると，オブジェクト・ファイル（REL ファイル）に対する相対パスの設定が無視されます。その結果，移動後のプロジェクト位置では元のセグメント属性が無効になります。

状況：移動後のプロジェクト位置で元のセグメント属性が復元されます。

SPR #181: Wrong memory type allocation option is assigned.

問題：セグメントのメモリ・タイプの割り当てオプションの"int, ext"と"ext. int"のどちらかを指定すると，セグメントは"internal or external"オプションが指定されたかのように割り当てられます。

状況：セグメントのメモリ・タイプの割り当てオプションは正しく適用されます。これらの属性の適用順序については，オンライン・ヘルプのトピック"Segment Attributes Dialog"を参照し，ハイパーリンク"Allocation order of segment attributes"に従ってください。

SPR #180: The Librarian user interface does not respond under Windows NT.

問題： μPD77016 **ライブラリアンを** Windows NT4.0 **環境で起動すると，ユーザ・インタフェースがユーザの入力に反応しません。また，ほかの** 16 **ビット用アプリケーションと同じように** WB77016 **もこの問題の影響を受けます。**

状況： μPD77016 **ライブラリアンのユーザ・インタフェースの反応をバージョン** 2.1 **で修正しました。**

SPR #177: Boot parameters (public symbols) cannot be resolved during linking.

問題：ソース・コードで外部メモリとして参照されたブート・パラメータ（パブリック・シンボル）がリンク中に解釈されません。WB77016 **はブート・セグメント・クラスを使って**μPD7701x **ファミリのブートを行います。セルフ・ブートまたはホスト・ブートのセグメント・クラスに対してリブート・オプションを指定すると，リンカがパブリック・シンボルとしてブート・パラメータをエクスポートします。このパブリック・シンボルがソース・コードで使われ，ブート・パラメータをリブート・サブルーチンで使うレジスタに割り当てます。プロジェクトが構築されると，リンカが指定したシンボルを解釈できないというエラー・メッセージを表示します。**

状況：パブリック・シンボルとしてエクスポートしたブート・パラメータは，リンク中に正しく解釈されます。

SPR #176: Relocatable object file and XCOFF file segment information is not available.

問題： リロケータブル・オブジェクト・ファイル（アセンブラ出力ファイル）と XCOFF **ファイル（リンク・ファイル）の中のセグメントが** Edit Segments **ダイアログで認識されません。**WB77016 **のプロジェクトにはアセンブラ・ソース・ファイル，ライブラリ・ファイル，リロケータブル・オブジェクト・ファイルが混在することがありますが，**1 **つしかファイルがない場合は，プロジェクトの中のファイルはリンク・ファイルです。プロジェクトの構成要素であるオブジェクト・ファイルと** XCOFF **ファイルに含まれるセグメント情報は** Edit Segments **ダイアログに表示する必要があります。以前の** WB77016 **のバージョンではこの情報は表示されません。**

状況：リロケータブル・オブジェクト・ファイルと XCOFF **ファイルのセグメント情報が** Edit Segments **ダイアログに表示されます。**

SPR #175: Relative model file path is not stored correctly by the Save As... command.

問題：Project **メニューの**"Save As..."**コマンドでモデル・ファイルの相対パスが正しく保存されません。**WB77016 **のプロジェクトを別の場所へ移動するには，次の** 2 **つの方法があります。**

(1**) プロジェクトのソース・ファイルのパスとモデル・ファイルのパスを相対パスとして指定し，プロジェクトを保存して関連するファイルを含むプロジェクト・ファイルを別の場所へ移動します。**

(2**) プロジェクトのソース・ファイルのパスとモデル・ファイルのパスを相対パスとして指定し，**Project **メニューの**"Save As..."**コマンドを使ってプロジェクト・ファイルを別の場所に保存します。続いて関連するファイルをその場所へ移動します。**

(2**) の方法で移動した後にプロジェクト・ファイルを再び開くと，「モデル・ファイルが見つからない」というメッセージが表示されます**

状況：Project **メニューの**"Save As..."**コマンドの機能を修正しました。**

保守／廃止

F.10　バージョン 2.32 ベータ版

(1) 変更

●データ・メモリ DMA **に対応するセグメント・クラスを追加しました。**DMA **セグメント・クラスはリンカの** Edit Segments **ダイアログで**"New Class..."**ボタンをクリックすると作成できます。**

