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。

ユーザーズ・マニュアル

RENESAS

保守／廃止

RA78K/I

アセンブラ・パッケージ

操作編

資料番号　EEU-883（第1版）
発行年月　September 1992 P

ユーザーズ・マニュアル

NEC
電子デバイス

保守／廃止

RA78K/ I

アセンブラ・パッケージ

操作編

保守／廃止

V30™ は，日本電気株式会社の商標です。

IBM PC™，IBM PC/AT™，IBM PC/XT™，PC DOS™ は，米国 IBM 社の商標です。

MS-DOS™ は，米国マイクロソフト社の商標です。

8086™，80286™，80386™は，米国インテル社の商標です。

保守／廃止

本製品は外国為替および外国貿易管理法の規定により戦略物資等（または役務）に該当しますので，日本国外に輸出する場合には，同法に基づき日本国政府の輸出許可が必要です。

保守／廃止

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

このマニュアルは，RA78K/Ⅰ　アセンブラ・パッケージ（以下，RA78K/Ⅰ）に含まれる各プログラムの機能および操作方法を正しく理解していただくことを目的としています。

このマニュアルではRA78K/Ⅰの疑似命令やソース・プログラムの様式など，言語に関する解説はしていません。したがって，このマニュアルをお読みになる前に“RA78K/Ⅰ　アセンブラ・パッケージ　ユーザーズ・マニュアル　言語編”（以下，言語編）をお読みください。

【対 象 者】

このマニュアルは，開発対象となるマイクロコンピュータの機能およびインストラクションについて理解しているユーザを対象としています。

【構　　成】

このマニュアルは次のように構成されています。

○第1章　概　　説

：マイクロコンピュータの開発におけるRA78K/Ⅰの役割など，RA78K/Ⅰ全体の機能概要について説明します。

○第2章　製品概要

：RA78K/Ⅰが提供するプログラムのファイル名，動作環境などについて説明します。

○第3章　RA78K/Ⅰの実行

：サンプル・プログラムを使用して，開発手順について説明します。

○第4章　アセンブラ

○第5章　リンカ

○第6章　オブジェクト・コンバータ

○第7章　ライブラリアン

○第8章　リスト・コンバータ

：各プログラムの機能，操作方法について詳細に説明します。

これらの章は，実際にRA78K/Ⅰの各プログラムを操作するうえで重要です。

○第9章　プログラムの出力リスト

：各プログラムが出力する各種リストのフォーマットについて説明します。

○第10章　RA78K/Ⅰの活用法

：RA78K/Ⅰをうまく使うための方法を紹介します。

○第11章　エラー・メッセージ一覧

：各プログラムが出力するエラー・メッセージについて説明します。

○付　　録

：プログラムのオプション一覧表，サンプル・プログラム・リスト，使用上の注意一覧などを紹介します。

なお，このマニュアルではインストラクションについての詳細説明はしておりません。

インストラクションの詳細については，開発対象となるマイクロコンピュータのユーザーズ・マニュアルをお読みください。

【読 み 方】

アセンブラを初めて使われる方は，“**第１章　概　　説**”からお読みください。アセンブラに関する一般的知識のある方は読み飛ばされても結構です。

実際にRA78K/Ⅰを使用する場合は，“**第３章　RA78K/Ⅰの実行**”をお読みください。

各プログラムの操作に慣れてからは，付録の一覧表をご活用ください。

【凡　　例】

このマニュアルの中で共通に使用される記号などの意味を示します。

…	：同一の形式を繰り返します。
[ ]	：[ ] の中は省略可能です。
「 」	：「 」で囲まれた文字そのものを表します。
‘ ’	：‘ ’で囲まれた文字そのものを表します。
〈 〉	：〈 〉で囲まれた文字そのもの（おもにタイトル）を表します。
“ ”	：このマニュアルでの参照箇所（章，節，項，図，表）を表します。
＿＿＿	：重要箇所，また，使用例での下線は入力文字列を表します。
□	：1個の空白を表します。
△	：1個以上の空白またはタブを表します。
▲	：0個以上の空白またはタブ（省略可能の意）を表します。
⋮	：プログラム記述の省略形を表します。
／	：文字の区切りを表します。
～	：連続性を表します。
↵	：リターン・キーの入力を表します。

注	：本文中に付けた**注**の説明
注意	：特に気をつけて読んでいただきたい内容
備考	：本文の補足説明

【対象デバイス】

RA78K/Ⅰで，次のマイクロコンピュータのソフトウエア開発が可能です。

- μPD78134，78134A，78136，78138，78P138
- μPD78146，78148，78P148

【関連資料】

	資　料　名	資料番号
デバイス	μPD78134　ユーザーズ・マニュアル	IEU-668
	μPD78138　ユーザーズ・マニュアル	IEU-740
	μPD78148　ユーザーズ・マニュアル	IEU-764
開発ツール	RA78K/Ⅰ　アセンブラ・パッケージ ユーザーズ・マニュアル　操作編	このマニュアル
	RA78K/Ⅰ　アセンブラ・パッケージ ユーザーズ・マニュアル　言語編	EEU-892
	RA78K/Ⅰ　構造化アセンブラ・プリプロセッサ ユーザーズ・マニュアル	作成中
	IE-78130-R　ユーザーズ・マニュアル ハードウエア編	EEU-647
	IE-78130-R　ユーザーズ・マニュアル ソフトウエア編	EEU-648
	IE-78140-R　ユーザーズ・マニュアル ハードウエア編	EEU-771
	IE-78140-R　ユーザーズ・マニュアル ソフトウエア編	EEU-767

保守／廃止

保守／廃止

第１章 概説 1

第２章 製品概要 2

第３章 RA78K/Ｉの実行 3

第４章 アセンブラ 4

第５章 リンカ 5

第６章 オブジェクト・コンバータ 6

第７章 ライブラリアン 7

第８章 リスト・コンバータ 8

第９章 プログラムの出力リスト 9

第10章 RA78K/Ｉの活用法 10

第11章 エラー・メッセージ 11

付録Ａ サンプル・プログラム A

付録Ｂ 使用上の注意一覧 B

付録Ｃ オプション一覧 C

付録Ｄ サブコマンド一覧 D

保守／廃止

保守／廃止

目　　次

第１章　概　　説 … 1

1.1 アセンブラの概要 … 2
1.1.1 アセンブラとは … 3
1.1.2 リロケータブル・アセンブラとは … 7
1.2 RA78K/Ⅰの機能概要 … 9
1.2.1 エディタによるソース・モジュール・ファイルの作成 … 10
1.2.2 構造化アセンブラ・プリプロセッサ … 11
1.2.3 アセンブラ … 12
1.2.4 リ ン カ … 13
1.2.5 オブジェクト・コンバータ … 14
1.2.6 ライブラリアン … 15
1.2.7 リスト・コンバータ … 16
1.3 メモリ・マップ … 17
1.4 プログラム開発をはじめる前に … 19
1.4.1 リンカに入力可能なファイル数 … 19
1.4.2 シンボル数の制限 … 19
1.4.3 RA78K/Ⅰの最大性能 … 19
1.5 RA78K/Ⅰの特徴 … 20

第２章　製品概要 … 21

2.1 プログラム・ファイル … 22
2.2 ホスト・マシン対象機種と対象OS … 23
2.2.1 PC-9800シリーズ … 23
2.2.2 IBM PCシリーズ … 24

第３章　RA78K/Ⅰの実行 … 25

3.1 RA78K/Ⅰ実行の前に … 26
3.1.1 ディスク内容の確認 … 26
3.1.2 サンプル・プログラム … 26
3.2 RA78K/Ⅰの実行手順 … 29
3.3 RA78K/Ⅰの実行手順のまとめ … 33

第４章　アセンブラ … 37

4.1 アセンブラの入出力ファイル … 38
4.2 アセンブラの機能 … 40
4.3 アセンブラの起動方法 … 41
4.3.1 アセンブラの起動 … 41

4.3.2 実行開始，終了メッセージ … 43
4.4 アセンブラ・オプション … 45
4.4.1 アセンブラ・オプションの種類 … 45
4.4.2 アセンブラ・オプションの優先度 … 46
4.4.3 アセンブラ・オプションの説明 … 48

第5章 リ ン カ … 85

5.1 リンカの入出力ファイル … 86
5.2 リンカの機能 … 87
5.3 メモリ空間とメモリ領域 … 88
5.4 リンク・ディレクティブ … 89
5.4.1 ディレクティブ・ファイル … 90
5.4.2 メモリ・ディレクティブ … 92
5.4.3 セグメント配置ディレクティブ … 95
5.5 リンカの起動方法 … 98
5.5.1 リンカの起動 … 98
5.5.2 実行開始，終了メッセージ … 100
5.6 リンカ・オプション … 102
5.6.1 リンカ・オプションの種類 … 102
5.6.2 リンカ・オプションの優先度 … 103
5.6.3 リンカ・オプションの説明 … 105

第6章 オブジェクト・コンバータ … 137

6.1 オブジェクト・コンバータの入出力ファイル … 138
6.2 オブジェクト・コンバータの機能 … 140
6.3 オブジェクト・コンバータの起動方法 … 143
6.3.1 オブジェクト・コンバータの起動 … 143
6.3.2 実行開始，終了メッセージ … 145
6.4 オブジェクト・コンバータ・オプション … 147
6.4.1 オブジェクト・コンバータ・オプションの種類 … 147
6.4.2 オブジェクト・コンバータ・オプションの説明 … 148

第7章 ライブラリアン … 161

7.1 ライブラリアンの入出力ファイル … 162
7.2 ライブラリアンの機能 … 163
7.3 ライブラリアンの起動方法 … 165
7.3.1 ライブラリアンの起動 … 165
7.3.2 実行開始，終了メッセージ … 168
7.4 ライブラリアン・オプション … 170
7.4.1 ライブラリアン・オプションの種類 … 170
7.4.2 ライブラリアン・オプションの説明 … 171
7.5 サブコマンド … 177
7.5.1 サブコマンドの種類 … 177
7.5.2 サブコマンドの説明 … 177

第8章　リスト・コンバータ　…　191

8.1　リスト・コンバータの入出力ファイル　…　192
8.2　リスト・コンバータの機能　…　194
8.3　リスト・コンバータの起動方法　…　197
8.3.1　リスト・コンバータの起動　…　197
8.3.2　実行開始，終了メッセージ　…　199
8.4　リスト・コンバータ・オプション　…　201
8.4.1　リスト・コンバータ・オプションの種類　…　201
8.4.2　リスト・コンバータ・オプションの説明　…　202

第9章　プログラムの出力リスト　…　211

9.1　アセンブラの出力リスト　…　212
9.1.1　アセンブル・リスト・ファイルのヘッダ　…　212
9.1.2　アセンブル・リスト　…　213
9.1.3　シンボル・リスト　…　215
9.1.4　クロスレファレンス・リスト　…　216
9.1.5　エラー・リスト　…　217
9.2　リンカの出力リスト　…　218
9.2.1　リンク・リスト・ファイルのヘッダ　…　218
9.2.2　マップ・リスト　…　220
9.2.3　パブリック・シンボル・リスト　…　221
9.2.4　ローカル・シンボル・リスト　…　222
9.2.5　エラー・リスト　…　223
9.3　オブジェクト・コンバータの出力リスト　…　223
9.3.1　エラー・リスト　…　223
9.4　ライブラリアンの出力リスト　…　224
9.4.1　ライブラリ情報出力リスト　…　224
9.5　リスト・コンバータの出力リスト　…　225
9.5.1　アブソリュート・アセンブル・リスト　…　225
9.5.2　エラー・リスト　…　225

第10章　RA78K/Ⅰの活用法　…　227

10.1　作業の効率化（EXIT ステータス機能）　…　228
10.2　開発環境の整備（環境変数）　…　229
10.3　プログラム実行の中断　…　229
10.4　アセンブル・リストを見やすくする　…　230
10.5　プログラム起動時の手間を省く　…　231
10.5.1　ソース・プログラムに制御命令を記述する　…　231
10.5.2　パラメータ・ファイルやサブコマンド・ファイルを作成する　…　231
10.6　オブジェクト・モジュールのライブラリ化　…　232

保守／廃止

第11章　エラー・メッセージ　…　233

11.1　エラー・メッセージの概要　…　234
11.2　RA78K/I 共通のエラー・メッセージ　…　235
11.3　アセンブラのエラー・メッセージ　…　237
11.4　リンカのエラー・メッセージ　…　247
11.5　オブジェクト・コンバータのエラー・メッセージ　…　253
11.6　ライブラリアンのエラー・メッセージ　…　255
11.7　リスト・コンバータのエラー・メッセージ　…　257

付録A　サンプル・プログラム　…　259

A.1　ソース・リスト　…　260
A.2　実 行 例　…　262
A.3　出力リスト　…　263
A.3.1　アセンブル・リスト　…　263
A.3.2　シンボル・リスト　…　265
A.3.3　クロスレファレンス・リスト　…　265
A.3.4　マップ・リスト　…　267
A.3.5　パブリック・シンボル・リスト　…　268
A.3.6　ローカル・シンボル・リスト　…　268
A.3.7　ライブラリ情報出力リスト　…　268
A.3.8　アブソリュート・アセンブル・リスト　…　269

付録B　使用上の注意一覧　…　271

付録C　オプション一覧　…　273

C.1　アセンブラ・オプション一覧　…　274
C.2　リンカ・オプション一覧　…　276
C.3　オブジェクト・コンバータ・オプション一覧　…　279
C.4　ライブラリアン・オプション一覧　…　280
C.5　リスト・コンバータ・オプション一覧　…　280

付録D　サブコマンド一覧　…　281

保守／廃止

図　の　目　次

図番号	タイトル，ページ
1-1	RA78K/I　アセンブラ・パッケージ　… 2
1-2	アセンブラの流れ　… 3
1-3	マイクロコンピュータ応用製品の開発工程　… 4
1-4	ソフトウエアの開発工程　… 5
1-5	RA78K/I のアセンブル工程　… 6
1-6	アセンブルのやり直し　… 8
1-7	既成モジュールを利用したプログラム作成　… 8
1-8	RA78K/I によるプログラム開発手順　… 9
1-9	ソース・モジュール・ファイルの作成　… 10
1-10	構造化アセンブラ・プリプロセッサの機能　… 11
1-11	アセンブラの機能　… 12
1-12	リンカの機能　… 13
1-13	オブジェクト・コンバータの機能　… 14
1-14	ライブラリアンの機能　… 15
1-15	リスト・コンバータの機能　… 16
1-16	μPD78138 のメモリ・マップ　… 17
3-1	サンプル・プログラムの構造　… 26
3-2	リンク・ディレクティブ　… 30
4-1	アセンブラの入出力ファイル　… 39
5-1	メモリ領域名　… 92
5-2	不正なメモリ領域の定義　… 93
6-1	オブジェクト・コンバータの入出力ファイル　… 139
7-1	ライブラリアンの入出力ファイル　… 162
7-2	ライブラリ・ファイルの作成手順　… 164
8-1	リスト・コンバータの入出力ファイル　… 193

保守／廃止

表　の　目　次

表番号　　タイトル，ページ

2-1　プログラム・ファイル　…　22

2-2　ホスト・マシン対象機種一覧（PC-9800 シリーズ）　…　23

4-1　アセンブラの入出力ファイル　…　38

4-2　アセンブラ・オプション　…　45

4-3　アセンブラ・オプションの優先度　…　46

4-4　対象デバイスとデバイス種別との対応　…　49

4-5　タイトルとして記述可能な文字　…　72

5-1　リンカの入出力ファイル　…　86

5-2　ディレクティブの種類　…　89

5-3　ディフォールトのメモリ領域と再定義可能な範囲　…　93

5-4　メモリ領域名指定とメモリ空間の組み合わせによるセグメントの配置　…　96

5-5　リンカ・オプション　…　102

5-6　リンカ・オプションの優先度　…　103

6-1　オブジェクト・コンバータの入出力ファイル　…　138

6-2　拡張空間に対する出力ファイルのファイル・タイプ　…　140

6-3　オブジェクト・コンバータ・オプション　…　147

7-1　ライブラリアンの入出力ファイル　…　162

7-2　ライブラリアン・オプション　…　170

7-3　サブコマンド　…　177

8-1　リスト・コンバータの入出力ファイル　…　192

8-2　リスト・コンバータ・オプション　…　201

11-1　RA78K/I 共通のエラー・メッセージ　…　235

11-2　アセンブラのエラー・メッセージ　…　237

11-3　リンカのエラー・メッセージ　…　247

11-4　オブジェクト・コンバータのエラー・メッセージ　…　253

11-5　ライブラリアンのエラー・メッセージ　…　255

11-6　リスト・コンバータのエラー・メッセージ　…　257

保守／廃止

保守／廃止

第1章　概　　説

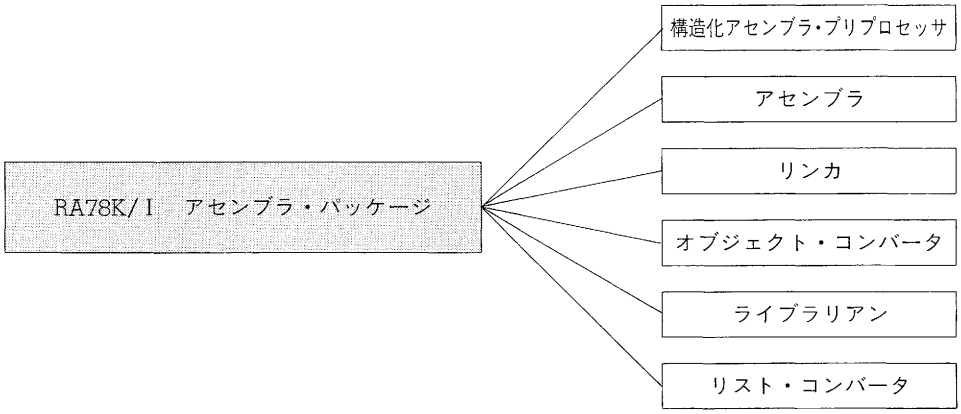
この章では，マイクロコンピュータの開発におけるRA78K/Ⅰの役割など，RA78K/Ⅰ全体の機能概要について説明します。

1.1 アセンブラの概要

RA78K/I　アセンブラ・パッケージは，78K/I シリーズのアセンブリ言語で記述されたソース・プログラムを機械語に変換する一連のプログラムの総称です。

RA78K/I の中には，構造化アセンブラ・プリプロセッサ，アセンブラ，リンカ，オブジェクト・コンバータ，ライブラリアンおよびリスト・コンバータの６つのプログラムがあります。

図１－１　RA78K/I　アセンブラ・パッケージ



1.1.1 アセンブラとは

(1) アセンブリ言語と機械語

アセンブリ言語は，マイクロプロセッサ用の最も基本的なプログラミング言語です。

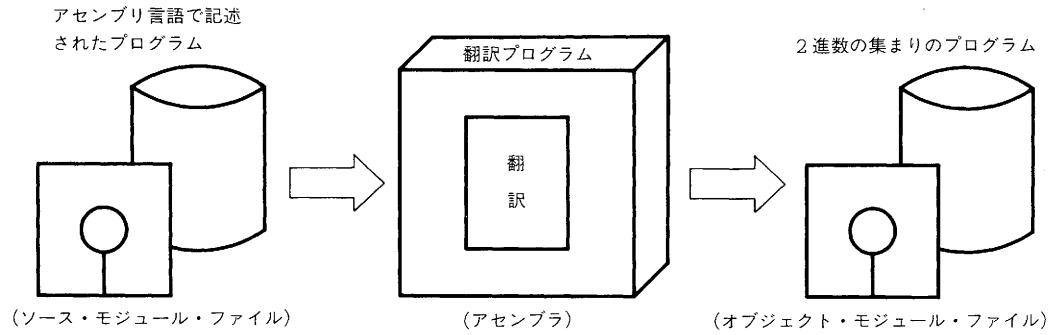
マイクロプロセッサに仕事をさせるためには，プログラムやデータが必要です。これを人間がプログラミングして，マイクロコンピュータのメモリ部に記憶させます。マイクロコンピュータが扱うことのできるプログラムやデータは２進数の集まりで，これを機械語（コンピュータの理解できる言葉）といいます。