μPD77015，77017，77018，77018A，77019 **用の新しいマスク・オプションを利用できます。**μPD77111 **では独立した** Mask Option **ダイアログを利用できます。マスク・オプションは，**Hex Conversion **ダイアログでマスク・ファイルの出力を指定する**"*.MSK"**ボックスをオンにし，続いて**"Mask options..."**ボタンをクリックして設定します。利用できるマスク・オプションは，使用しているプロセッサ・モデルによって異なります。**

●ブート・セグメント・クラスで内部インストラクション RAM **と外部インストラクション** RAM **のセグメントに対する** 2 **つのディフォールト・メンバを設定できます。ブート・セグメント・クラスのディフォールト・メンバは，リンカの** Edit Segments **ダイアログで**"New class..."**または**"Class Properties"**ボタンをクリックすると表示される** Boot Class Properties **ダイアログで選択できます。**

●新しい機能に関連するアセンブラとリンカのエラーやワーニングのメッセージを追加しました。

F.11　バージョン 2.31

(1) 2.31 **ベータ** 2 **以降のバグ修正**

整列させた最大のデータ RAM **セグメントが常にアドレス** 0x0 **に割り当てられます。**

問題：あるセグメントがほかのデータ・セグメントよりも大きい場合に，データ・メモリに整列させたデータ RAM **セグメントが常にアドレス** 0x0 **に割り当てられます。このため，アドレス** 0x0 **がほかのデータ・セグメントで使われているときに，リンカ・エラーが発生することがあります。**

対応：最大のデータ RAM **セグメントは，指定した間隔で正しく配置されます。**

SPR #169: Relative include file path settings are ineffective after workbench startup.

問題：WB77016 **の起動後にインクルード・ファイルの相対パス設定**(.¥，..¥)**を含むプロジェクトが読み込まれると，これらのパス設定が無効になります。プロジェクトの保存もできません。プロジェクトは保存して再び開き，インクルード・ファイルのパス設定を回復しなければなりません。**

対応：インクルード・ファイルの相対パスの設定は，プロジェクトが読み込まれたときに認識されます。

SPR #168: Settings from a former project file are lost.

問題：WB77016 **バージョン** 2.21 **以前に生成したプロジェクト・ファイルをバージョン** 2.31 **ベータ** 2 **に読み込むと，このプロジェクト・ファイルの出力先のパスとマスク・オプションが消失します。**

対応：バージョン 2.21 **で保存した出力先のパスとマスク・オプションはバージョン** 2.31 **に正しく読み込まれます。**

SPR #167: The last line in an include file must be terminated by a CR/LF.

問題：インクルード・ファイルの最終行の行末には CR **と** LF **を付けなければなりません。**CR **と** RF **がない場合は，親ファイルにある次の**#INCLUDE **コントロールが無視されます。**

対応：インクルード・ファイルの最終行に対する特定の行末レコードは必要ありません。連続する#INCLUDE **コントロールはすべて正しく処理されます。**

SPR #166: Include file information is available only after a make or build.

問題：まだアセンブルしていないプロジェクトの中のファイルに対して Project **メニューの**"Include Files..." **コマンドを実行すると，メッセージが表示されてファイルをアセンブルし，利用できるインクルード・ファイル情報を生成します。ところがファイルをアセンブルしても，インクルード・ファイル情報が生成されません。作成や構築を実行した場合のみ，インクルード・ファイル情報が表示されます。**

対応：プロジェクトの中から選択したファイルをアセンブルした後に，インクルード・ファイル情報が表示されます。

SPR #165: Relative model file path option is ignored.

問題：Make Settings **ダイアログで相対モデル・ファイル・パス・オプションが無視されます。このオプションを選択すると，モデル・ファイルのパスが絶対パスとしてプロジェクト・ファイルに保存されます。プロジェクト・ファイルとモデル・ファイルを別の位置に移動すると，プロジェクトを読み込んだときにモデル・ファイルが見つかりません。**

対応：Make Settings **ダイアログの相対モデル・ファイル・パス・オプションが正しく適用されます。**

SPR #164: Make process cannot be aborted.