機械語すなわち，２進数でプログラムを作るのは，人間にとって覚えにくく，また誤りを起こしやすいものです。

そこで機械語の意味を人間にとって理解しやすい英語の略号で表し，この記号を使ってプログラムを作成する方法があります。この記号によるプログラムの言語体系をアセンブリ言語といいます。

アセンブリ言語で作成したプログラムを，マイクロプロセッサが理解できる２進数の集まりに翻訳するプログラムが必要となります。これをアセンブラと呼びます。

図１-２　アセンブラの流れ

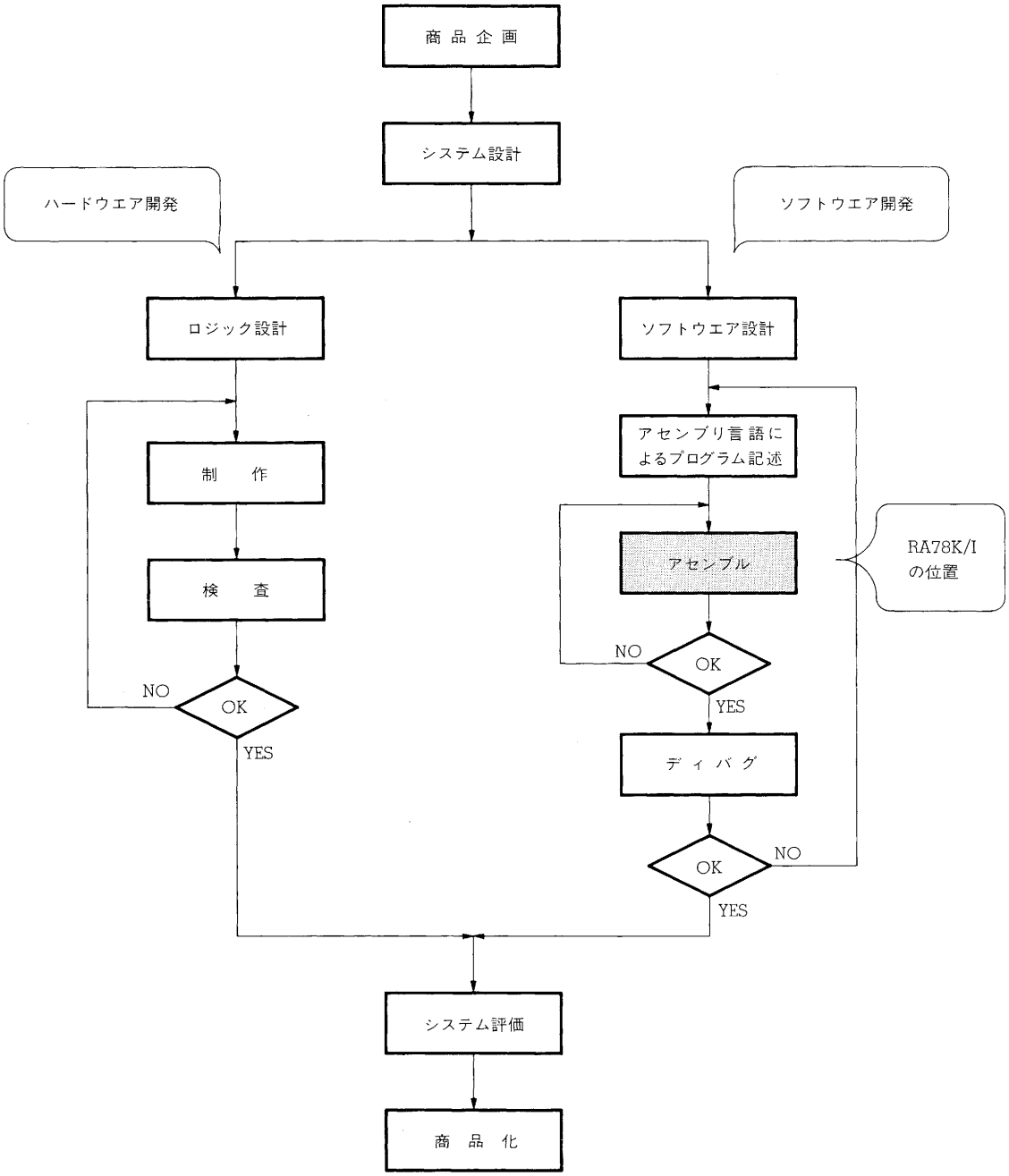


保守／廃止

(2) マイクロコンピュータ応用製品の開発と RA78K/I の役割

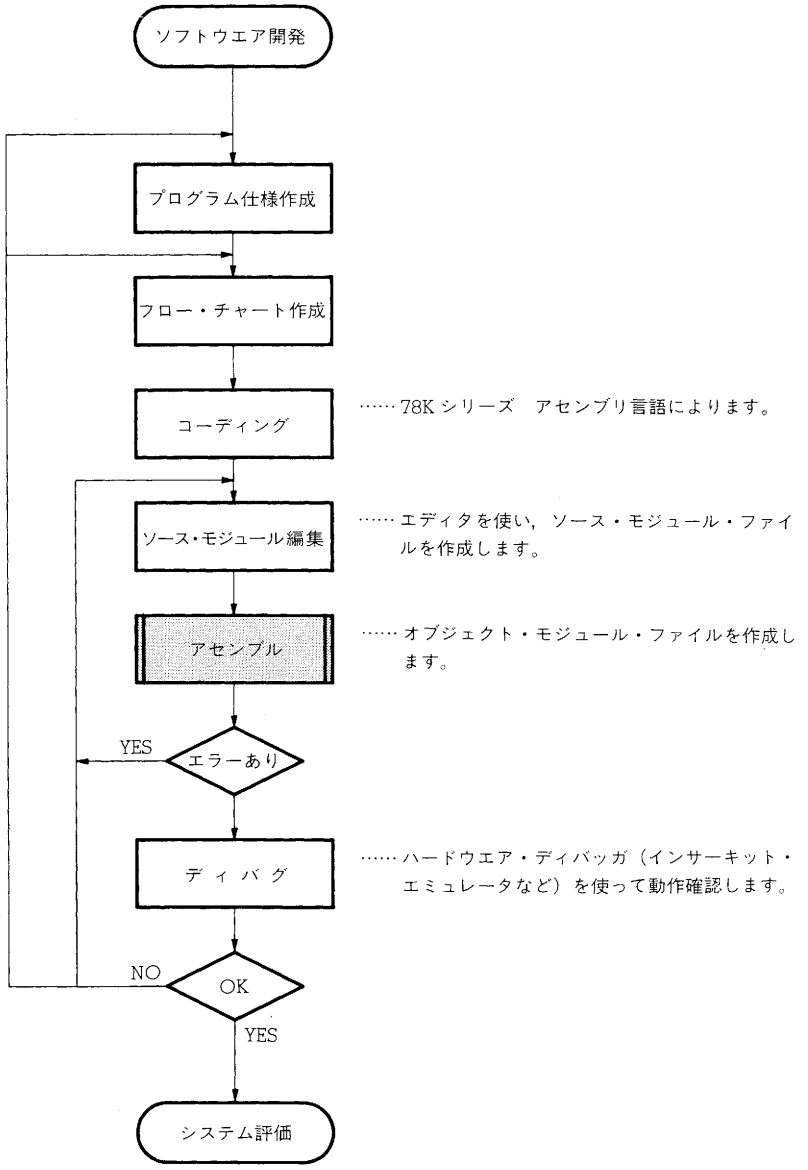
アセンブリ言語でのプログラミングの製品開発における位置付けを“図1-3　マイクロコンピュータ応用製品の開発工程”で説明します。

図1-3　マイクロコンピュータ応用製品の開発工程



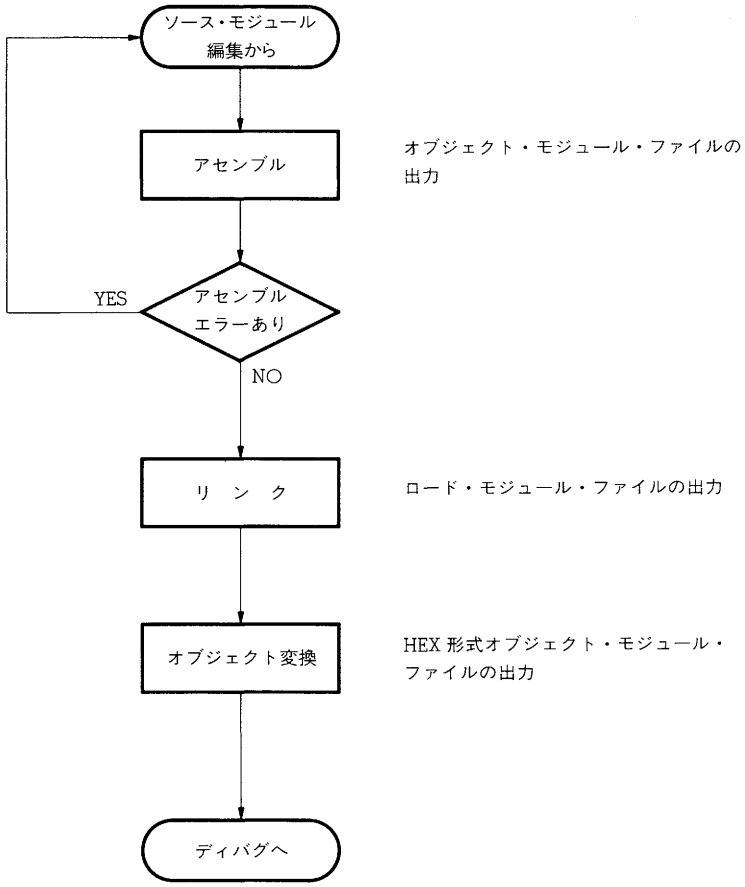
ソフトウエア開発の工程をもう少し詳しく，“図1－4　ソフトウエアの開発工程”で説明します。

図1－4　ソフトウエアの開発工程



さらに，アセンブル工程に，RA78K/Ⅰ をあてはめてみます。

図1-5　RA78K/Ⅰ のアセンブル工程



保守／廃止

1.1.2 リロケータブル・アセンブラとは

アセンブラが変換した機械語は，マイクロコンピュータ内のメモリに書き込まれて使用されます。このとき，変換された機械語がメモリのどこに書き込まれるかが，決定されていなければなりません。したがって，アセンブラの変換する機械語には，「各機械語がメモリ中のどのアドレスに配置されるべきか」という情報が付加されています。

機械語をメモリ・アドレスに配置する方法により，アセンブラは，「アブソリュート・アセンブラ」と「リロケータブル・アセンブラ」とに大別することができます。

- アブソリュート・アセンブラ
  アセンブリ言語から変換した機械語を，絶対（アブソリュート）アドレスに配置します。
- リロケータブル・アセンブラ
  アセンブリ言語から変換した機械語には，一時的なアドレスを与えます。絶対的なアドレスの決定は，リンカと呼ばれるプログラムにより行います。

アブソリュート・アセンブラで1つのプログラムを作成する際には，原則として一度にプログラミングしなければなりませんでした。しかし，大きなプログラムを1つのまとまりとして一度に作成すると，プログラムが複雑になり，また保守する際にもプログラムの解析が大変になります。そこで，1つ1つの機能単位ごとにいくつかのサブプログラム（モジュール）に分割して，プログラム開発を行います。これをモジュラ・プログラミングといいます。

リロケータブル・アセンブラは，モジュラ・プログラミングに適したアセンブラです。

リロケータブル・アセンブラを使用してモジュラ・プログラミングを行うことにより，次の利点があげられます。

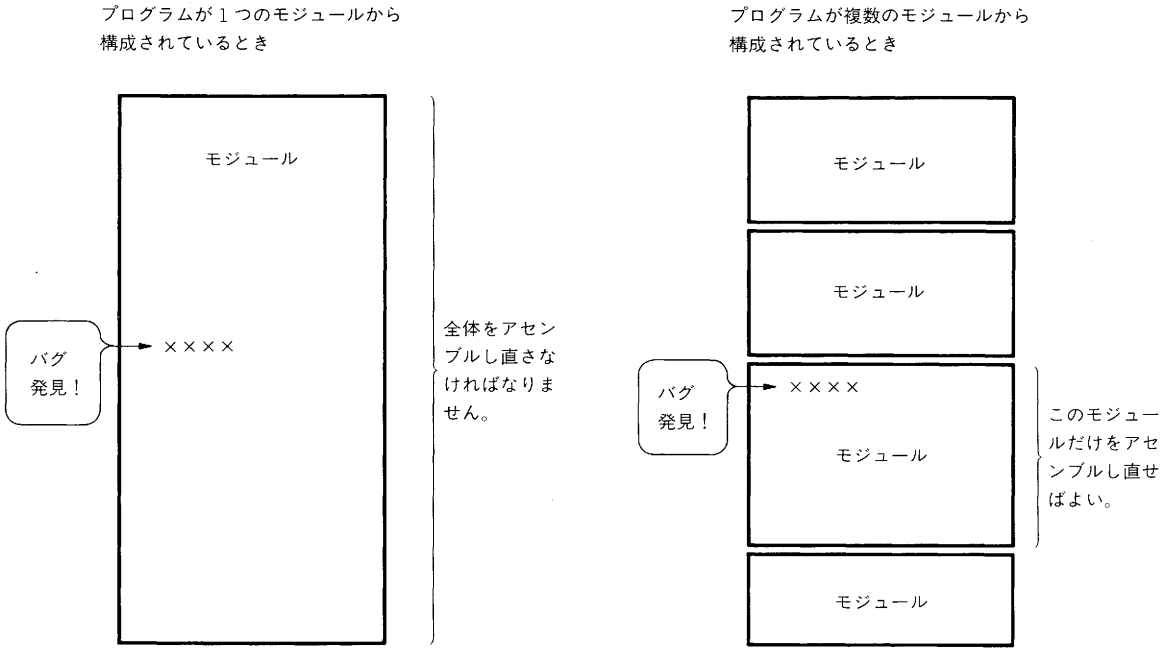
(1) 開発効率が上がる

大きなプログラムを一度にプログラミングすることは困難です。このような場合，プログラムを機能ごとにモジュール分割すれば，複数の人数でプログラムの並行開発ができ，開発効率が上がります。

また，プログラム中にバグが発見された場合，一部の修正のために全プログラムをアセンブルすることなく，修正が必要なモジュールだけアセンブルし直すことができます。

これにより，ディバグ時間を短くすることができます。

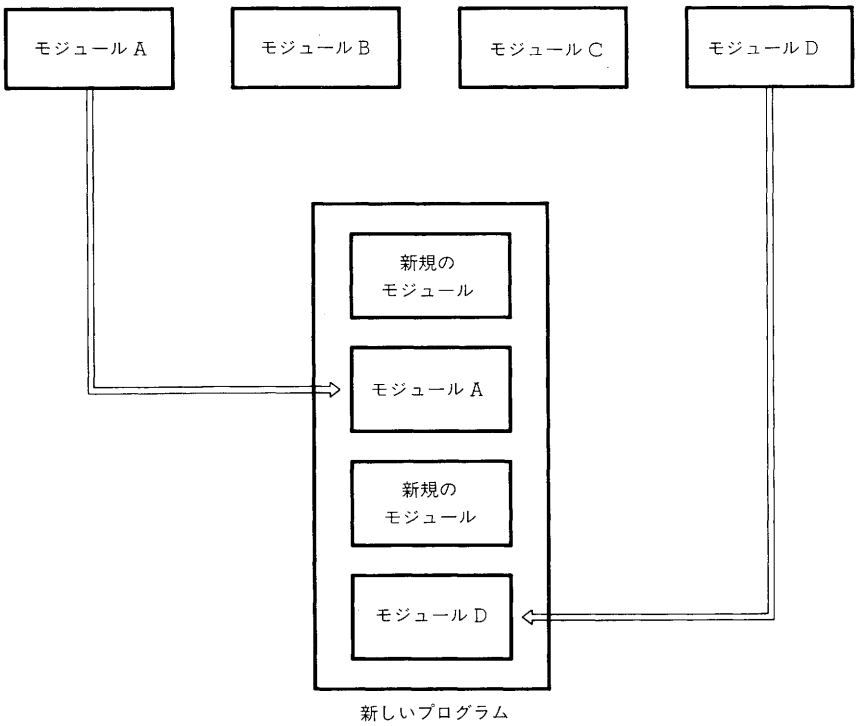
図1－6　アセンブルのやり直し



(2) 資源の活用ができる

以前に作成された信頼性，汎用性の高いモジュールを，他のプログラムの開発に利用することができます。このような汎用性の高いモジュールを蓄積することにより，新規にプログラム開発する部分を少なくすることができます。

図1－7　既成モジュールを利用したプログラム作成



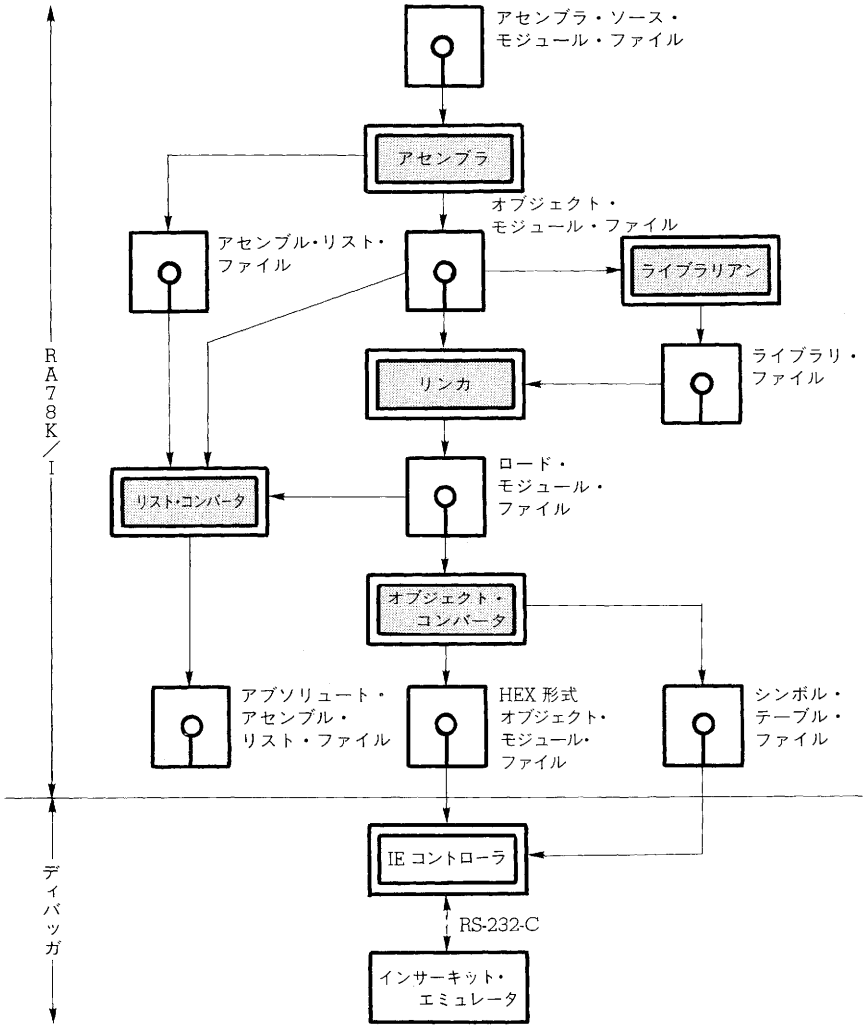
保守／廃止

1.2 RA78K/I の機能概要

RA78K/I における一般的なプログラム開発手順を，“図1-8 RA78K/I によるプログラム開発手順”に示します。プログラムの開発は，基本的にアセンブラ→リンカ→オブジェクト・コンバータを使用して行います。