問題：プロジェクトの作成や構築の処理中に Status **ダイアログの**"Abort"**ボタンをクリックすると，フェータル・エラー**'A001: User break'**が表示され，実行中のファイルの作成や構築の処理が終了します。処理中のプロジェクトに複数のファイルがある場合は，残りのファイルで作成や構築の処理を再開します。もう一度**"Abort"**ボタンをクリックすると処理中のファイルだけが終了し，プロジェクトの作成や構築は続行します。**

対応：プロジェクトの作成や構築の処理中に Status **ダイアログの**"Abort"**ボタンをクリックすると，フェータル・エラー**'A001: User break'**が表示され，プロジェクトの作成や構築の処理が終了します。**

F.12　**バージョン** 2.31 **ベータ** 2

（1**）機能の強化**

プロジェクト・ファイルの設定を ASCII **テキスト形式で保存できます。**

（2**）** 2.31 **ベータ版以降のバグ修正**

●μ PD77116 **のモデル記述を次のように修正しました。**

ブロック・サイズ： 128 **ワード**

バンク・サイズ： 512 **バイト，**1 **バンク当たり** 8 **ブロック，**4 **バンク**

●不正なアドレス（キャッシュできない領域のアドレス）を指定すると，ワーニング・メッセージが表示されます。

●異なるプリロード・モジュール・クラスのセグメント間で検出されるジャンプや呼び出しに対して，次の規則が適用されます。

a. **異なるプリロード・モジュール・クラスのセグメント間の最初のジャンプや呼び出しだけでなく，検出したすべてのジャンプや呼び出しがワーニングとして表示されます。**

b. **同じプリロード・モジュール・クラスのセグメント間のジャンプや呼び出しは，ワーニングとして表示されません。**

●WB77016 **のバージョン** 2.21 **と** 2.31 **の間のプロジェクト・ファイルの変換を修正しました。**

●プリロード・モジュール・セグメント・クラスのメンバでないセグメントをキャッシュ可能メモリに割り当てることができます。

●オンライン・ヘルプのスペルと表現の誤りを修正しました。

F.13　バージョン 2.31 ベータ版

このバージョンでは，インストラクション・メモリ・キャッシュを持つμPD77116 に対応しています。このために次の機能を強化しました。

(1) 機能の強化

●主要なチップ記述ファイルにはμ PD77116 の記述が含まれます。μ PD77116 に対するモデル・ファイル 'UPD77116.model'を利用できます。Windows 3.xx 環境では，さらにμPD77116 の記述を含むモデル定義ファイル（UPD77016.INI）を利用できます。

●キャッシュ可能インストラクション・メモリのセグメントを含むようにしたプリロード・モジュール・セグメント・クラスを利用できます。プリロード・モジュール・セグメント・クラスのメンバであるセグメントを割り当てるために 3 つのモードを利用できます。

●リンカのワーニング・メッセージに次の 2 つを追加しました。

セグメントを指定以外のバンクに配置すると，"Warning L022: Segment '<mod:seg>' was allocated in bank '<banknumber>'."が表示されます。

モジュール間のジャンプや呼び出しが検出されると，"Warning L023: Jump/Call in segment '<mod 1:seg1>', address <addr> to segment '< mod 2:seg2>', <addr>"が表示されます。

この機能は Linker Settings ダイアログで無効にできます。

●マップ・ファイルの出力項目にプロジェクトで定義したプリロード・モジュール・クラスの記述を追加しました。

F.14　バージョン 2.30 および 2.30a

このバージョンはモデル・エディタで配布されます。モデル・エディタはユーザがモデルを作成したり変更するときに役立ち，モデル定義シンタックスを覚えたりモデル・ファイルを編集する必要がありません。モデル・エディタは Windows 95 と Windows NT4.0 用に設計されており，どちらかのオペレーティング・システム環境で WB77016 をインストールした場合のみ利用できます。

Windows 3.1x や Windows 32S の環境では，これまでのモデル・ファイルによる処理を利用できます。モデル・エディタについての詳細は，マニュアルの該当する部分を参照してください。