なお，以降は，アセンブラ，リンカ，オブジェクト・コンバータなどの各プログラムを総称して「RA78K/I」といい，アセンブラ・プログラムのことを「アセンブラ」といいます。

図1-8 RA78K/I によるプログラム開発手順



1.2.1 エディタによるソース・モジュール・ファイルの作成

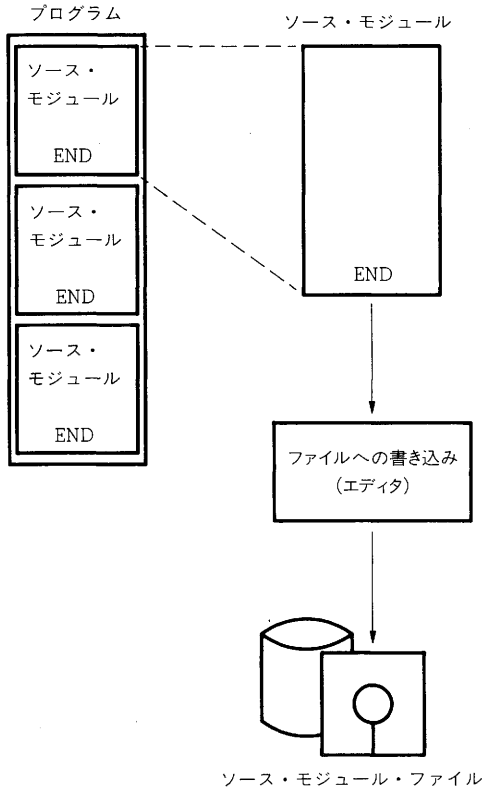
1つのプログラムを，機能的にいくつかのモジュールに分割します。

1つのモジュールは，コーディングの単位となるもので，またアセンブラの入力単位ともなります。アセンブラの入力単位となるモジュールを，ソース・モジュールと呼びます。

各ソース・モジュールのコーディング終了後，エディタを使ってソース・モジュールをファイルに書き込みます。こうしてできたファイルを，ソース・モジュール・ファイルと呼びます。

ソース・モジュール・ファイルは，アセンブラの入力となります。

図1-9 ソース・モジュール・ファイルの作成



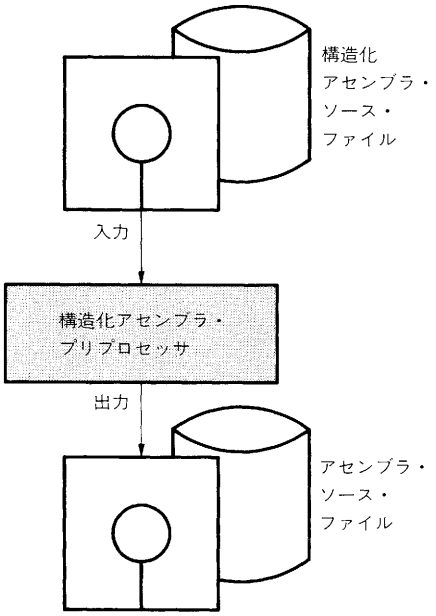
保守／廃止

1.2.2 構造化アセンブラ・プリプロセッサ

構造化アセンブラ・プリプロセッサは，アセンブリ言語で構造化プログラミングを実現するためのプログラムです。構造化アセンブリ言語で書かれたソース・プログラムを入力し，アセンブラのソース・プログラムを出力します。

構造化アセンブラ・プリプロセッサや構造化アセンブリ言語については，別冊の“RA78K/I　構造化アセンブラ・プリプロセッサ　ユーザーズ・マニュアル”をご覧ください。

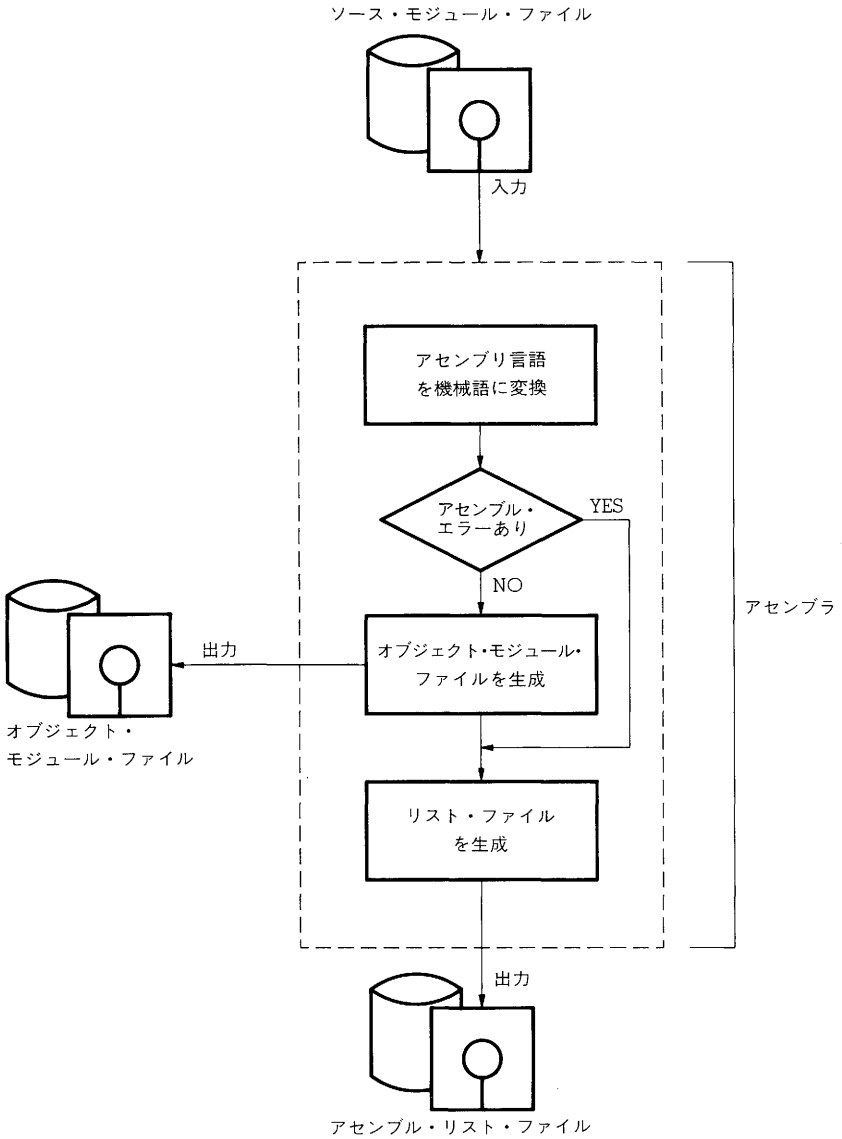
図1-10 構造化アセンブラ・プリプロセッサの機能



1.2.3 アセンブラ

アセンブラは，ソース・モジュール・ファイルを入力し，アセンブリ言語を２進数の集まり（機械語）に変換するプログラムです。ソース・モジュールの中に記述ミスを発見した場合は，アセンブル・エラーを出力します。アセンブル・エラーがない場合には，機械語情報と各機械語がメモリ中のどのアドレスに配置されるべきか，などの配置情報とを含むオブジェクト・モジュール・ファイルを出力します。また，アセンブル時の情報をアセンブル・リスト・ファイルとして出力します。

図１－１１　アセンブラの機能

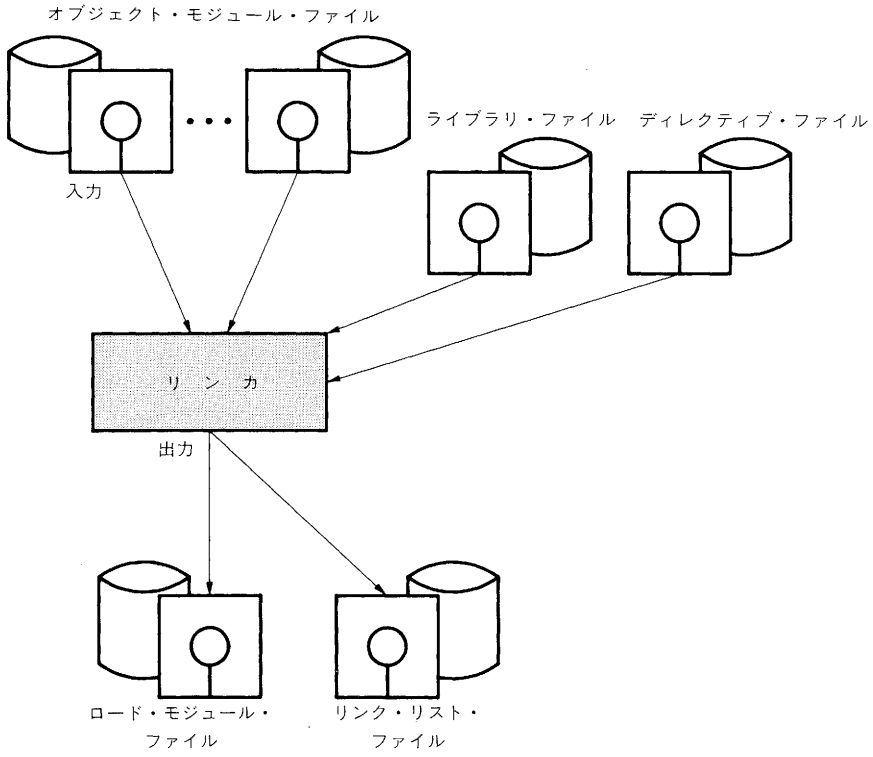


1.2.4 リ ン カ

リンカは,アセンブラの出力したオブジェクト・モジュール・ファイルを複数個入力し,1つのロード・モジュール・ファイルを出力します（オブジェクト・モジュール・ファイルが1つの場合でもリンクしなければなりません)。

この際リンカは，入力されたモジュール中のリロケータブル・セグメントの配置アドレスを決定します。これにより，リロケータブル・シンボルや外部参照シンボルの値を決定し，ロード・モジュール・ファイルに正しい値を埋め込みます。

図1-12 リンカの機能

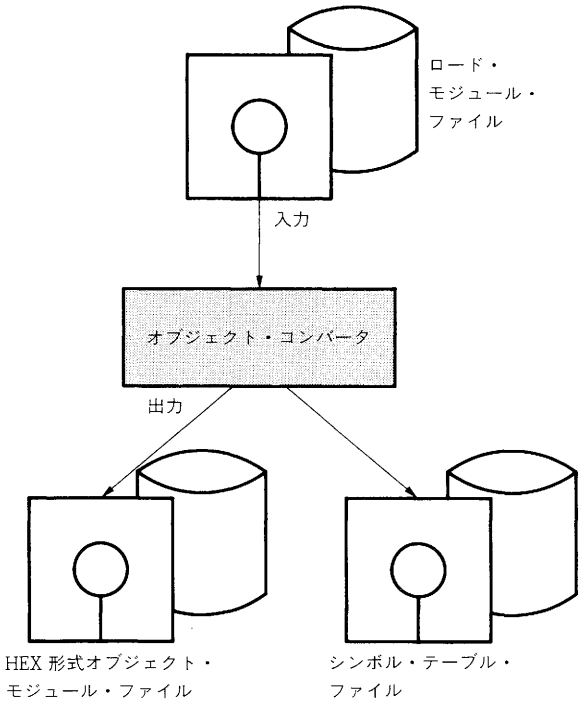


1.2.5 オブジェクト・コンバータ

オブジェクト・コンバータは，リンカの出力したロード・モジュール・ファイルを入力し，ファイル形式を変換します。そして，その結果をHEX形式オブジェクト・モジュール・ファイルとして出力します。

また，シンボリック・ディバグ時に必要となるシンボル情報をシンボル・テーブル・ファイルとして出力します。

図1-13 オブジェクト・コンバータの機能



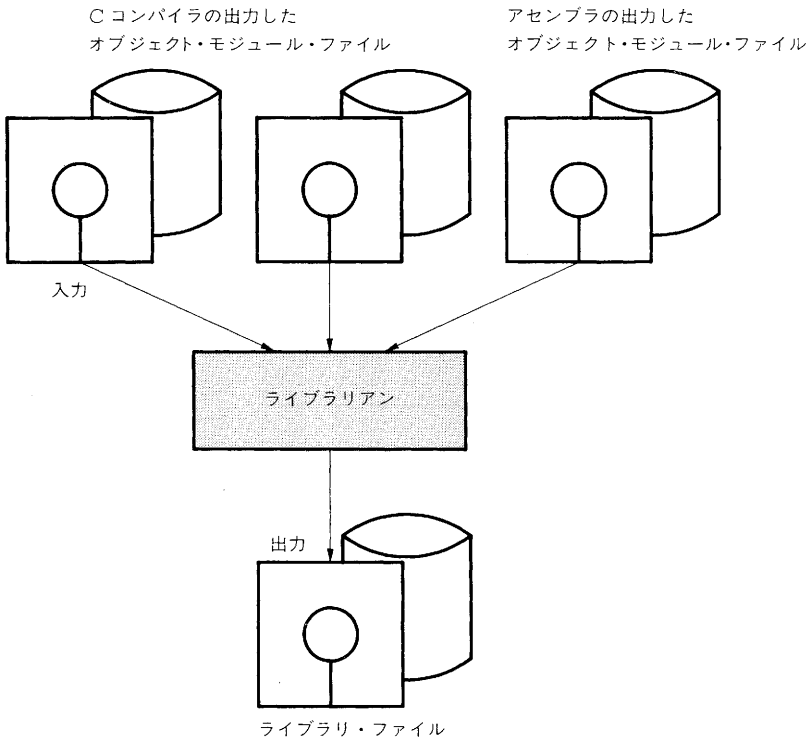
1.2.6 ライブラリアン

ライブラリ・ファイルの作成と更新にライブラリアンを使用します。

汎用性のある，インタフェースの明確なモジュールは，ライブラリ化しておくと便利です。ライブラリ化を行うことにより，多くのオブジェクト・モジュールも1つのファイルになり，扱いやすくなります。

リンカには，ライブラリ・ファイルの中から必要なモジュールだけを取り出してリンクする機能があります。したがって，複数のモジュールを1つのライブラリ・ファイルに登録しておけば，リンク時に必要なモジュール・ファイル名をいちいち指定する必要はなくなります。

図1－14　ライブラリアンの機能



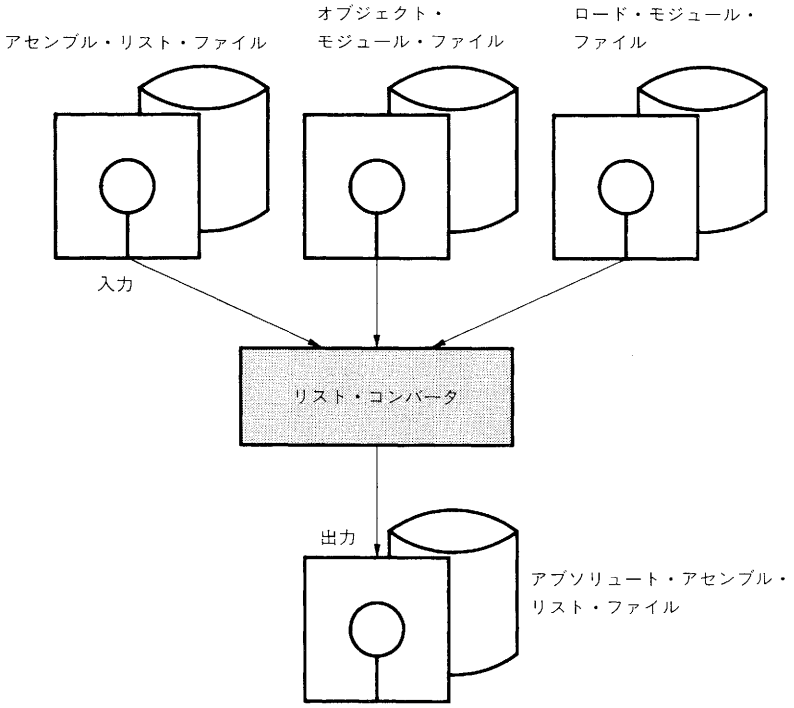
保守／廃止

1.2.7 リスト・コンバータ

リスト・コンバータは，アセンブラの出力したオブジェクト・モジュール・ファイルとアセンブル・リスト・ファイルおよびリンカの出力したロード・モジュール・ファイルを入力し，アブソリュート・アセンブル・リスト・ファイルを出力します。

リロケータブルなアセンブル・リストでは，リスト中のアドレスやリロケータブルな値は，実際の値とは異なるなどの欠点があります。しかし，アブソリュート・アセンブル・リストでは，これらが解決されるので，ディバグやプログラムの保守が容易になります。

図1－15　リスト・コンバータの機能

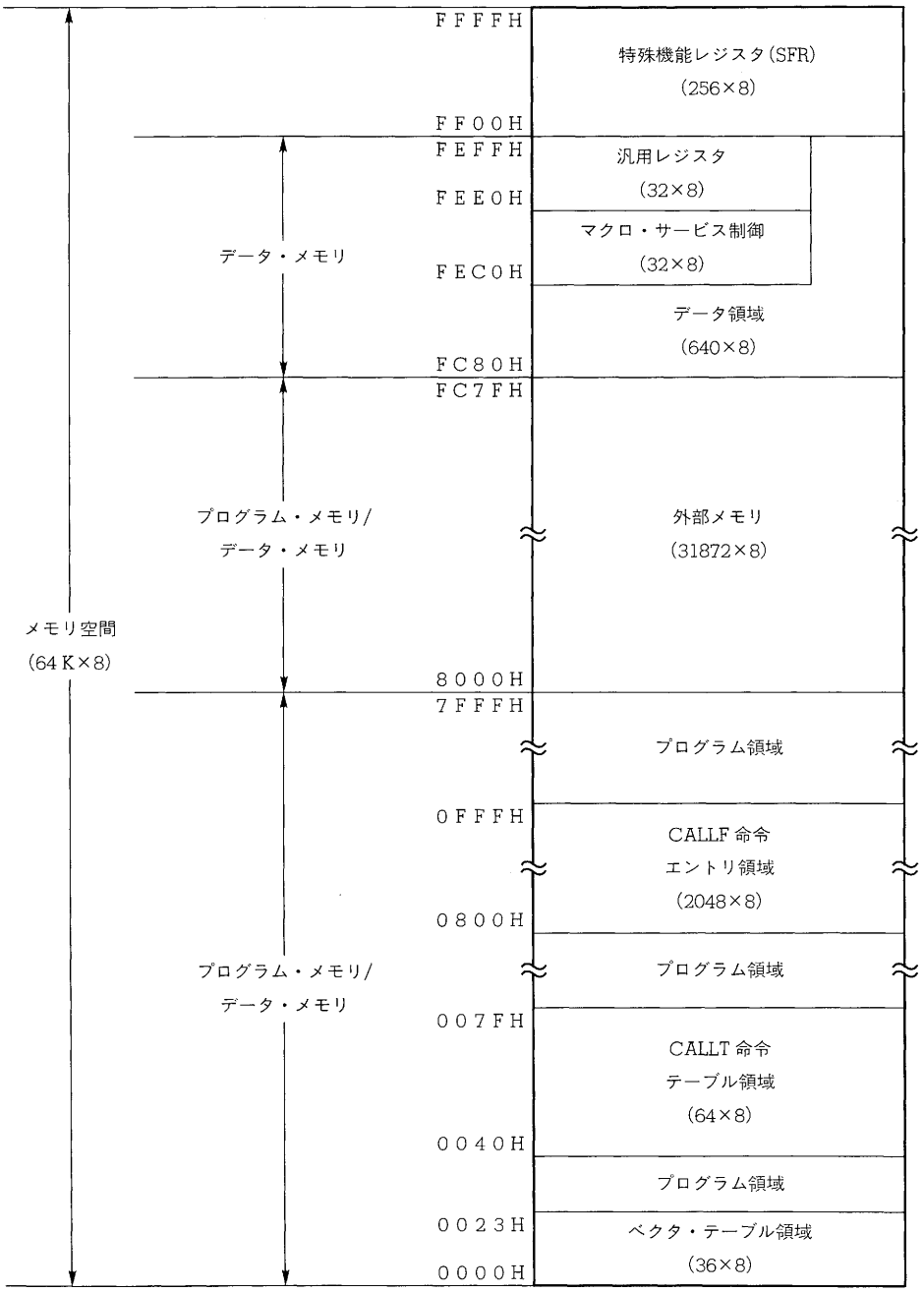


保守／廃止

1.3 メモリ・マップ

78K/I シリーズのメモリ・マップの例として，μPD78138 のメモリ・マップを以下に示します。

図1-16 μPD78138 のメモリ・マップ



内部 ROM，RAM 領域一覧を次に示します。

品　種	内部 ROM 領域	内部 RAM 領域	外部メモリ
78134	0000H-3FFFH	FD80H-FEFFH	4000H-FD7FH
78134A			
78136	0000H-5FFFH	FC80H-FEFFH	6000H-FC7FH
78138	0000H-7FFFH	FC80H-FEFFH	8000H-FC7FH
78P138			
78146	0000H-5FFFH	FC80H-FEFFH	な　し
78148	0000H-7FFFH	FC00H-FEFFH	
78P148			

保守／廃止

1.4 プログラム開発をはじめる前に

実際にプログラム開発をはじめる前に，次のことを頭に入れておいてください。

1.4.1 リンカに入力可能なファイル数

リンカに入力可能なファイル数は128個です。

1.4.2 シンボル数の制限

定義可能なシンボルの数は，次のとおりです。

	ローカル・シンボル数	PUBLICシンボル数
アセンブラ	2900個[注1]	
リンカ	2900×モジュール数	無制限[注2]

注 1. シンボルの種別による制限はありません。なお，未定義シンボルもシンボル総数にカウントされます。

2. PUBLICシンボルが2000個以上の場合は，テンポラリ・ファイルをディスク上に作成しますので，実行速度が遅くなります。

1.4.3 RA78K/I の最大性能

RA78K/I の最大性能を示します。

(1) アセンブラの最大性能

項　　目		最大性能
シンボル長		8文字
1行の文字数		218文字
セグメントの数	ORG疑似命令で定義したセグメント	20個
	CSEG，DSEGおよびBSEG疑似命令で定義したセグメント	80個

(2) リンカの最大性能

項　　目	最大性能
入力ファイル数	128個

1.5 RA78K/I の特徴

RA78K/I は，次の特徴的な機能を備えています。

(1) マクロ機能

ソース・プログラム中で同じ命令群を何回も記述する場合，その一連の命令群を，1つのマクロ名に対応させてマクロ定義をすることができます。

マクロ機能を利用することにより，コーディングの効率を上げることができます。

(2) 分岐命令の最適化機能

分岐命令自動選択疑似命令（BR 疑似命令）を備えています。

メモリ効率のよいプログラムを生成するためには，分岐命令の分岐先範囲に応じて2バイトの分岐命令を記述する必要があります。

しかし，分岐先範囲をいちいち意識して，分岐命令を記述することは面倒です。

BR 疑似命令を記述することにより，アセンブラが分岐先範囲に応じて適切な分岐命令のコードを生成します。これを分岐命令の最適化機能といいます。

(3) 条件付きアセンブル機能

ソース・プログラムの一部分を条件によりアセンブルするかしないかを設定できます。

ソース・プログラム中にディバグ文などを記述した場合，ディバグ文を機械語に変換するか，しないかを条件付きアセンブルのスイッチ設定により選択できます。ディバグ文が必要なくなった場合でも，ソース・プログラムに大幅な変更を加えることなくアセンブルできます。

第２章　製品概要

この章では，RA78K/Ｉ で提供するプログラム・ファイル，また，その動作環境などの概要について説明します。

保守／廃止

2.1　プログラム・ファイル

RA78K/Ⅰに含まれるプログラム・ファイルを，“**表2-1　プログラム・ファイル**”に示します。

表2-1　プログラム・ファイル

プログラム・ファイル名	
①　構造化アセンブラ・プリプロセッサ	
ST78K1.EXE	（コマンド・ファイル）
ST78K1.OMn	（オーバレイ・ファイル）
	n＝1，2，3
TEST1.S	（サンプル・プログラム・ファイル）
TEST2.S	（　　〃　　）
TESTINC.S	（　　〃　　）
ST.BAT	（バッチ・ファイル）
②　リロケータブル・アセンブラ	
RA78K1.EXE	（コマンド・ファイル）
RA78K1.OMn	（オーバレイ・ファイル）
	n＝1，2，3，4，5
RA78K1.HLP	（ヘルプ・ファイル）
③　リンカ	
LK78K1.EXE	（コマンド・ファイル）
RA78K1.OM1	（オーバレイ・ファイル：アセンブラと共用）
LK78K1.HLP	（ヘルプ・ファイル）
④　オブジェクト・コンバータ	
OC78K1.EXE	（コマンド・ファイル）
RA78K1.OM1	（オーバレイ・ファイル：アセンブラと共用）
OC78K1.HLP	（ヘルプ・ファイル）
⑤　ライブラリアン	
LB78K1.EXE	（コマンド・ファイル）
RA78K1.OM1	（オーバレイ・ファイル：アセンブラと共用）
LB78K1.HLP	（ヘルプ・ファイル）
⑥　リスト・コンバータ	
LCNV78K1.EXE	（コマンド・ファイル）
RA78K1.OM1	（オーバレイ・ファイル：アセンブラと共用）
LCNV78K1.HLP	（ヘルプ・ファイル）
⑦　その他の供給ファイル	
78K1MAIN.ASM	（サンプル・プログラム・ファイル）
78K1SUB.ASM	（　　〃　　）

備考1. コマンド・ファイルは，各プログラムが起動されたときに最初にメモリ内に読み込まれるファイルです。

2. オーバレイ・ファイルは，各プログラムの実行中に必要となる場合にだけメモリ内に読み込まれるファイルです。

保守／廃止

2.2　ホスト・マシン対象機種と対象 OS

RA78K/I を動作できるホスト・マシンと，OS について説明します。

2.2.1　PC-9800 シリーズ

(1) PC-9800 シリーズの対象機種

表 2-2　ホスト・マシン対象機種一覧（PC-9800 シリーズ）

CPU	8086/V30	80286		80386
モード	ノーマル	ノーマル	ハイレゾ	ノーマル
	無印 E F1/2/3 M2/3 VF2 VM0/2/4/21/11 U2 UV2/21/11 CV21 UR/20 UF		XA model 1/2/3/11 /21/31	
	XL model 1/2/4 VX0/2/4/01/21/41 UX21/41 RX2/4/21/51 EX2/4	XL model 1/2/4 VX0/2/4/01/21/41 UX21/41 RX2/4/21/51 EX2/4 DX2/5/U2/U5	XL model 1/2/4	XL[2]（ハイレゾ可）
	XL[2] RL2/5/21/51 RA2/5/21/51 ES2/5 RS21/51 T model W2/W5/S5/F5 LV21/22 LX2/4/5 LS2/5 N NV	LX2/4/5		RL2/5/21/51 （ハイレゾ可） RA2/5/21/51 ES2/5 RS21/51 T model W2/W5/S5/F5 DS2/5/U2/U5 DA2/5/7 U2/U5/U7 LS2/5 NS/-20

注意　内蔵メモリは 384K 以上必要です。

(2) PC-9800 シリーズの対象 OS

MS-DOS™ ver 2.11/3.10/3.30/3.30A/3.30B/3.30C

(3) プログラム・ファイル供給媒体

5 インチ FD　　(2HD)

3.5 インチ FD　(2HD)

2.2.2 IBM PC シリーズ

(1) IBM PC シリーズの対象機種

IBM PC™, PC/AT™, PC/XT™

(2) IBM PC シリーズの対象 OS

PC DOS ver.3.10

(3) プログラム・ファイル供給媒体

5 インチ FD　　(2HC)

RA78K/Ⅰ のプログラムを正常に動作するために，次に示す注意事項をお守りください。

注意 1. RA78K/Ⅰ の各プログラムは，NEC 製の PC-9800 シリーズ用 MS-DOS 上にて正常に動作します。

このほかの市販の MS-DOS 上におけるプログラムの動作については，その責任を負いかねますのでご了承ください。

2. CONFIG.SYS の設定では，FILES は 13 以上を設定してください（FILES = 13)。

3. 内蔵メモリ・サイズは 384K 以上必要です。また，メモリ・サイズは RA78K/Ⅰ の各プログラムが動作時に必要とするメモリの最大値であり，システム領域は含みません。

第３章　RA78K/Ｉの実行

この章では，RA78K/Ｉの実行手順を示します。この章で説明する実行手順に従って実際に各プログラムを実行することにより，RA78K/Ｉの操作に慣れることができます。

なお，この章以降のすべての操作例は，PC-9800シリーズ（MS-DOS）上で，行った場合のものです。

3.1　RA78K/Ｉ実行の前に

3.1.1　ディスク内容の確認

RA78K/Ｉのシステム・ディスクに，“**2.1　プログラム・ファイル**”で紹介した，すべてのファイルがあることを確認してください。

3.1.2　サンプル・プログラム

システム・ディスクに格納されているファイルのうち「78K1MAIN.ASM」，「78K1SUB.ASM」は，動作確認用サンプル・プログラムのファイルです。

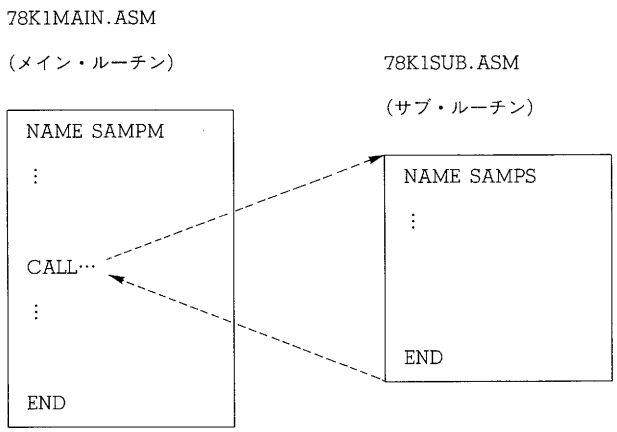
これらのファイルは，以降のアセンブラの操作でソース・プログラム・ファイルとして，アセンブラへの入力フアイルとなります。

次に，サンプル・プログラムの内容について簡単に説明します。このサンプル・プログラムは，16進数のデータをASCIIコードに変換するもので，１つのプログラムをメイン・ルーチンとサブルーチンの２つのモジュールに分けています。

メイン・ルーチンのモジュール名はSAMPMで，ソース・モジュール・ファイル（78K1MAIN.ASM）に格納されています。

また，サブルーチンのモジュール名はSAMPSで，ソース・モジュール・ファイル（78K1SUB.ASM）に格納されています。

図３-１　サンプル・プログラムの構造



■78K1MAIN.ASM（メイン・ルーチン）

```
A>type 78k1main.asm

$       PROCESSOR(138)

        NAME    SAMPM
;************************************************
;*                                              *
;*      HEX -> ASCII Conversion Program         *
;*                                              *
;*              main-routine                    *
;*                                              *
;************************************************

        PUBLIC  MAIN,START
        EXTRN   CONVAH

        DSEG
STASC:  DS      2

        CSEG
        ORG     0H
MAIN:   DW      START

        CSEG
START:  MOVW    SP,#0FEE0H
        MOV     MM,#00
        MOV     STBC,#00

        MOV     E,#LOW HDTSA            ;set hex 2-code data in HL registor

        CALL    !CONVAH                 ;convert ASCII <- HEX
                                        ;output BC-register <- ASCII code
        MOVW    DE,#STASC               ;set DE <- store ASCII code table
        MOV     A,B
        MOV     [DE+],A
        MOV     A,C
        MOV     [DE+],A

        BR      $$

DATA    DSEG    AT 0FE00H
HDTSA:  DB      1AH

        END
```

■78K1SUB.ASM（サブルーチン）

```
A>type 78k1sub.asm

$       PROCESSOR(138)

        NAME    SAMPS
;*****************************************************
;*                                                   *
;*   HEX -> ASCII Conversion Program                 *
;*                                                   *
;*              sub-routine                          *
;*                                                   *
;* input condition  : (FE00H + E-reg) <- hex 2 code *
;*                                                   *
;* output condition : BC-reg <-ASCII 2 code          *
;*                                                   *
;*****************************************************

        PUBLIC  CONVAH

        CSEG
CONVAH: MOV     A,#0
        ROL4    [E]             ;hex upper code load
        CALL    !SASC
        MOV     B,A             ;store result

        MOV     A,#0
        ROL4    [E]             ;hex lower code load
        CALL    !SASC
        MOV     C,A             ;store result

        RET

;*******************************************************
;* subroutine   convert ASCII code                    *
;*      input   Acc (lower 4bits) <- hex code         *
;*      output  Acc               <- ASCII code       *
;*******************************************************

        CSEG
SASC:   CMP     A,#0AH          ;check hex code > 9
        BC      $SASC1
        ADD     A,#07H          ;bias(+7)
SASC1:  ADD     A,#30H          ;bias(+30)
        RET

        END
```

備考　本サンプル・プログラムは，RA78K/I の機能や操作を習得していただく目的で用意した参考プログラムです。したがって，アプリケーション・プログラムとして，そのまま利用することはできません。

保守／廃止

3.2　RA78K/Ｉの実行手順

RA78K/Ｉの基本的な実行手順を紹介します。

(1) サンプル・プログラム（78K1MAIN.ASM）をアセンブルします。

コマンド行には次のように入力します。

```
A > ra78k1 78k1main.asm -g
```

次のメッセージがディスプレイに出力されます。

```
78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

(2) ドライブＡの内容をチェックします。

アセンブラは，オブジェクト・モジュール・ファイル(78K1MAIN.REL)とアセンブル・リスト・ファイル（78K1MAIN.PRN）を出力します。

なお，アセンブル時に'－Ｅ'オプションを指定することにより，アセンブラはエラー・リスト・ファイル（アセンブル・エラーになった行とそのエラー・メッセージの内容のリスト）を出力します。

(3) サンプル・プログラム（78K1SUB.ASM）をアセンブルします。

コマンド行には次のように入力します。

```
A > ra78k1 78k1sub.asm -g
```

次のメッセージがディスプレイに出力されます。

```
78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

(4) ドライブＡの内容をチェックします。

アセンブラは，オブジェクト・モジュール・ファイル（78K1SUB.REL）とアセンブル・リスト・ファイル（78K1SUB.PRN）を出力します。

なお，アセンブル時に'－E'オプションを指定することにより，アセンブラはエラー・リスト・ファイルを出力します。

(5) ディレクティブ・ファイルを作成します。

ディレクティブ・ファイルは，リンカに対してセグメントの配置を指定します。

したがって，セグメントを配置するためにディフォールトのROM/RAM領域を拡張したり，新たにメモリ領域を定義する必要のある場合には，ディレクティブ・ファイルを作成します。また，ソース・モジュール中でアブソリュート・セグメントとして定義していないセグメントをメモリ上の特定番地に配置しようとする場合にもディレクティブ・ファイルを作成する必要があります。

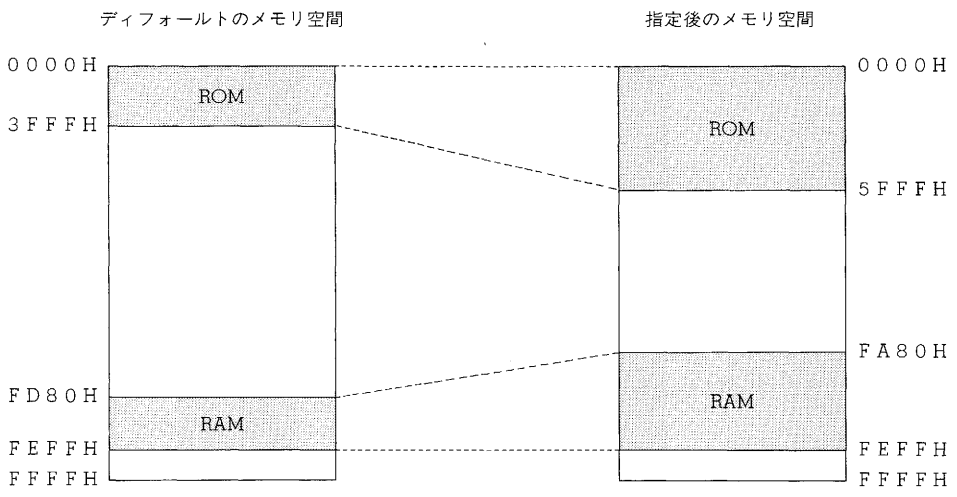
ディレクティブ・ファイルは，リンク時に'－D'オプションを用いてリンカに入力します。

例　ディフォールトのROM/RAM領域を拡張し，セグメントCSEG1を2000H番地に配置する場合

ディレクティブ・ファイルには，次のように記述します（μPD78134Aの場合）。

```
MEMORY ROM : (0H, 5FFFH)
MEMORY RAM : (FA80H, 1FFH)
MERGE CSEG1 : AT (2000H)
```

図３－２　リンク・ディレクティブ



(6) アセンブルの結果，出力されたオブジェクト・モジュール・ファイル「78K1MAIN.REL」と「78K1SUB.REL」をリンクします。

また，ディレクティブ・ファイルとして，78K1.DR を入力します。

コマンド行には次のように入力します。

```
A > lk78k1 78k1main.rel 78k1sub.rel -d78k1.dr -o78k1.lnk -p78k1.map -g
                                    └── ディレクティブ・ファイルを指定
                                        しない場合は不要
```

次のメッセージがディスプレイに出力されます。

```
78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

(7) ドライブ A の内容をチェックします。

リンカは，ロード・モジュール・ファイル(78K1.LNK)とリンク・リスト・ファイル(78K1.PRN)を出力しました。

なお，リンク時に '－ E' オプションを指定することにより，リンカはエラー・リスト・ファイルを出力します。

(8) リンクの結果出力されたロード・モジュール・ファイル (78K1.LNK) を HEX 形式ファイルに変換します。

コマンド行には次のように入力します。

```
A > oc78k1 78k1.lnk
```

次のメッセージがディスプレイに出力されます。

```
78K/I Series Object Converter Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