(1) 機能の強化

●ブート・クラス，RAM 初期化クラス，オーバレイ・クラスに対応する新しいセグメント・クラスの概念を具体化しました。μPD7701x ファミリ　ユーザーズ・マニュアル　アーキテクチャ編，またはμPD77111 ファミリ　ユーザーズ・マニュアル　アーキテクチャ編　で説明したブート・モードを利用できます。

●マジック番号の処理，相対プロジェクト・パス，プロジェクト・ウインドウの複数選択に関してプロジェクト管理を改善しました。

●「マジック番号を無視する」機能により別のプロセッサ・モデルでアセンブルしたオブジェクト・ファイルをリンクできます。

(2) 確認されているバグ

モデル・エディタのシェルの拡張機能を使うには，セットアップ中に指定するアプリケーションのパスに空白を入れないでください。

F.15　バージョン 2.21（前回のリリース製品）

(1) 機能の強化

●"INID"クラスのメンバとしてのデータ RAM セグメントに対するディフォールトのオプションを"data"としました。

●InterTools77016 C コンパイラと互換性を持たせるために，"INID"クラスのメンバにすることができる INI クラスの RAM セグメントの名前は"in_x_idata"，"in_y_idata"，"ex_x_idata"，"ex_y_idata"に変更します。INID クラスによる INI クラスの ROM セグメントの名前は"I_ROM"のままです。

(2) 2.21 以降のバグ修正

●"OVL"クラスのセグメントのオーバレイ割り当ては，セグメントを直接的に配置することで正しく行われます。

●アセンブラが C ディバグ命令".func"のフラグ（static または dynamic）を認識します。

●アセンブラの命令"DW"と"DWL"は，再配置された複数の引数で正しく機能します。

●Make メニューのコマンド"Build ALL"と"Assemble"で処理を開始した後に，アセンブラ・ソース・ファイルの存在を確認します。

(3) 注意

●平行して命令の読み取りや保存を行う場合は，必ず次の順序で行ってください。

1.. 同時に記述された操作（三項式，二項式，単項式）
2. X メモリの読み取りと保存
3. Y メモリの読み取りと保存

●アセンブラ・ソース・ファイル内の式でメモリのタイプを確認することはできません。式の値を評価するときには，1 つの式の引数を定義したメモリのタイプ（I，X，Y）は考慮されません。

F.16　バージョン 2.21 最終ベータ版

(1) 機能の強化

●"INI"クラスのメンバとしてのデータ RAM セグメントに対するディフォールトのオプションを"data"としました。

●セグメント・クラスの識別子は 0 から 255 の間で設定します。このうち識別子 0 は"INID"クラスのメンバのみに使えます。

保守／廃止

（2）2.2 以降のバグ修正

●"I_ROM"（"INID"クラスのメンバ）と指定されたセグメントの属性は，セルフ・ブートに設定されていても編集することができます。

●データ RAM の初期化は 4 ワード以上のゼロ・ブロックに対しても正しく行われます。

F.17　バージョン 2.2

このバージョンの主な強化内容は，米国 TASKING 社の InterTools77016 C コンパイラに対応するために C ソース・レベルのディバグ処理を実現したことです。ユーザ・インタフェースの C 言語の強化についての詳細は，ヘルプの"System"を参照してください。

（1）機能の強化

●WB77016 は InterTools77016 C コンパイラで生成したファイルを認識し，C ソース・ディバグ情報を XCOFF 出力ファイルに変換します。この機能は Assembler Settings ダイアログの"Include debug information"ボックスをオンにすると利用できます。

●セグメント"BOOT"と"BOOT_HEADER"の一部の属性を Edit segments ダイアログで変更できます（"BOOT"による開始アドレス，内部または外部メモリ）。このようにして，これらのセグメントを割り当てるメモリのタイプ（内部または外部）を指定することができます。この機能は内部と外部の両方のデータ ROM があるシステムで役立ちます。

●ディフォールトのメモリで割り当てを間違えると，リロケータブル・インストラクション・セグメントが内部メモリと外部メモリとの間で自動的に移動します。移動した場合は，次のような新しいワーニングが表示されます。

"Warning L020: Internal segment <name> (module '<name>') allocated in external memory"

または，

"Warning L021: External segment <name> (module '<name>') allocated in internal memory".

●make ユーティリティはどんな拡張子のソース・ファイルも認識できます（初期のバージョンでは拡張子".ASM"しか認識されませんでした）。

●テキストと一部のエラー・メッセージの構文を変更しました。どのメッセージにも識別番号を付けました

（2）2.13 以降のバグ修正

●リンカの拡張子".MSK"の HEX ファイル生成部にバグがあります。

●DEL キーを押すと意味不明の文字が表示されます。

●モデル定義ファイルのメモリ領域の重複が見つかりません。

●不正な条件付きレジスタ転送（if (r2 > 0) r3L = LC）が原因で，見つからないことを示すエラー・メッセージが表示されます。

●ネストされたループの中でエラー・メッセージ"No valid use of data pointer register"が表示されます。

保守／廃止

F.18　バージョン 2.13

(1) 2.12 以降のバグ修正

●Directories ダイアログでハイフンと「..¥」が使えませんでした。

●マスク ROM の HEX ファイルがμPD77015 およびμPD77018 プロセッサに対応します。

F.19　バージョン 2.12

(1) 機能の強化

UPD77016.INI ファイルがμPD77015 とμPD77018 モデルによって拡張されました。

(2) 2.11 以降のバグ修正

●読み取り専用ファイルをエディタで開くことができます。

●マスク ROM の HEX ファイルに内部 ROM のすべての命令があります。

●エラー・メッセージ"No valid use of data pointer register"は，ある特定のループの組み合わせでは表示されませんでした。

F.20　バージョン 2.11

(1) 機能の強化

●モデル・ダイアログを HSM77016 のモデル・ダイアログの形式に変更しました。

●すべてのダイアログを HSM77016 と ID77016 の 3 次元表示に変更しました。

F.21　バージョン 2.1

(1) 機能の強化

●絶対セグメントの開始アドレスを，定数レーベルの後方参照を含む式でも指定できるようにしました。これにより，連続するメモリ領域を占める静的なセグメント系統（ソース・ファイルで定義）を指定できます。

●アセンブラ・ソースやリンカの Segment Attributes ダイアログで，0-0xFFFF の範囲で任意の配列の境界を指定できます。指定した境界が 0 の場合，セグメントはそのサイズよりも大きい次の 2 の累乗（自然境界）に整列します。

●μPD77017 用のマスク ROM オプションを利用できます。

●アセンブラとリンカの処理速度が速くなりました。

●ステータス・ウインドウで文字列"linking"を点滅させて，リンク処理中であることを示します。

●Boot Options ダイアログの待機サイクルの入力を改善しました。

(2) 2.04 以降のバグ修正

●HEX コンバータが.HXI ファイルの 2 番目の拡張アドレス・レコードを出力しませんでした。

F.22　バージョン 2.04

(1) 機能の強化

●絶対セグメントの開始アドレスを，定数ラベルの後方参照を含む式でも指定できるようにしました。これにより，連続するメモリ領域を占める静的なセグメント系統（ソース・ファイルで定義）を指定できます。

●アセンブラ・ソースやリンカの Segment Attributes ダイアログで， 0-0xFFFF の範囲で任意の配列の境界を指定できます。指定した境界が 0 の場合，セグメントはそのサイズよりも大きい次の 2 の累乗（自然境界）に整列します。