(9) ドライブ A の内容をチェックします。

オブジェクト・コンバータは，HEX 形式オブジェクト・モジュール・ファイル (78K1.HEX) とシンボル・テーブル・ファイル (78K1.SYM) を出力しました。

(10) ライブラリ・ファイルを作成します。

アセンブラの出力したオブジェクト・モジュール・ファイル(78K1SUB.REL)をライブラリ・ファイルとして登録します。

コマンド行には次のように入力します。

```
A > lb78k1 < 78k1.job
```

次のメッセージがディスプレイに出力されます。

```
78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*create 78k1.lib
*add 78k1.lib 78k1sub.rel
*exit
```

(11) ドライブ A の内容をチェックします。

ライブラリアンは，ライブラリ・ファイル（78K1.LIB）を出力しました。

(12) アブソリュート・アセンブル・リストを作成します。

78K1MAIN.ASM のアブソリュート・アセンブル・リストを作成するため「78K1MAIN.REL」，「78K1MAIN.ASM」および「78K1.LNK」をリスト・コンバータに入力します。

コマンド行には次のように入力します。

```
A > lcnv78k1 78k1main -l78k1.lnk
```

次のメッセージがディスプレイ出力されます。

```
List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

(13) ドライブ A の内容をチェックします。

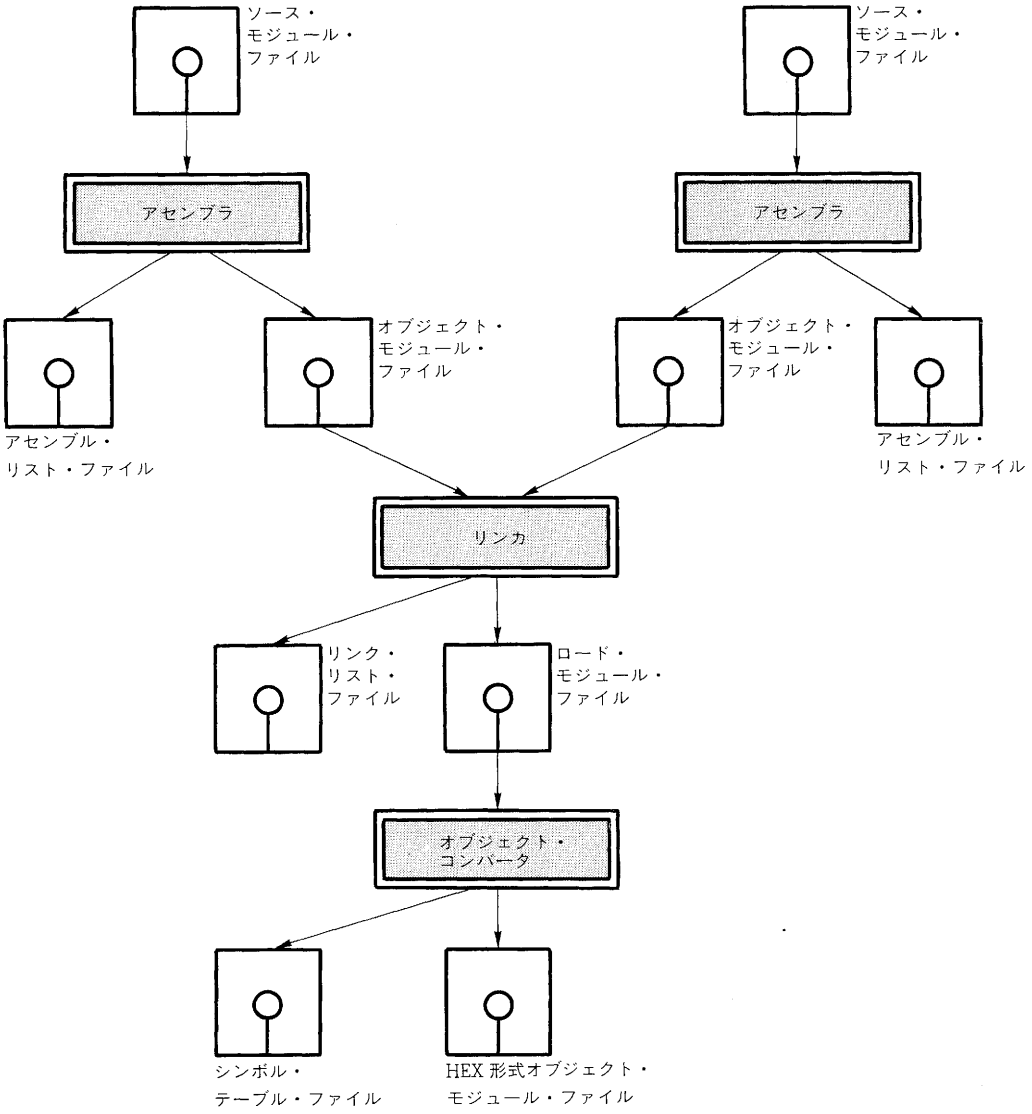
リスト・コンバータは，アブソリュート・アセンブル・リスト・ファイル（78K1MAIN.P）を出力しました。

保守／廃止

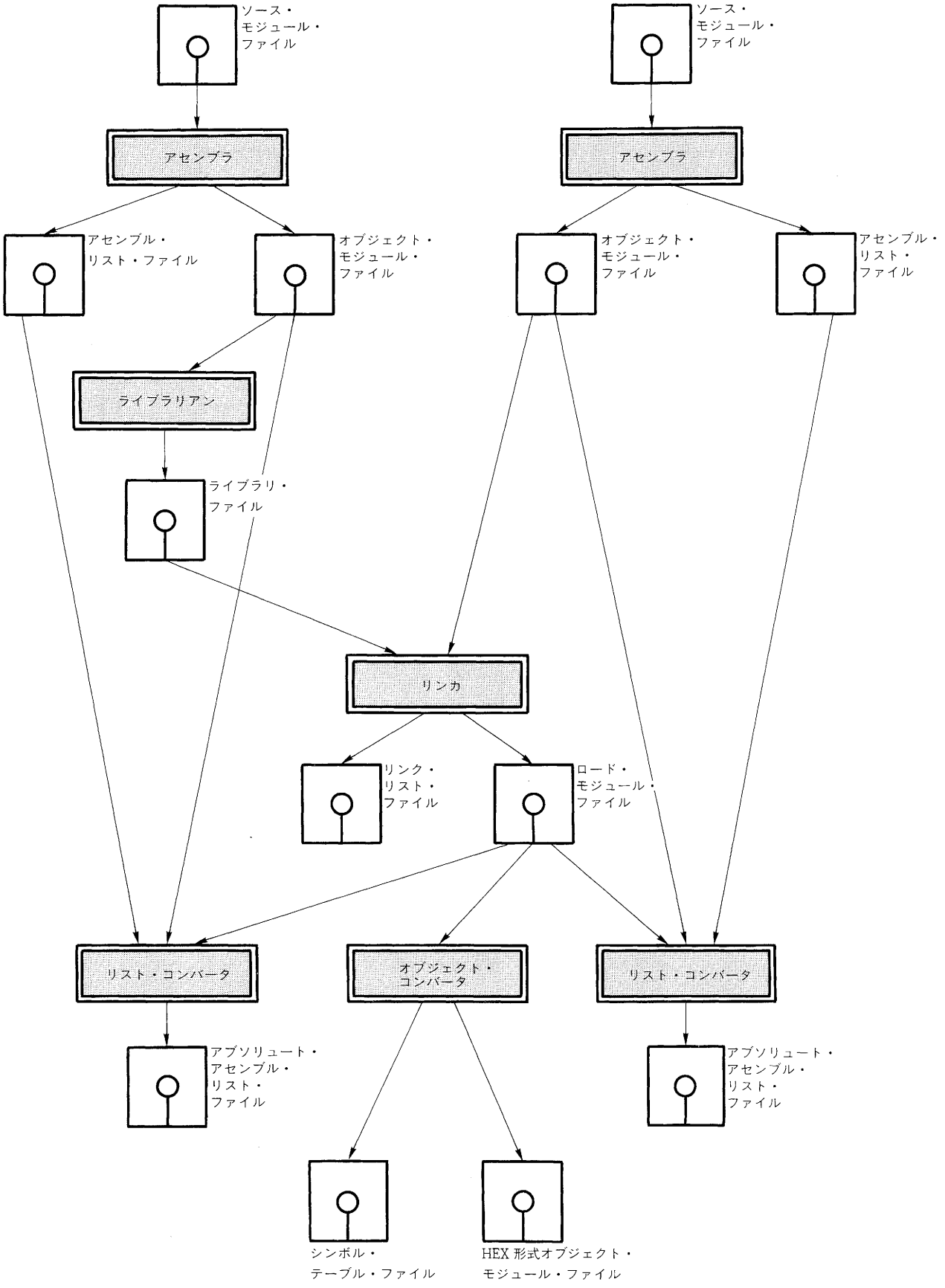
3.3　RA78K/Ⅰの実行手順のまとめ

“3.2　RA78K/Ⅰの実行手順”をまとめると次のようになります。

(1) 実行手順 1

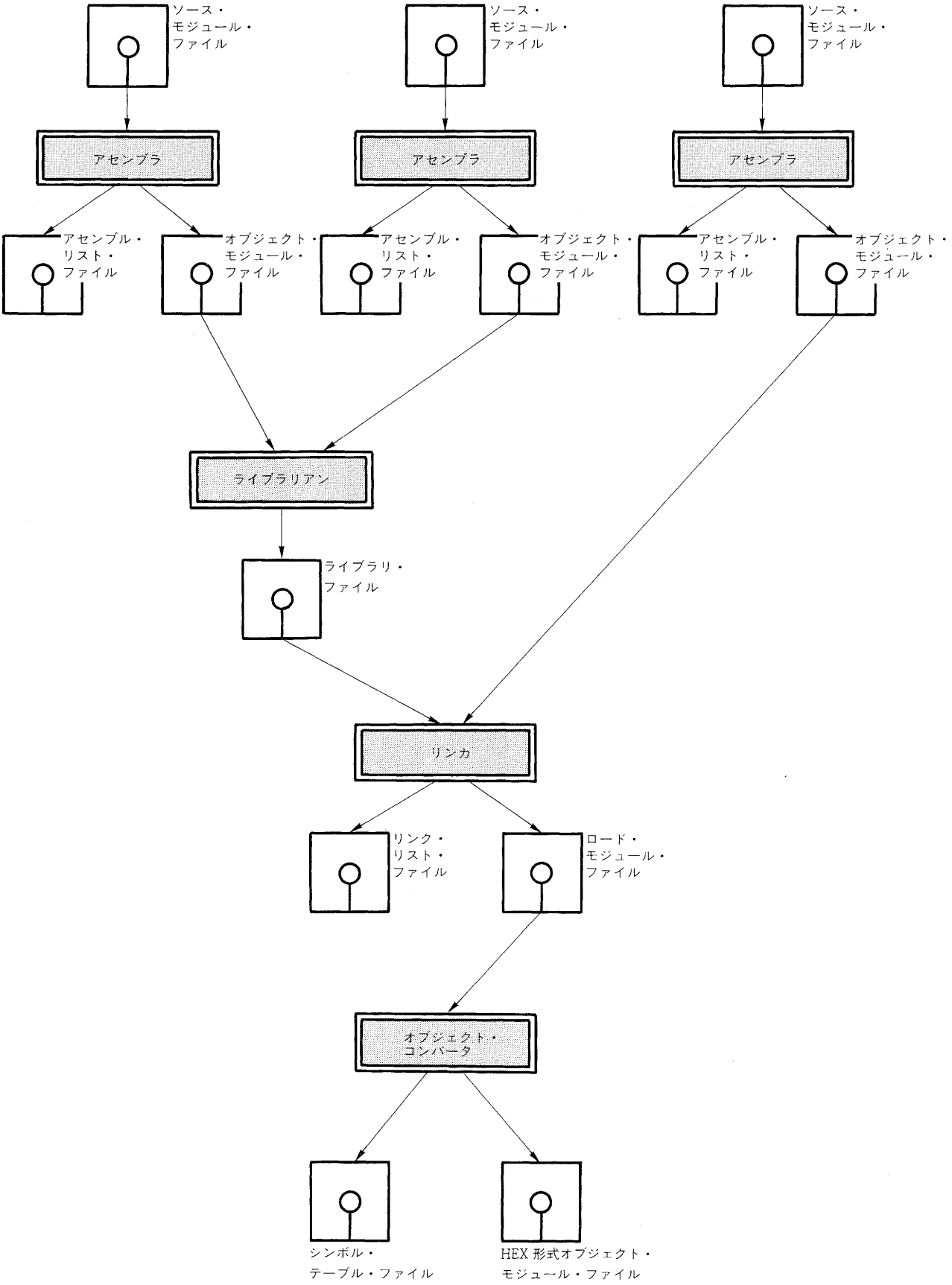


(2) 実行手順2


ソース・
モジュール・
ファイル
ソース・
モジュール・
ファイル
アセンブラ
アセンブラ
アセンブル・
リスト・ファイル
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
アセンブル・
リスト・
ファイル
ライブラリアン
ライブラリ・
ファイル
リンカ
リンク・
リスト・
ファイル
ロード・
モジュール・
ファイル
リスト・コンバータ
オブジェクト・
コンバータ
リスト・コンバータ
アブソリュート・
アセンブル・
リスト・
ファイル
アブソリュート・
アセンブル・
リスト・
ファイル
シンボル・
テーブル・ファイル
HEX 形式オブジェクト・
モジュール・ファイル

保守／廃止

(3) 実行手順３


ソース・
モジュール・
ファイル
ソース・
モジュール・
ファイル
ソース・
モジュール・
ファイル
アセンブラ
アセンブラ
アセンブラ
アセンブル・
リスト・
ファイル
オブジェクト・
モジュール・
ファイル
アセンブル・
リスト・
ファイル
オブジェクト・
モジュール・
ファイル
アセンブル・
リスト・
ファイル
オブジェクト・
モジュール・
ファイル
ライブラリアン
ライブラリ・
ファイル
リンカ
リンク・
リスト・
ファイル
ロード・
モジュール・
ファイル
オブジェクト・
コンバータ
シンボル・
テーブル・ファイル
HEX 形式オブジェクト・
モジュール・ファイル

保守／廃止

保守／廃止

第４章　アセンブラ

アセンブラは，78K/Ⅰシリーズのアセンブリ言語で記述されたソース・モジュール・ファイルを入力し，それを機械語に変換してオブジェクト・モジュール・ファイルとして出力します。

さらに，アセンブル・リスト・ファイルやエラー・リスト・ファイルなどのリスト・ファイルを出力します。

アセンブル・エラーがある場合は，エラー・メッセージをアセンブル・リスト・ファイルやエラー・リスト・ファイルに出力し，エラーの原因を明示します。

保守／廃止

4.1　アセンブラの入出力ファイル

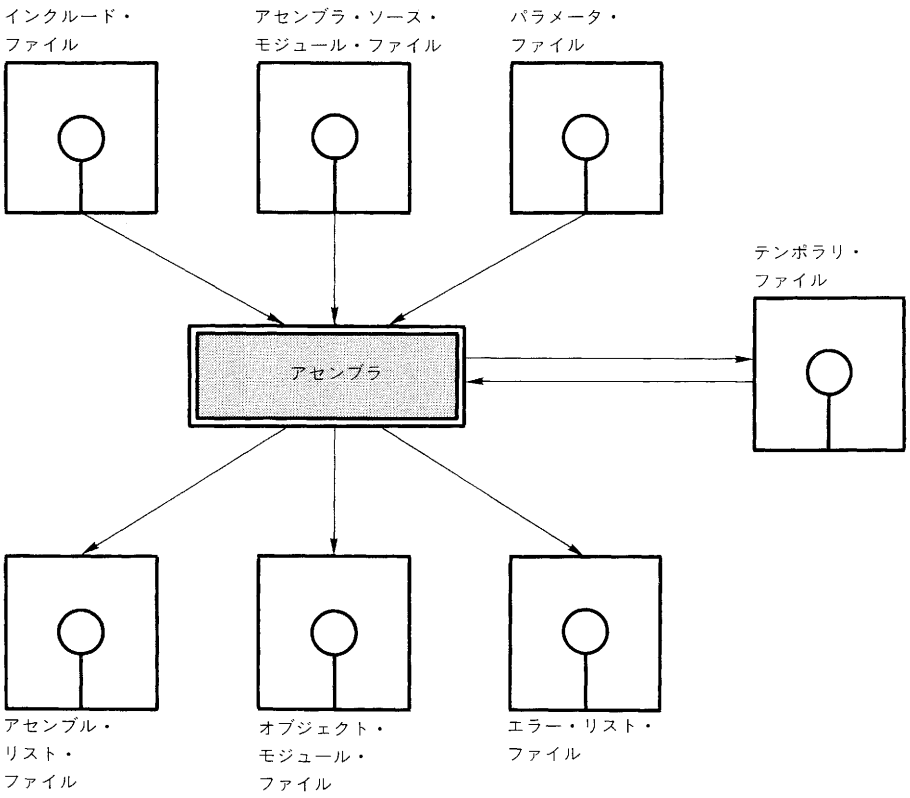
アセンブラの入出力ファイルを以下に示します。

表4-1　アセンブラの入出力ファイル

種類	ファイル名	説　　明	ディフォールト・ファイル・タイプ
入力ファイル	アセンブラ・ソース・モジュール・ファイル	○78K/Ⅰシリーズのアセンブリ言語で記述されたソース・モジュール・ファイルです。 ○ユーザ作成ファイルです。	.ASM
	インクルード・ファイル	○アセンブラ・ソース・モジュール・ファイルで参照するファイルです。 ○78K/Ⅰシリーズのアセンブリ言語で記述されたファイルです。 ○ユーザ作成ファイルです。	なし
	パラメータ・ファイル	○実行プログラムのパラメータを内容とするファイルです。 ○ユーザ作成ファイルです。	.PRA
出力ファイル	オブジェクト・モジュール・ファイル	○機械語情報と機械語の配置アドレスに関する再配置情報およびシンボル情報を含んだバイナリ・ファイルです。	.REL
	アセンブル・リスト・ファイル	○アセンブル・リスト，クロスレファレンス・リストなどのアセンブル情報を持つファイルです。	.PRN
	エラー・リスト・ファイル	○アセンブル時のエラー情報を持つファイルです。	.ERA
入出力ファイル	テンポラリ・ファイル	○アセンブルのためにアセンブラが自動生成するファイルです。アセンブル終了時には消去されます。	RA××××××.$$n (n = 1~4)

保守／廃止

図4－1　アセンブラの入出力ファイル



保守／廃止

4.2　アセンブラの機能

(1) アセンブラは，ソース・モジュール・ファイルを読み込み，アセンブリ言語を機械語に変換します。

(2) アセンブラは，ファイルやシステムに関するエラーがある場合は，アボート・エラーを出力し，ソース・モジュール中に記述エラーを発見した場合は，フェイタル・エラーやワーニング・エラーを出力します。

アボート・エラー，フェイタル・エラーの場合は，通常オブジェクト・モジュール・ファイルは出力されません。ただし，'－J' オプションが指定された場合には，フェイタル・エラーがある場合でもオブジェクト・モジュール・ファイルを出力します。

(3) アセンブラは，アセンブル起動時に指定されたアセンブラ・オプションに従ってアセンブル処理を行います。アセンブラ・オプションの詳細については，“**4.4　アセンブラ・オプション**”をお読みください。

(4) アセンブラは，その処理を正常に終了すると終了メッセージを出力し，制御を OS に戻します。

(5) アセンブラ・パッケージの最大性能を次に示します。

項　　目		制　限
シンボル長		8 文字
1 行の文字数		218 文字
セグメントの数	ORG 疑似命令で定義したセグメント	20 個
	CSEG，DSEG および BSEG 疑似命令で定義したセグメント	80 個

保守／廃止

4.3　アセンブラの起動方法

4.3.1　アセンブラの起動

アセンブラの起動には，次の２つの方法があります。

(1) コマンド行での起動

X＞[パス名] RA78K1 [△オプション]…△ソース・モジュール・ファイル名 [△オプション]…[△]
①　②　③　④　⑤　④

① OSのプロンプト。

② アセンブラのコマンド・ファイルを格納しているパス名。

③ アセンブラのコマンド・ファイル名。

MS-DOS ver2.11の場合，すべてのプログラム・ファイルはカレント・ディレクトリになければなりません。

④ アセンブラに対して動作の詳細を指示します。

複数のアセンブラ・オプションを指定する場合は，おのおののアセンブラ・オプション間を空白で区切ってください。アセンブラ・オプションの詳細については，“**4.4　アセンブラ・オプション**”をお読みください。

⑤ アセンブルするソース・モジュール・ファイル名。

例　A＞ra78k1 -g 78k1main.asm -e -np

(2) パラメータ・ファイルによる起動

パラメータ・ファイルは，起動に必要な情報がコマンド行に指定しきれない場合やアセンブルするたびに同じオプションを繰り返し指定するような場合に使用します。

パラメータ・ファイルを使用する場合には，コマンド行にパラメータ・ファイル指定オプション（－F）を指定します。

パラメータ・ファイルによる起動方法は，次のようになります。

X＞RA78K1［△ソース・モジュール・ファイル］△－Fパラメータ・ファイル名
　　①　　②

① パラメータ・ファイルの指定オプションです。
　パラメータ・ファイルは，エディタなどで作成します。
② アセンブラの起動に必要な情報を含んだファイルです。

パラメータ・ファイル内での記述規則を次に示します。

[[[△] オプション［△オプション］…［△］△]] …

コマンド行でソース・モジュール・ファイル名を省略した場合は，パラメータ・ファイル内でソース・モジュール・ファイル名を1つだけ指定できます。

ソース・モジュール・ファイル名は，オプションのあとに記述することも可能です。

パラメータ・ファイルには，コマンド行で指定するすべてのアセンブラ・オプション，出力ファイル名を記述します。

パラメータ・ファイルについての詳細は“**4.4.3　アセンブラ・オプションの説明**”をお読みください。

例　パラメータ・ファイル（78K1MAIN.PRA）をエディタで作成します。

●78K1MAIN.PRAの内容