●μPD77017 用のマスク ROM オプションを利用できます。

●アセンブラとリンカの処理速度が速くなりました。

●ステータス・ウインドウで文字列"linking"を点滅させて，リンク処理中であることを示します。

●Boot Options ダイアログの待機サイクルの入力を改善しました。

(2) さらに改善された機能

●内部インストラクション RAM と外部インストラクション RAM に対する別々の HEX ファイルを生成できます。

●待機サイクルに個別の番号を付けて，別々のメモリ領域に指定できます。

●リンカの"Abort"ボタンをクリックする機会が増えました。

●プロジェクトの中のロード・モジュール・ファイル（.LNK）を単独で指定して，対応する HEX ファイルを生成できます。

(3) 2.03 以降のバグ修正

●式の中で"EQU"命令と合わせて使われている EXTRN シンボルに対してエラー・メッセージが表示されませんでした。

●セグメントの開始アドレスが前方参照シンボルを含む式で定義されたときに，エラー・メッセージが表示されませんでした。

●C エラー・メッセージにソース・ファイルのパスが含まれていたときに，"Jump to"が機能しませんでした。

●Options ダイアログで存在しないディレクトリを使うと，限りなくループが実行されました。

●アセンブラ・ソース・ファイルに".ASM"以外の拡張子を付けた場合に，エラー・メッセージが表示されませんでした。

F.23　バージョン 2.03

(1) 機能の強化

●プロジェクト・ファイル・ボックスを閉じると，プロジェクト自体も閉じます。

●シンボルを外部メモリとして複数回宣言すると，エラーでなくワーニングが表示されます。

●UPD77016.INI ファイル内のμPD77017 モデルを実際のμPD77017 の仕様と一致するように変更しました。

●UPD77016.INI ファイル内の小さいモデルも改善しました。

●データ・セグメントの命令を循環式バッファリングに対応できるように改善しました。ALIGN を指定すると，これに対応するセグメントがそのワード数よりも大きい次の 2 の累乗の境界に整列して割り当てられます。

segmentname XRAMSEG [AT expression] | [[INTERNAL | EXTERNAL] [ALIGN]]
segmentname YRAMSEG [AT expression] | [[INTERNAL | EXTERNAL] [ALIGN]]
segmentname XROMSEG [AT expression] | [[INTERNAL | EXTERNAL] [ALIGN]]
segmentname YROMSEG [AT expression] | [[INTERNAL | EXTERNAL] [ALIGN]]

(2) 2.02 以降のバグ修正

●マクロのエラーが，実際にエラーが発生している行の 1 行後で表示されました。

●一部のエラーが誤ってフェータル・エラーとして処理されたため，それ以降のエラーの処理が中止されました。

保守／廃止

F.24　バージョン 2.02

(1) 機能の強化

●ライブラリアンのリスト・ボックスを複数のエントリが選択できるように改善しました。.

●ライブラリアンの形式を改善して，NEC ライブラリを認識するようにしました。

注意　ご使用のライブラリは新しいライブラリアン V 2.00 で再構築してください。

F.25　バージョン 2.01

(1) 機能の強化

μPD77017 に対応するために，STOP 命令を追加しました。

(2) 2.00 以降のバグ修正

●Edit Segments ダイアログで誤った外部割り当てが行われることがあります。

●リロケータブル Y データをリンクした後にブート・データをリンクします。

●誤った未解決の外部リンクが，ライブラリアンでそれらの順序に従ってリンクを行っているときに発生します。

●ポップ・アップ・ヘルプが指示どおりに表示されません。

F.26　バージョン 2.00

(1) 機能の強化

リンカが次に示す機能によって強化されました。

a. Combined segments

このオプションを選択すると，同じ名前と属性を持つリロケータブル・セグメントに定義されたすべてのデータとシンボルが，1 つの連続するセグメントにコピーされます。この新しい共通のセグメントはマップ・ファイルに"COMMON"と表示されます。

b. Code analysis for C-compiler

このオプションを選択すると，対応する C ソースのネストされた C 関数の呼び出し数が 15 を越えたときにワーニングが表示されます。

c. Static stack overlay

リンカをこのモードで操作すると，C 関数の呼び出しが分析されます。自動変数を持たない関数の静的スタック・セグメントや同時に存在するスタック・フレーム（ネストされた呼び出し）が，可能であれば重複して割り当てられます。

d. Alignment for circular buffer

リンカがセグメントのワード数よりも大きい次の 2 の累乗の境界に指定したセグメントを割り当てます。この配列はセグメントを循環式バッファ（アドレス指定モード"modulo index addition"で利用できる）として使うときに役立ちます。

e. Boot-info in HEX file

新しい Boot Options ダイアログを使って，次に示す補助的なブート情報を生成できます。

リンカで IRAM 領域を外部 X データ・メモリか外部 Y データ・メモリにコピーできます。

内部ブートを行うために，μ PD77016 はすべての命令を外部 X データ・メモリか外部 Y データ・メモリ（ROM）からインストラクション RAM へコピーできます。