```
;parameter file
78k1main.asm -osample.rel -g
-psample.prn
```

●パラメータ・ファイル（78K1MAIN.PRA）を使用してアセンブラを起動します。

```
A＞ra78k1 -f78k1main.pra
```

保守／廃止

4.3.2　実行開始，終了メッセージ

(1) 実行開始メッセージ

アセンブラが起動すると，ディスプレイに実行開始メッセージが表示されます。

```
78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
```

(2) 実行終了メッセージ

アセンブルの結果，アセンブル・エラーが検出されなかった場合，アセンブラは次のメッセージをディスプレイに出力して制御をOSに戻します。

```
Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

アセンブルの結果，アセンブル・エラーが検出された場合，アセンブラはエラーの数をディスプレイに出力して制御をOSに戻します。

```
Pass1 Start
78K1MAIN.ASM(20) : F201 Syntax error
Pass2 Start
78K1MAIN.ASM(20) : F201 Syntax error

Assembly complete,      1 error(s) and      0 warning(s) found.
```

アセンブル中にアセンブラ処理継続が不可能な致命的エラーが検出された場合，アセンブラはメッセージをディスプレイに出力してアセンブルを中止し，制御をOSに戻します。

例1. 存在しないソース・モジュール・ファイルを指定した場合

```
A>ra78k1 sample.asm

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

A006 File not found 'SAMPLE.ASM'
Program aborted.
```

上記の例では，存在しないソース・モジュール・ファイルを指定したためにエラーとなり，アセンブルを中止しました。

例 2. 存在しないアセンブラ・オプションを指定した場合

```
78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

A018 Option is not recognized '-w'
Program aborted.
```

この例では，存在しないアセンブラ・オプションを指定したためにエラーとなり，アセンブルを中止しました。

アセンブラがエラー・メッセージを出力してアセンブルを中止した場合，そのエラー・メッセージの原因を“**第11章　エラー・メッセージ**”で調べて対処してください。

保守／廃止

4.4　アセンブラ・オプション

4.4.1　アセンブラ・オプションの種類

アセンブラ・オプションは，アセンブラの動作に細かい指示を与えるものです。アセンブラ・オプションは，12種類のオプションに分類できます。

表4-2　アセンブラ・オプション（1/2）

項　番	分　　類	オプション	説　　明
1	デバイス種別指定	－C	対象デバイスの種別を指定します。
2	オブジェクト・モジュール・ファイル出力指定	－O	オブジェクト・モジュール・ファイルの出力を指定します。
		－NO	
3	オブジェクト・モジュール・ファイル強制出力指定	－J	強制的にオブジェクト・モジュール・ファイルを出力します。
		－NJ	
4	ディバグ情報出力指定	－G	ディバグ情報をオブジェクト・モジュール・ファイルへ出力します。
		－NG	
5	インクルード・ファイル読み込みパス指定	－I	インクルード・ファイルを指定したパスから読み込みます。
6	アセンブル・リスト・ファイル出力指定	－P	アセンブル・リスト・ファイルの出力を指定します。
		－NP	
7	アセンブル・リスト・ファイル情報指定	－KA	アセンブル・リスト・ファイル中にアセンブル・リストを出力します。
		－NKA	
		－KS	アセンブル・リスト・ファイル中にシンボル・リストを出力します。
		－NKS	
		－KX	アセンブル・リスト・ファイル中にクロスレファレンス・リストを出力します。
		－NKX	
8	アセンブル・リスト・ファイル形式指定	－LW	アセンブル・リスト・ファイルの1行に印字する文字数を変更します。
		－LL	アセンブル・リスト・ファイルの1頁に印字する行数を変更します。
		－LH	アセンブル・リスト・ファイルのヘッダに指定された文字列を出力します。
		－LT	タブの展開文字数を変更します。
		－LF	アセンブル・リスト・ファイルの最後に改頁コードを付加します。
		－NLF	
9	エラー・リスト・ファイル出力指定	－E	エラー・リスト・ファイルを出力します。
		－NE	
10	パラメータ・ファイル指定	－F	入力ファイル名，オプションを指定したファイルより入力します。

表4-2　アセンブラ・オプション（2/2）

項　番	分　　類	オプション	説　　明
11	テンポラリ・ファイル作成パス指定	－T	テンポラリ・ファイルを指定したパスに作成します。
12	ヘルプ指定	－－	ディスプレイにヘルプ・メッセージを出力します。

備考　この表は，アセンブラ・オプションを紹介するための表です。実際にアセンブラ・オプションを使用する場合には，“**付録C.1　アセンブラ・オプション一覧**”をお読みください。

4.4.2　アセンブラ・オプションの優先度

次の表に示すアセンブラ・オプションの内，縦軸のものと横軸のものを同時に2つ以上指定した場合の優先度について説明します。

表4-3　アセンブラ・オプションの優先度

	－NO	－NP	－NKA	－NKS	－KX	－NKX	－－
－J	×						×
－G	×						×
－P			△	△		△	×
－KA		×					×
－KS		×			×		×
－KX		×					×
－LW		×					×
－LL		×					×
－LH		×					×
－LT		×					×
－LF		×					×

← 横軸

↑
縦軸

【×で記した箇所】

横軸に示したオプションを指定した場合，縦軸に示したオプションは無効となります。

例　A＞ra78k1 -c138 78k1main.asm -np -lw80 -lf

－LW，－LFオプションは，無効となります。

保守／廃止

【△で記した箇所】

横軸に示したオプション３つをすべて同時に指定した場合，縦軸に示したオプションは無効となります。

例　A＞ra78k1 -c138 78k1main.asm -p -nka -nks -nkx

－NKA，－NKSおよび－NKXが同時に指定されたので，－Pオプションは無効となります。

また，－O/－NOオプションのようにオプション名の前に'N'を付加できるオプションを同時に指定した場合，あとで指定した方が有効となります。

例　A＞ra78k1 -c138 78k1main.asm -o -no

－NOオプションがあとに指定されているので，－Oオプションは無効となり，－NOオプションが有効となります。

“**表４－３　アセンブラ・オプションの優先度**”に記述されていないオプションは，他のオプションの影響を特に受けません。しかし，ヘルプ指定オプション‘－－’が指定された場合には，すべてのオプション指定が無効となります。

保守／廃止

4.4.3　アセンブラ・オプションの説明

各アセンブラ・オプションの詳細を説明します。

(1) デバイス種別指定　(－C)

記述形式　：－Ｃ　デバイス種別

省略時解釈：省略不可

【機　　能】

－Ｃオプションで，アセンブルの対象となる対象デバイスを指定します。

【用　　途】

－Ｃオプションは必ず指定してください。アセンブラは指定された対象デバイスに対してアセンブルを行い，それに対応したオブジェクト・コードを生成します。

【説　　明】

－Ｃオプションで指定できる対象デバイス（デバイス種別）を**表４－４**に示します。

【注　　意】

－Ｃオプションは省略不可能です。ただし，ソース・モジュールの先頭で，－Ｃオプションと同機能の制御命令を記述すれば，コマンド行での指定を省略できます。

```
▲$▲ PROCESSOR ▲（▲デバイス種別▲）
▲$▲ PC ▲（▲デバイス種別▲）              ；短縮形
```

制御命令については言語編“**第４章　制御命令**”をお読みください。

表4-4　対象デバイスとデバイス種別との対応

対象デバイス	デバイス種別
μPD78134	134
μPD78134A	134A
μPD78136	136
μPD78138 μPD78P138	138
μPD78146	146
μPD78148 μPD78P148	148

【使 用 例】

例1. コマンド行で指定します。

```
A>ra78k1 -c138 78k1main.asm

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

例２. ソース・モジュール中で指定してアセンブラを起動します。

《ソース・モジュール》

```
A>type 78k1main.asm

$       PROCESSOR(138)

        NAME    SAMPM
;************************************************
;*                                              *
;*      HEX -> ASCII Conversion Program         *
;*                                              *
;*              main-routine                    *
;*                                              *
;************************************************

        PUBLIC  MAIN,START
        EXTRN   CONVAH
                :
```

ソース・モジュールの先頭で対象デバイスを指定しているので、
コマンド行での対象デバイス指定は省略可能です。

《デバイス種別指定を省略してアセンブラを起動》

```
A>ra78k1 78k1main.asm

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

例3. ソース・モジュール中とコマンド行で異なるデバイス指定をし，アセンブラを起動します。

《ソース・モジュール》

```
A>type 78k1main.asm

$       PROCESSOR(138)

        NAME    SAMPM
;************************************************
;*                                              *
;*      HEX -> ASCII Conversion Program         *
;*                                              *
;*              main-routine                    *
;*                                              *
;************************************************

        PUBLIC  MAIN,START
        EXTRN   CONVAH
                :
```

《ソース・モジュールとは異なる対象デバイスを指定します》

```
A>ra78k1 -c148 78k1main.asm

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
78K1MAIN.ASM(1) : W702 Duplicate PROCESSOR option and control
Pass2 Start
78K1MAIN.ASM(1) : W702 Duplicate PROCESSOR option and control

Assembly complete,      0 error(s) and      1 warning(s) found.
```

ｱｾﾝﾌﾞﾙ･ﾘｽﾄ･ﾌｧｲﾙ(78k1main.prn)の最後の２行を参照すると、コマンド行で指定した対象デバイスを優先していることが分かります。

《78k1main.prnの最後の２行》

```
A>type 78k1main.prn

                                        :
Target chip:uPD78148
Assembly complete,      0 error(s) and      1 warning(s) found. (    1)
```

コマンド行で指定した対象デバイスを優先します。

保守／廃止

(2) オブジェクト・モジュール・ファイル出力指定　(－O/－NO)

記述形式　：－O　[出力ファイル名]
　　：－NO
省略時解釈：－O　入力ファイル名．REL

【機　　能】

① －Oオプションは，オブジェクト・モジュール・ファイルの出力を指定します。また，その出力先や出力ファイル名を指定します。

② －NOオプションは，オブジェクト・モジュール・ファイルを出力しないことを指定します。

【用　　途】

オブジェクト・モジュール・ファイルの出力先や出力ファイル名を変更したいときに，－Oオプションを指定します。

アセンブル・リスト・ファイルの出力のみが目的でアセンブルする場合などは，－NOオプションを指定します。これによりアセンブル時間が短縮されます。

【説　　明】

① －Oオプションを指定しても，フェイタル・エラーがある場合には，オブジェクト・モジュール・ファイルは出力されません。

② －Oオプションを指定する際にドライブ名を省略すると，カレント・ドライブにオブジェクト・モジュール・ファイルが出力されます。

③ －Oオプションを指定する際に出力ファイル名を省略すると，オブジェクト・モジュール・ファイル名は，'入力ファイル名．REL' となります。

④ －Oと－NOの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例1． オブジェクト・モジュール・ファイル（SAMPLE.REL）を出力します。

```
A>ra78k1 -c138 78k1main.asm -osample.rel

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

例2. －NOと－Oの両オプションを指定します。

```
A>ra78k1 -c138 78k1main.asm -no -n

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

－NOオプションは無効となり，－Oオプションが有効となります。

保守／廃止

(3) オブジェクト・モジュール・ファイル強制出力指定　(－J/－NJ)

記述形式　：－J
　　：－NJ
省略時解釈：－NJ

【機　　能】

① －Jオプションは,フェイタル・エラーでもオブジェクト・モジュール・ファイルを出力するように指定します。

② －NJオプションは，－Jオプションを無効にします。

【用　　途】

通常フェイタル・エラーがある場合には，オブジェクト・モジュール・ファイルは出力されません。したがって，フェイタル・エラーがあるのを承知でプログラムを実行させたい場合には，－Jオプションを指定しオブジェクト・モジュール・ファイルを出力します。

【説　　明】

① －Jオプションを指定すると,フェイタル・エラーがある場合でも,オブジェクト・モジュール・ファイルが出力されます。

② －Jと－NJの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例　フェイタル・エラーの場合でもオブジェクト・モジュール・ファイルを出力するよう指定します。

```
A>ra78k1 -c138 78k1main.asm -j

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(4) ディバグ情報出力指定　(－G/－NG)

どのようなディバグを行うかによって，オブジェクト中に生成する情報を制御するオプションです。

記述形式　：－G
　　：－NG
省略時解釈：－G

【機　能】

① －Gオプションは，オブジェクト・モジュール・ファイル中に，ディバグ情報（ローカル・シンボル情報）を付加するよう指示します。

② －NGオプションは，－Gオプションを無効にします。

【用　途】

① －Gオプションは，ローカル・シンボルも含めシンボリック・ディバグを行うときに使用します。

② －NGオプションは，次の3種類の場合に使用します。

1. グローバル・シンボルのみのシンボリック・ディバグ
2. シンボル無しでのディバグ
3. オブジェクトのみを必要とするとき（PROMによる評価時など）

【説　明】

－Gと－NGの両オプションが同時に指定された場合は，あとに指定した方を有効とします。

【注　　意】

ソース・モジュールの先頭で，－ G，－ NG オプションと同機能の制御命令が記述できます。次に記述形式を示します。

▲$▲ DEBUG	
▲$▲ DG	；短縮形
▲$▲ NODEBUG	
▲$▲ NODG	；短縮形

制御命令については，言語編の“**第4章　制御命令**”をお読みください。

【使 用 例】

例　オブジェクト・モジュール・ファイルにディバグ情報を付加します。

```
A>ra78k1 -c138 78k1main.asm -g

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(5) インクルード・ファイル読み込みパス指定　(－I)

記述形式　：－I　パス名　[, パス名] …　(複数指定可能)

省略時解釈：環境変数（INC78K1）により指定されたパス。

：指定されていない場合，ソース・ファイルのあるパス。

【機　　能】

ソース・モジュール中の '$include' で指定されたインクルード・ファイルを指定したパスから入力するよう指示します。

【用　　途】

インクルード・ファイルを，あるパスから検索したいときに指定します。

【説　　明】

① ', ' で区切ることにより，一度に複数のパス名を指定できます。

② ', ' の前後には空白を入れることはできません。

③ －Iに続いてパス名が複数指定されるか，あるいは－Iオプションが複数指定された場合，指定された順番に '$include' で指定したファイルを検索します。その後，省略時解釈と同じ順序で検索します。

④ －Iに続けてパス名以外のものを指定した場合やパス名を省略した場合は，アボート・エラーとなります。

⑤ －Iが９個以上指定された場合には，アボート・エラーとなります。

【使 用 例】

例　インクルード・ファイルをディレクトリ SAMPLE から読み込みます。

```
A>ra78k1 -c138 78k1main.asm -ib:¥sample

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(6) アセンブル・リスト・ファイル出力指定　(－P/－NP)

記述形式　：－P［出力ファイル名］

　　－NP

省略時解釈：－P　入力ファイル名．PRN

【機　　能】

① －Pオプションは，アセンブル・リスト・ファイルの出力を指定します。
　また，その出力先や出力ファイル名を指定します。

② －NPオプションは，－Pオプションを無効にします。

【用　　途】

① アセンブル・リスト・ファイルの出力先や出力ファイル名を変更したいときに，－Pオプションを指定します。

② オブジェクト・モジュール・ファイルの出力のみが目的でアセンブルする場合などに，－NPオプションを指定します。これによりアセンブル時間が短縮されます。

【説　　明】

① ファイル名としてディスク型ファイル名とデバイス型ファイル名を指定できます。指定できるデバイス型ファイル名は，CON，PRN，NULおよびAUXです。CLOCKを指定した場合，アボート・エラーとなります。

② －Pオプションを指定する際に出力ファイル名を省略するとアセンブル・リスト・ファイル名は，‘入力ファイル名．PRN’になります。

③ －Pオプションを指定する際にドライブ名を省略すると，カレント・ドライブにアセンブル・リスト・ファイルが出力されます。

④ －Pと－NPの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例 1.　アセンブル・リスト・ファイル（SAMPLE.PRN）を作成します。

```
A>ra78k1 -c138 78k1main.asm -psample.prn

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

例2.　アセンブル・リスト・ファイルをプリンタに出力します。

```
A>ra78k1 -c138 78k1main.asm -pprn

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(7) アセンブル・リスト・ファイル情報指定

(－ KA/－ NKA, － KS/－ NKS, － KX/－ NKX)

(a) － KA/－ NKA

記述形式　：－ KA

　　：－ NKA

省略時解釈：－ KA

【機　　能】

① －KAオプションは,アセンブル・リスト・ファイル中にアセンブル・リストを出力します。

② －NKAオプションは,－KAオプションを無効にします。

【用　　途】

アセンブル・リストを出力したいときに－KAオプションを指定します。

【説　　明】

① －KAと－NKAの両オプションを同時に指定した場合は,あとで指定した方を優先します。

② －NKA,－NKSおよび－NKXオプションを,すべて指定した場合は,アセンブル・リスト・ファイルは出力されません。

【使 用 例】

例 1. アセンブル・リストを出力します。

```
A>ra78k1 -c138 78k1main.asm -ka -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

例 2. 78K1MAIN.PRN を参照します。

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx                    Date:xx xxx xxxx Page:   1


Command:  -c138 78k1main.asm -ka -lw80
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

     Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                    $       PROCESSOR(138)
    2     2
    3     3                            NAME    SAMPM
    4     4                    ;*********************************************
                               *
    5     5                    ;*
                               *
    6     6                    ;*      HEX -> ASCII Conversion Program
                               *
    7     7                    ;*
                               *
    8     8                    ;*              main-routine
                               *
    9     9                    ;*
                               *
   10    10                    ;*********************************************
                               *
   11    11
   12    12                            PUBLIC  MAIN,START
   13    13                            EXTRN   CONVAH
   14    14
   15    15 ----                       DSEG
   16    16 0000               STASC:  DS      2
   17    17
   18    18 ----                       CSEG
   19    19 ----                       ORG     0H
   20    20 0000 R0000         MAIN:   DW      START
   21    21
   22    22 ----                       CSEG
   23    23 0000  0BFCE0FE     START:  MOVW    SP,#0FEE0H
   24    24 0004  2BC400               MOV     MM,#00
   25    25 0007  09C0FF00             MOV     STBC,#00
   26    26
   27    27 000B  BC00                 MOV     E,#LOW HDTSA            ;set hex
                                2-code data in HL registor
   28    28
   29    29 000D R280000               CALL    !CONVAH                 ;convert
                                ASCII <- HEX
   30    30                                                            ;output
                               BC-register <- ASCII code

                                  :

Target chip:uPD78138
Assembly complete,     0 error(s) and     0 warning(s) found. (    0)
```

保守／廃止

(b) －KS/－NKS

記述形式　：－KS

　　：－NKS

省略時解釈：－NKS

【機　　能】

① －KSオプションは,アセンブル・リストに続いてシンボル・リストをアセンブル・リスト・ファイル中に出力します。

② －NKSオプションは，－KSオプションを無効にします。

【用　　途】

シンボル・リストを出力したいときに，－KSオプションを指定します。

【説　　明】

① －KSと－NKSの両オプションを同時に指定した場合は，あとで指定した方を優先します。

② －KSと－KXを同時に指定した場合，－KSを無視します。

③ －NKA，－NKSおよび－NKXオプションが，すべて指定された場合は，アセンブル・リスト・ファイルは出力されません。

【使 用 例】

例1. シンボル・リストを出力します。

```
A>ra78k1 -c138 78k1main.asm -ks -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

例 2. 78K1MAIN.PRN を参照します(アセンブル・リストに続いてシンボル・リストが出力されています)。

```
A>type 78k1main.prn

                                        :

78K/I Series Assembler Vx.xx                        Date:xx xxx xxxx Page:   3

         Symbol Table List

 VALUE  ATTR  RTYP NAME          VALUE  ATTR  RTYP NAME

        CSEG       ?ASEG1              CSEG       ?CSEG
        DSEG       ?DSEG         ----H        EXT  CONVAH
        DSEG       DATA          FE00H  ADDR       HDTSA
   0H   ADDR  PUB  MAIN                 MOD        SAMPM
   0H   ADDR  PUB  START             0H  ADDR       STASC

Target chip:uPD78138
Assembly complete,     0 error(s) and     0 warning(s) found. (    0)
```

保守／廃止

(c) － KX/－ NKX

記述形式　：－ KX
　　：－ NKX
省略時解釈：－ NKX

【機　　能】

① －KXオプションは,アセンブル・リストに続いてクロスレファレンス・リストをアセンブル・リスト・ファイル中に出力します。

② －NKXオプションは，－KXオプションを無効にします。

【用　　途】

ソース・モジュール・ファイルで定義された各シンボルが，ソース・モジュール中のどこでどれだけ参照されているか，また，アセンブル・リストの何行目の記述で，そのシンボルを参照しているのかなどの情報を知りたいときにクロスレファレンス・リストを出力します。

【説　　明】

① －KXと－NKXの両オプションを同時に指定した場合は,あとで指定した方を優先します。

② －KSと －KXを同時に指定した場合，－KSを無視します。

③ －NKA，－NKSおよび－NKXを同時に指定した場合，アセンブル・リスト・ファイルは出力されません。

【注　　意】

ソース・モジュールの先頭で，－KX/－NKXオプションと同機能の制御命令を記述できます。その記述形式を次に示します。

```
▲$▲ XREF
▲$▲ XR                    ; 短縮形
▲$▲ NOXREF
▲$▲ NOXR                  ; 短縮形
```

制御命令については，言語編の“**第４章　制御命令**”をご覧ください。

【使 用 例】

例 1. クロスレファレンス・リストを出力します。

```
A>ra78k1 -c138 78k1main.asm -kx -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```


例 2. 78K1MAIN.PRN を参照します(アセンブル・リストに続いてクロスレファレンス・リストが出力されます)。

```
A>type 78k1main.prn

                                        ⋮

78K/I Series Assembler Vx.xx                          Date:xx xxx xxxx Page:   3

          Cross-Reference List

NAME      VALUE  R ATTR  RTYP SEGNAME  XREFS

?ASEG1              CSEG        ?ASEG1      19#
?CSEG               CSEG        ?CSEG       18#     22#
?DSEG               DSEG        ?DSEG       15#
CONVAH    ----H   E         EXT             13@     29
DATA                DSEG        DATA        39#
HDTSA     FE00H     ADDR        DATA        27      40#
MAIN        0H      ADDR  PUB   ?ASEG1      12@     20#
SAMPM               MOD                      3#
START       0H    R ADDR  PUB   ?CSEG       12@     20      23#
STASC       0H    R ADDR        ?DSEG       16#     31

Target chip:uPD78138
Assembly complete,      0 error(s) and      0 warning(s) found. (    0)
```

(8) アセンブル・リスト・ファイル形式指定

(－LW, －LL, －LH, －LT, －LF/－NLF)

(a) －LW

記述形式　：－LW［文字数］

省略時解釈：－LW132（ディスプレイ出力の場合は80文字とします）

【機　　能】

－LWオプションは，リスト・ファイルの1行の文字数を指定します。

【用　　途】

各種リスト・ファイルの1行の文字数を変更したいとき，－LWオプションを指定します。

【説　　明】

① －LWオプションで指定できる文字数の範囲を以下に示します（ディスプレイ出力の場合は80文字まで）。

72　≦　1行に印字する文字数　≦　250

範囲外の数値や数値以外を指定した場合は，アボート・エラーとなります。

② 文字数が省略された場合は，132を指定したものとみなします。

ただし，アセンブル・リスト・ファイルの出力先がディスプレイの場合は80とします。

③ 指定する文字数にはターミネータ（CR, LF）は含みません。

④ －NPオプションを指定した場合，－LWオプションは無効となります。

【使 用 例】

例1. －LWオプションを省略し，アセンブル・リストをプリンタに出力します。

```
A>ra78k1 -c138 78k1main.asm -pprn

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

アセンブル・リストを参照します。

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx                    Date:xx xxx xxxx Page:   1


Command:  -c138 78k1main.asm -pprn
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list


 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                     $       PROCESSOR(138)
    2     2
    3     3                             NAME    SAMPM
    4     4                     ;*************************************************
    5     5                     ;*                                               *
    6     6                     ;*      HEX -> ASCII Conversion Program          *
    7     7                     ;*                                               *
    8     8                     ;*            main-routine                       *
    9     9                     ;*                                               *
   10    10                     ;*************************************************
   11    11
   12    12                             PUBLIC  MAIN,START
   13    13                             EXTRN   CONVAH
   14    14
   15    15 ----                        DSEG
   16    16 0000                STASC:  DS      2
   17    17
   18    18 ----                        CSEG
   19    19 ----                        ORG     0H
   20    20 0000 R0000          MAIN:   DW      START
   21    21
   22    22 ----                        CSEG
   23    23 0000  0BFCE0FE      START:  MOVW    SP,#0FEE0H
   24    24 0004  2BC400                MOV     MM,#00
   25    25 0007  09C0FF00              MOV     STBC,#00
   26    26
   27    27 000B  BC00                  MOV     E,#LOW HDTSA        ;set hex 2-code data in HL registor
   28    28
   29    29 000D R280000                CALL    !CONVAH             ;convert ASCII <- HEX
   30    30                                                         ;output BC-register <- ASCII code
   31    31 0010 R640000                MOVW    DE,#STASC           ;set DE <- store ASCII code table
   32    32 0013  D3                    MOV     A,B
   33    33 0014  50                    MOV     [DE+],A
   34    34 0015  D2                    MOV     A,C
   35    35 0016  50                    MOV     [DE+],A
                                                 :
```

例 2. アセンブル・リスト・ファイルの１行の文字数を 80 文字に指定します。

```
A>ra78k1 -c138 78k1main.asm -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

《アセンブル・リストを参照します》

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx                          Date:xx xxx xxxx Page:   1


Command:  -c138 78k1main.asm -ka -lw80
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                    $       PROCESSOR(138)
    2     2
    3     3                            NAME    SAMPM
    4     4                    ;*********************************************
                               *
    5     5                    ;*
                               *
    6     6                    ;*      HEX -> ASCII Conversion Program
                               *
    7     7                    ;*
                               *
    8     8                    ;*              main-routine
                               *
    9     9                    ;*
                               *
   10    10                    ;*********************************************
                               *
   11    11
   12    12                            PUBLIC  MAIN,START
   13    13                            EXTRN   CONVAH
   14    14
   15    15 ----                       DSEG
   16    16 0000              STASC:   DS      2
   17    17
   18    18 ----                       CSEG
   19    19 ----                       ORG     0H
   20    20 0000 R0000        MAIN:    DW      START
   21    21
                                  :
```

保守／廃止

(b) －LL

記述形式　：－LL［行数］

省略時解釈：－LL66（ディスプレイ出力の場合は改頁しません）

【機　　能】

－LLオプションは，アセンブル・リスト・ファイルの1頁の行数を指定します。

【用　　途】

アセンブル・リスト・ファイルの1頁の行数を変更したいときに，－LLオプションで指定します。

【説　　明】

① －LLオプションで指定できる行数の範囲を以下に示します。

20　≦　1頁に印字する行数　≦　32767

範囲外の数値や数値以外のものが指定された場合には，アボート・エラーとなります。

② 行数を省略した場合，66を指定したものとみなします。

③ 行数の0を指定した場合は改頁しません。

④ －NPオプションを指定した場合，－LLオプションは無効となります。

【使 用 例】

例1. アセンブル・リスト・ファイルの1頁の行数を20行に指定します。

```
A>ra78k1 -c138 78k1main.asm -l120 -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

例 2. 78K1MAIN.PRN を参照します。

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx                          Date:xx xxx xxxx Page:   1


Command:  -c138 78k1main.asm -l120 -lw80
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list
--------------------------------------------------------------------------------

78K/I Series Assembler Vx.xx                          Date:xx xxx xxxx Page:   2


 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                    $       PROCESSOR(138)
    2     2
    3     3                            NAME    SAMPM
    4     4                    ;*************************************************
    5     5                    ;*                                               *
    6     6                    ;*      HEX -> ASCII Conversion Program          *
--------------------------------------------------------------------------------

78K/I Series Assembler Vx.xx                          Date:xx xxx xxxx Page:   3


 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

                              *
    7     7                    ;*                                               *
    8     8                    ;*              main-routine                     *
    9     9                    ;*                                               *
   10    10                    ;*************************************************

                                        :
```

保守／廃止

(c) －LH

記述形式　：－LH 文字列

省略時解釈：なし

【機　　能】

－LH オプションは，アセンブル・リスト・ファイルのヘッダのタイトル欄に印字する文字列を指定します。

【用　　途】

① アセンブル・リスト・ファイルの内容を端的に表すようなタイトルを表示したいときに，－LH オプションを指定します。

② タイトルを各頁に印字することで，アセンブル・リスト・ファイルの内容が一目でわかります。

【説　　明】

① 指定可能な文字列は，60 文字以内です。ただし，文字列内に空白は記述できません。

② 61 文字以上記述された場合には，先頭の 60 文字を有効とし，エラーは出力されません。なお，漢字，ひらがなは，1 文字を 2 文字として計算します。

1 行の最大文字数が 119 以下の場合，タイトルとしての文字列の有効長は次のとおりです。

有効長＝（1 行の最大文字数）－ 60

③ 文字列が指定されなかった場合は，アボート・エラーとなります。

④ －NP オプションを指定した場合，－LH オプションは無効となります。

⑤ －LH オプションを省略した場合，アセンブル・リスト・ファイルのタイトル欄は，空白となります。

⑥ 記述可能な文字セットを次に示します。

保守／廃止

表４－５　タイトルとして記述可能な文字

文　　字	コマンド行	パラメータ・ファイル中
＊？＞＜｜	” ”でくくることで記述可能注	記述可能 ” ”でくくってもコマンド行と同じように解釈します。
;	” ”でくくることで記述可能	記述不可（コメントとみなされます。）
#	記述可能	記述不可（コメントとみなされます。）
” （ダブル・クォーテーション）	有効文字としては記述不可	有効文字としては記述不可
00H	記述不可	記述可能 ただし文字列が途切れたとみなされます。
03H，06H，08H，0DH 0EH，10H，15H，17H 18H，1BH，7FH	記述不可	記述可能 ただしアセンブル・リスト・ファイル中では’!’で表示 （単独の 0DH はリストには出力されません。）
01H，02H，04H，05H， 07H，0BH，0CH，0FH， 11H，12H，13H，14H， 16H，19H，1CH，1DH， 1EH，1FH	記述可能 ただしアセンブル・リフト・ファイル中では’!’で表示	記述可能 ただしアセンブル・リスト・ファイル中では’!’で表示
1AH	記述可能 ただしアセンブル・リスト・ファイル中では’!’で表示	記述不可 （ファイルの終わり）
英字	大/小文字がそのまま入力されます。	大/小文字がそのまま入力されます。
その他	記述可能	記述可能

注　起動行上の＊は，ワイルドカード展開の対象とならない場合は” ”でくくらなくても記述可能です。

【注　　意】

ソース・モジュールの先頭で，－LH オプションと同機能の制御命令を記述できます。次に記述形式を示します。

```
▲$▲ TITLE ▲（▲ ’文字列’ ▲）
▲$▲ TT ▲（▲ ’文字列’ ▲）                    ；短縮形
```

制御命令については，言語編の“**第４章　制御命令**”をお読みください。

【使 用 例】

例　アセンブル・リスト・ファイルのヘッダにタイトルを印字します。

```
A>ra78k1 -c138 78k1main.asm -1hRA78K1-MAINROUTINE

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```


《78k1main.prn を参照します》

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx   RA78K1-MAINROUTINE            Date:xx xxx xxxx Page:   1
                                  └ ｺﾏﾝﾄﾞ・ﾗｲﾝで指定したﾀｲﾄﾙ

Command:  -c138 78k1main.asm -1hRA78K1-MAINROUTINE
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                    $       PROCESSOR(138)
    2     2
    3     3                            NAME    SAMPM
    4     4                    ;*************************************************
    5     5                    ;*                                               *
    6     6                    ;*      HEX -> ASCII Conversion Program          *
    7     7                    ;*                                               *
    8     8                    ;*              main-routine                     *
    9     9                    ;*                                               *
   10    10                    ;*************************************************
   11    11
   12    12                            PUBLIC  MAIN,START
   13    13                            EXTRN   CONVAH
   14    14
   15    15 ----                       DSEG
   16    16 0000               STASC:  DS      2
   17    17
   18    18 ----                       CSEG

                                        ⋮
```

保守／廃止

(d) － LT

記述形式　：－ LT 文字数

省略時解釈：－ LT8

【機　　能】

－ LT オプションは，ソース・モジュール中の HT (Horizontal Tabulation) コードを，各種リスト上でいくつかのブランク（空白）に置き換えて出力する（タビュレーション処理）ための基本となる文字数を指定します。

【用　　途】

－ LW オプションで，各種リストの 1 行の文字数を少なく指定した場合に，HT コードによるブランクを少なくし，文字数を節約するために，－ LT オプションを指定します。

【説　　明】

① －LT オプションで指定できる文字数の範囲は次のとおりです。

0 ≦ 指定できる文字数 ≦ 8

範囲外の数値や数値以外のものを指定した場合は，アボート・エラーとなります。

② －LT0 を指定した場合，タビュレーション処理は行わず，タブ・コードを出力します。

③ －NP オプションを指定した場合，－ LT オプションは無効となります。

【使 用 例】

例 1. －LT オプションを省略した場合の 78K1MAIN.PRN を参照します。

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx                                Date:xx xxx xxxx Page:   1


Command:  78K1MAIN.ASM
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                      $       PROCESSOR(138)
    2     2
    3     3                              NAME    SAMPM
    4     4                      ;*************************************************
    5     5                      ;*                                               *
    6     6                      ;*      HEX -> ASCII Conversion Program          *
    7     7                      ;*                                               *
    8     8                      ;*              main-routine                     *
    9     9                      ;*                                               *
   10    10                      ;*************************************************
   11    11
   12    12                              PUBLIC  MAIN,START
   13    13                              EXTRN   CONVAH
   14    14
   15    15 ----                         DSEG
   16    16 0000                 STASC:  DS      2
   17    17
   18    18 ----                         CSEG
   19    19 ----                         ORG     0H
   20    20 0000 R0000           MAIN:   DW      START
   21    21
   22    22 ----                         CSEG
   23    23 0000  0BFCE0FE       START:  MOVW    SP,#0FEE0H
   24    24 0004  2BC400                 MOV     MM,#00
   25    25 0007  09C0FF00               MOV     STBC,#00
   26    26
   27    27 000B  BC00                   MOV     E,#LOW HDTSA        ;set hex 2-code data in HL registor
   28    28
   29    29 000D R280000                 CALL    !CONVAH             ;convert ASCII <- HEX
                                                 :
```

例 2. HT コードによるブランクを 1 に指定します。

```
A>ra78k1 -c138 78k1main.asm -lt1

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

●78K1MAIN.PRN を参照します。

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx                                Date:xx xxx xxxx Page:   1


Command:  -c138 78k1main.asm -lt1
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                     $ PROCESSOR(138)
    2     2
    3     3                      NAME SAMPM
    4     4                     ;*************************************************
    5     5                     ;*                                               *
    6     6                     ;*      HEX -> ASCII Conversion Program          *
    7     7                     ;*                                               *
    8     8                     ;*              main-routine                     *
    9     9                     ;*                                               *
   10    10                     ;*************************************************
   11    11
   12    12                      PUBLIC MAIN,START
   13    13                      EXTRN CONVAH
   14    14
   15    15 ----                 DSEG
   16    16 0000                STASC: DS 2
   17    17
   18    18 ----                 CSEG
   19    19 ----                 ORG 0H
   20    20 0000 R0000          MAIN: DW START
   21    21
   22    22 ----                 CSEG
   23    23 0000  0BFCE0FE      START: MOVW SP,#0FEE0H
   24    24 0004  2BC400         MOV MM,#00
   25    25 0007  09C0FF00       MOV STBC,#00
   26    26
   27    27 000B  BC00           MOV E,#LOW HDTSA                ;set hex 2-code data in HL registor
   28    28
   29    29 000D R280000         CALL !CONVAH                    ;convert ASCII <- HEX
```