ホスト・ブートを行うために，インストラクション RAM のブートに関する内容のコピーを持つ新しい HEX ファイル（*.HBI）を生成できます。

f. Overlapping segments

データ・セグメントを重複して割り当てることができます。そのために，セグメントを Edit Segments ダイアログで専用の"OVL"クラスのメンバとして指定します。

（2）さらに改善された機能

●WB77016 は InterTools77016 C コンパイラで生成したエラー・ファイルを読み取り，翻訳することができます。

●アセンブラは InterTools77016 C コンパイラからのシンボル命令を認識します。

●NEC の RA77016 との互換性上の理由で削除した内部命令および外部命令を，再び次のように実装しました。

segmentname IMSEG [AT expression | INTERNAL | EXTERNAL]

・アセンブラのエラー検出とマクロ処理部分を改善しました。

（3）バグ修正

●固定小数点または整数の式の評価。

●"Debug Info"オプションに関するバグを修正しました。以前のバージョンでこのオプションを選択すると，リンカが誤った場所を生成していました。

●以前のバージョンのリンカは.LIB ファイルのシンボルの処理に問題が発生することがありました。

●XCOFF のチップ・タイプ（77016,77017）が不適当でした。

●プリント・ファイルのページ送り。

●ユーザが中断すると不適当なエラー・メッセージが表示されました。

［メ　　モ］

保守／廃止

付録G　索　　引

G.1　アルファベット順

保守／廃止

G.2 五十音順

保守／廃止

お問い合わせ先

【技術的なお問い合わせ先】

ＮＥＣ半導体テクニカルホットライン
（電話：午前 9:00～12:00，午後 1:00～5:00）

電　話　：044−435−9494
FAX　：044−435−9608
E-mail　：s-info@saed.tmg.nec.co.jp

【営業関係お問い合わせ先】

第一販売事業部
東　京　(03)3798−6106, 6107, 6108
名古屋　(052)222−2375
大　阪　(06)6945−3178, 3200, 3208, 3212
仙　台　(022)267−8740
郡　山　(024)923−5591
千　葉　(043)238−8116

第二販売事業部
東　京　(03)3798−6110, 6111, 6112
立　川　(042)526−5981, 6167
松　本　(0263)35−1662
静　岡　(054)254−4794
金　沢　(076)232−7303
松　山　(089)945−4149

第三販売事業部
東　京　(03)3798−6151, 6155, 6586, 1622, 1623, 6156
水　戸　(029)226−1702
広　島　(082)242−5504
高　崎　(027)326−1303
鳥　取　(0857)27−5313
太　田　(0276)46−4014
名古屋　(052)222−2170, 2190
福　岡　(092)261−2806

【資料の請求先】

上記営業関係お問い合わせ先またはNEC特約店へお申しつけください。

【インターネット電子デバイス・ニュース】

NECエレクトロンデバイ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.ic.nec.co.jp/

保守／廃止

アンケート記入のお願い

お手数ですが，このドキュメントに対するご意見をお寄せください。今後のドキュメント作成の参考にさせていただきます。

［ドキュメント名］ WB77016 ユーザーズ・マニュアル 操作編（Windows ベース）
（U11506JJ3V0UMJ1（第 3 版））

［お名前など］（さしつかえのない範囲で）

御社名（学校名，その他） （ ）
ご住所 （ ）
お電話番号 （ ）
お仕事の内容 （ ）
お名前 （ ）

1．ご評価（各欄に○をご記入ください）

項　　目	大変良い	良　い	普　通	悪　い	大変悪い
全体の構成					
説明内容					
用語解説					
調べやすさ					
デザイン，字の大きさなど					
その他（ ）					
（ ）					

2．わかりやすい所（第　章，第　章，第　章，第　章，その他 ）
理由［ ］

3．わかりにくい所（第　章，第　章，第　章，第　章，その他 ）
理由［ ］

4．ご意見，ご要望

5．このドキュメントをお届けしたのは
NEC 販売員，特約店販売員， その他（ ）

ご協力ありがとうございました。
下記あてに FAX で送信いただくか，最寄りの販売員にコピーをお渡しください。

日本電気(株) NEC エレクトロンデバイス
半導体テクニカルホットライン
FAX：（044）435-9608

2000.6

キリトリ