⋮

備考　HT コードによるブランクは 1 つです。

保守／廃止

(e) －LF/－NLF

記述形式　：－LF

　　：－NLF

省略時形式：－NLF

【機　　能】

① －LFオプションは，アセンブル・リスト・ファイルの最後に改頁コード（FF）の付加を指定します。

② －NLFオプションは，－LFオプションを無効にします。

【用　　途】

アセンブル・リスト・ファイルの内容を印字したあとで改頁しておきたい場合には，－LFオプションを指定して改頁コードを付加します。

【説　　明】

① －NPオプションを指定した場合，－LFオプションは無効となります。

② －LFと－NLFの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例　アセンブル・リスト・ファイルの最後に改頁コードを付加します。

```
A>ra78k1 -c138 78k1main.asm -pprn -lf

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(9) エラー・リスト・ファイル出力指定　(－E/－NE)

記述形式　：－E［出力ファイル名］
　　：－NE
省略時形式：－NE

【機　　能】

① －Eオプションは，エラー・リスト・ファイルの出力を指定します。また，その出力先や出力ファイル名を指定します。

② －NEオプションは，－Eオプションを無効にします。

【用　　途】

① エラー・メッセージをファイルに保存しておきたい場合，－E オプションを指定します。

② エラー・リスト・ファイルの出力先や出力ファイル名を変更したいときに，－Eオプションで指定します。

【説　　明】

① ファイル名としてディスク型ファイル名とデバイス型ファイル名を指定できます。
ただし，デバイス型ファイル名 CLOCK を指定した場合は，アボート・エラーとなります。

② －Eオプションを指定する際に，出力ファイル名を省略するとエラー・リスト・ファイル名は，‘入力ファイル名．ERA’となります。

③ －Eオプションを指定する際にドライブ名を省略すると，カレント・ドライブにエラー・リスト・ファイルが出力されます。

④ －Eと－NEの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例 1. エラー・リスト・ファイル（SAMPLE.ERA）を作成します。

```
A>ra78k1 -c138 78k1main.asm -esample.era

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
78K1MAIN.ASM(20) : F201 Syntax error
78K1MAIN.ASM(23) : F201 Syntax error
Pass2 Start
78K1MAIN.ASM(12) : F404 Public symbol is undefined 'MMAIN'
78K1MAIN.ASM(20) : F201 Syntax error
78K1MAIN.ASM(23) : F201 Syntax error

Assembly complete,      3 error(s) and      0 warning(s) found.
```

●エラー・リスト・ファイル（SAMPLE.ERA）を参照します。

```
A>type sample.era

Pass1 Start
78K1MAIN.ASM(20) : F201 Syntax error
78K1MAIN.ASM(23) : F201 Syntax error
Pass2 Start
78K1MAIN.ASM(12) : F404 Public symbol is undefined 'MMAIN'
78K1MAIN.ASM(20) : F201 Syntax error
78K1MAIN.ASM(23) : F201 Syntax error
```

保守／廃止

(10) パラメータ・ファイル指定　(－F)

記述形式　：－Fファイル名

省略時解釈：起動行上からのみオプションまたは入力ファイル名が入力可能。

【機　　能】

－Fオプションは，オプションあるいは入力ファイル名を指定のファイルから入力する指定をします。

【用　　途】

① コマンド行ではアセンブラの起動に必要な情報を指定しきれないときに，－Fオプションを指定します。

② アセンブルするたびに繰り返し同じようにオプションを指定する際には，それらをパラメータ・ファイルに記述しておき，－Fオプションで指定します。

【説　　明】

① 'ファイル名'として指定できるのは，ディスク型ファイル名のみです。
デバイス型ファイル名を指定するとアボート・エラーとなります。

② ファイル名を省略するとアボート・エラーとなります。

③ パラメータ・ファイルのネストは許されません。パラメータ・ファイル中で，－Fオプションを指定するとアボート・エラーとなります。

④ パラメータ・ファイル中に記述できる文字数の制限はありません。

⑤ 空白とタブおよび'↵'をオプションあるいは入力ファイル名の区切りとします。

⑥ パラメータ・ファイル中に記述したオプションあるいは入力ファイル名はコマンド行上のパラメータ・ファイル指定のあった位置に展開されます。

⑦ 展開されたオプションは，あとで指定したものを優先します。

⑧ ';'または'#'以降に記述された文字は'↵'またはEOFの前まですべてコメントと解釈します。

⑨ －Fオプションを複数指定するとアボート・エラーとなります。

保守／廃止

【使 用 例】

例　パラメータ・ファイルを使用してアセンブルします。

●パラメータ・ファイル（78K1MAIN.PRA）の内容

```
;parameter file
78k1main.asm -osample.rel -g
-psample.prn
```

●コマンド行には，次のように入力します。

```
A>ra78k1 -f78k1main.pra

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(11) テンポラリ・ファイル作成パス指定　(－T)

記述形式　：－T　パス

省略時解釈：環境変数 TMP により指定されたパスに作成します。
　　指定されていない場合は，カレント・パスに作成します。

【機　　能】

－Tオプションは，テンポラリ・ファイルを作成するパスを指定します。

【用　　途】

テンポラリ・ファイルの作成場所を指定できます。

【説　　明】

① パス名としてパス以外のものは指定できません。

② パス名は省略できません。

③ 以前に作成されたテンポラリ・ファイルが存在している場合でも，ファイル保護がされていなければ，上書きします。

④ 必要とするメモリ・サイズがある間は，テンポラリ・ファイルをメモリに展開します。メモリが足りなくなった時点で，メモリに展開していたテンポラリ・ファイルの内容をディスクに書き出します。

以降のテンポラリ・ファイルへのアクセスは，セーブしたディスク・ファイルに対して行います。

⑤ テンポラリ・ファイルは，アセンブル終了時に削除されます。また，キー入力（CTRL-C）によってアセンブルが中止されたときにも削除されます。

⑥ テンポラリ・ファイルの作成パスは，次の順番で決定されます。

a. －Tオプションで指定されたパス

b. 環境変数 TMP に設定されているパス（－Tオプション省略の場合）

c. カレント・パス（TMP が設定されていない場合）

なお，a または b を指定した場合，指定されたパスにテンポラリ・ファイルが作成できなければアボート・エラーとなります。

【使 用 例】

例　テンポラリ・ファイルをディレクトリ TMP に出力するよう指定します。

```
A>ra78k1 -c138 78k1main.asm -ttmp

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(12) ヘルプ指定　(－－)

記述形式　：－－

省略時形式：表示しない

【機　　能】

－－オプションは，ヘルプ・メッセージをディスプレイに表示します。

【用　　途】

ヘルプ・メッセージは，アセンブル・オプションとその説明の一覧です。アセンブラを実行するときに参照してください。

【説　　明】

① －－オプションを指定すると他のアセンブラ・オプションは，すべて無効となります。

② ヘルプ・メッセージの続きをお読みになる場合は，リターン・キーを入力してください。表示を途中で終了する場合は，リターン・キー以外の文字を入力したあとで，リターン・キーを入力してください。

【使 用 例】

例　－－オプションを指定すると，ヘルプ・メッセージがディスプレイに出力されます。

```
A>ra78k1 --

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

usage : RA78K1 [option[...]] input-file [option[...]]
The option is as follows ([] means omissible).
-Cx           :Select target chip. ( x = 112,138,etc. ) *Must be specified.
-O[file]/-NO :Create the object module file [with the specified name] / Not.
-E[file]/-NE :Create the error list file [with the specified name] / Not.
-P[file]/-NP :Create the print file [with the specified name] / Not.
-KA/-NKA      :Output the assemble list to print file / Not.
-KS/-NKS      :Output the symbol table list to print file / Not.
-KX/-NKX      :Output the cross reference list to print file / Not.
-LW[width]    :Specify print file columns per line.
-LL[length]   :Specify print file lines per page.
-LF/-NLF      :Add Form Feed at end of print file / Not.
-LT[n]        :Expand TAB character for print file(n=1 to 8)/ Not expand(n=0).
-LHstring     :Print list header with the specified string.
                          ⋮
```

保守／廃止

第5章　リ ン カ

リンカは，78K/I のアセンブラが出力したいくつかのオブジェクト・モジュール・ファイルを入力し，配置アドレスを決定して1つにまとめたロード・モジュール・ファイルを出力します。

さらに，リンク・リスト・ファイルやエラー・リスト・ファイルなどのリスト・ファイルを出力します。

リンク・エラーがある場合は，エラー・メッセージをエラー・リスト・ファイルに出力し，エラーの原因を明示します。なお，エラーがある場合は，ロード・モジュール・ファイルを出力しません。

保守／廃止

5.1 リンカの入出力ファイル

リンカの入出力ファイルを以下に示します。

表5-1 リンカの入出力ファイル

種類	ファイル名	説明	ディフォールト・ファイル・タイプ
入力ファイル	オブジェクト・モジュール・ファイル	○機械語情報と機械語の配置アドレスに関する再配置情報およびシンボル情報を含んだバイナリ・ファイルです。 ○アセンブラの出力したファイルです。	.REL
	ライブラリ・ファイル	○複数のオブジェクト・モジュール・ファイルが登録されたファイルです。 ○ライブラリアンの出力したファイルです。	.LIB
	ディレクティブ・ファイル	○リンクに対するリンク指示を記述したファイルです。 ○ユーザ作成ファイルです。	.DR
	パラメータ・ファイル	○実行プログラムのパラメータを内容とするファイルです。 ○ユーザ作成ファイルです。	.PLK
出力ファイル	ロード・モジュール・ファイル	○リンク結果の全情報を持つバイナリ・イメージのファイルです。オブジェクト・コンバータの入力ファイルとなります。	.LNK
	リンク・リスト・ファイル	○リンク結果を表示するリスト・ファイルです。	.MAP
	エラー・リスト・ファイル	○リンク時のエラー情報を持つファイルです。	.ELK
入出力ファイル	テンポラリ・ファイル	○リンクのためにリンカが自動生成するファイルです。アセンブル終了時には消去されます。	LK××××××. $$n (n=1~3)

保守／廃止

5.2 リンカの機能

リンカの機能を次に示します。

(1) 入力モジュールの決定

入力としてライブラリ・ファイルが指定されていると，入力オブジェクト・モジュール・ファイルが参照しているモジュールをライブラリから探し出して，入力モジュールとして扱います。

(2) 入力セグメントの結合

リンカは，セグメントごとに配置するアドレスを決定し，管理します。

別々のオブジェクト・モジュール・ファイルにあっても，セグメント名が同じであれば，そのセグメントを1つとみなして結合します。

(3) 入力セグメントの配置アドレス決定

入力モジュールの配置アドレスをセグメントごとに決定します。ソース・モジュール・ファイル中で指定された配置アドレスに従って配置します。また，リンカの「リンク・ディレクティブ・ファイル」で配置アドレスを指定できます。

(4) オブジェクト・コードの修正

(3) で決定した配置アドレスに従ってオブジェクト・コードを修正します。

保守／廃止

5.3　メモリ空間とメモリ領域

メモリ空間とは，メモリ領域を定義するための空間です。また，メモリ領域とはセグメントを配置するためにメモリ空間中に定義した領域です。

メモリ空間：64K バイトごと

メモリ領域：1 つのメモリ空間をさらにいくつかの領域に分割する。

実装しているメモリのアドレスを宣言する。

セグメントの配置のグループ分け（外付け ROM など）

メモリ領域名	ディフォールトのアドレス	ディフォールトで配置されるセグメント
ROM	内蔵 ROM：ROM レスは RAM の前まで	CSEG
RAM	内蔵 RAM	DSEG，BSEG

備考　メモリ領域のディフォールトのアドレスを変更したい場合やプログラムで記述した各セグメントの配置を指定したい場合に，ディレクティブ・ファイルを使用します。

保守／廃止

5.4 リンク・ディレクティブ

リンク・ディレクティブ（以降ディレクティブとします）とは，リンカに対して入力ファイルや使用可能なメモリ領域，セグメントの配置など，リンク時の各種指示を行うための命令群です。

ディレクティブ・ファイルの役割

(1) 実装メモリのアドレスを宣言。
(2) メモリをいくつかの領域に分割。

例：CALLT 領域
内蔵 ROM
外付け ROM
SADDR
SADDR 以外の内蔵 RAM

(3) セグメントの配置をリンカで指定する。
各セグメントに対し，次の内容を指定します。

- アブソリュート・アドレス
- メモリ領域のみ指定

エディタなどでディレクティブ・ファイル（ディレクティブを記述したファイル）を作成し，リンカの起動時に，－D オプションを指定して作成したファイルを読み込みます。
リンカはファイルからディレクティブを読み込み，解釈しながらリンク処理を行います。
ディレクティブには，次の 2 種類があります。

表 5-2 ディレクティブの種類

項番	ディレクティブの種類	説　明
1	メモリ・ディレクティブ	○実装メモリのアドレスを宣言します。 ○メモリをいくつかの領域に分割して， メモリ領域を指定します。
2	セグメント配置 ディレクティブ	セグメントの配置を指定します。

保守／廃止

5.4.1 ディレクティブ・ファイル

ディレクティブ・ファイル中に記述するディレクティブの記述フォーマットを次に示します。

ディレクティブは，1つのディレクティブ・ファイル中に複数記述できます。

① メモリ・ディレクティブ

MEMORY メモリ領域名：(スタート・アドレス値，サイズ) [/メモリ空間名]

② セグメント配置ディレクティブ

MERGE セグメント名：[AT△ (△スタート・アドレス△)] [＝メモリ領域名指定]

[/メモリ空間名]

(1) 予 約 語

ディレクティブ・ファイル中での予約語を以下に示します。

MEMORY, MERGE, AT, SEQUENT, COMPLETE

ディレクティブ・ファイル中で予約語を他の意味（セグメント名やメモリ領域名など）に使用することはできません。

予約語の記述は，大文字でも小文字でもかまいません。ただし，大文字と小文字を混在させた記述はできません。

例 MEMORY

memory

Memory ：使用不可

(2) シンボル

セグメント名，メモリ領域名，メモリ空間名の記述では大文字と小文字を区別します。

ソース・モジュール・ファイル中でセグメント名を小文字で記述しても，アセンブル時に“－NCA”オプション（大文字と小文字を区別する）を指定しないとシンボルはすべて大文字として扱われます。この場合，ディレクティブ・ファイル中のセグメント名は大文字で指定してください。

(3) 数 値

各ディレクティブの項目のうち数値定数を記述する場合は，10進数または16進数を記述できます。記述方法はソース・プログラムと同じで，16進数の場合は最後に“H”を付けます。また，先頭がA～Fの場合は前に“0”を付けます。

例 23H, 0FC80H

保守／廃止

(4) コメント文

ディレクティブ・ファイル中に，‘;’または‘#’を記述した場合，そこから改行文字（LF）までは，コメントとして扱われます。なお，改行文字が現れる前にディレクティブ・ファイルが終了した場合は，終了までをコメントとして扱います。

例　下線部がコメントとなります。

```
; DIRECTIVE FILE FOR 78312
MEMORY MEM1 : (1000H, 1000H)    #SECOND MEMORY AREA
```

保守／廃止

5.4.2　メモリ・ディレクティブ

メモリ・ディレクティブは，メモリ領域（実装するメモリのアドレスと名前）を定義するディレクティブです。

定義したメモリ領域は，その名前（メモリ領域名）によってセグメント配置ディレクティブで参照できます。

メモリ領域は，ディフォールトで定義されているメモリ領域を含め100個まで定義できます。

【構　　文】

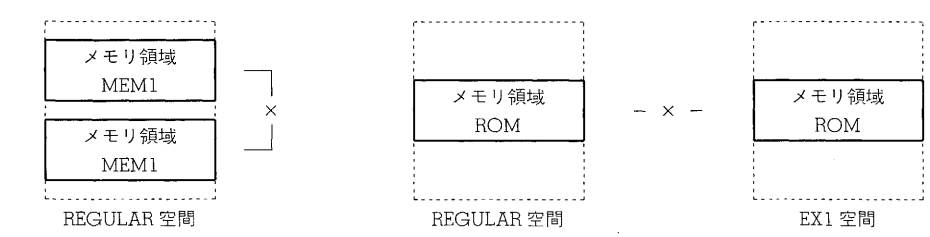
MEMORY△メモリ領域名▲：▲（▲スタート・アドレス▲，▲サイズ▲）[/▲メモリ空間名]

(1) メモリ領域名

定義するメモリ領域の名前を指定します。指定時の条件は以下のとおりです。

① メモリ領域名に使用できる文字は，A～Z，a～z，0～9，_，?，@です。ただし，0～9はメモリ領域名の先頭には使用できません。
② 大文字と小文字は別の文字として区別します。
③ 大文字と小文字は混在できます。
④ メモリ領域名の長さは，最大31文字です。32文字以上記述するとエラーとなります。
⑤ 各メモリ領域名は，全メモリ空間を通じて1つでなくてはなりません。異なるメモリ領域に同じメモリ領域名を付けることは，メモリ空間が同一である場合でも，異なる場合でも許されません。

図5-1　メモリ領域名



(2) スタート・アドレス

定義するメモリ領域の先頭アドレスを指定します。

●0H～FFFFHまでの数値定数を記述します。

保守／廃止

(3) サイズ

定義するメモリ領域のサイズを指定します。指定時の条件は以下のとおりです。

① 1以上の数値定数を記述します。

② リンカがディフォールトで定義しているメモリ領域のサイズを指定しなおす場合には，定義可能な範囲の制約があります。

次にディフォールトで定義されているメモリ領域のサイズと再定義可能な範囲を示します。

表5-3 ディフォールトのメモリ領域と再定義可能な範囲

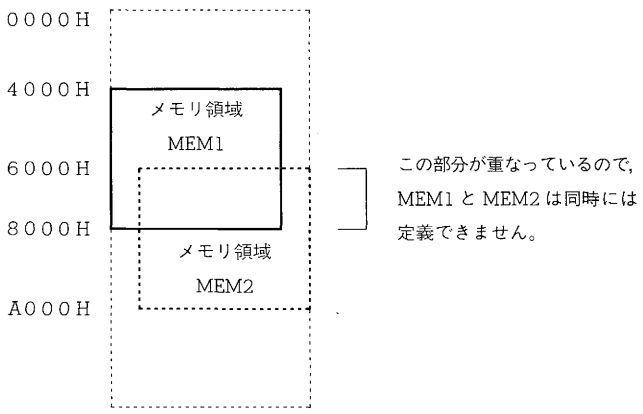
対象デバイス	ROM領域		RAM領域	
	ディフォールト	再定義可能範囲	ディフォールト	再定義可能範囲
μPD78134 μPD78134A	0000H-3FFFH	0000H-FD7FH	FD80H-FEFFH	4000H-FEFFH
μPD78136	0000H-5FFFH	0000H-FC7FH	FC80H-FEFFH	6000H-FEFFH
μPD78138 μPD78P138	0000H-7FFFH	0000H-FC7FH	FC80H-FEFFH	8000H-FEFFH
μPD78146	0000H-5FFFH	なし	FC80H-FEFFH	なし
μPD78148 μPD78P148	0000H-7FFFH		FC00H-FEFFH	

内蔵ROM領域があるデバイスは，その領域をディフォールトのROM領域としています。ディフォールトのRAM領域は，内蔵RAMの領域そのものです。

再定義可能な範囲は，ROM領域については0番地から内蔵RAMの直前までで，RAM領域については，内蔵ROM領域とsfr領域を除いた部分です。

1つのメモリ空間上において，同一のアドレス上に複数のメモリ領域は定義できません。

図5-2 不正なメモリ領域の定義



保守／廃止

(4) メモリ空間名

メモリ空間名は，メモリ空間を64Kバイトごとに分けた次の16個の名前で表されます。

REGULAR, EX1, EX2, EX3, EX4, EX5, EX6, EX7, EX8, EX9, EX10, EX11,
EX12, EX13, EX14, EX15

メモリ空間名は，メモリ領域をどのメモリ空間に割り付けるかを指定するときに使います。以下に指定時の条件を示します。

① メモリ空間名は，すべて大文字で記述します。
② メモリ空間名を省略した場合，REGULARを指定したものとみなされます。
③ '/' を記述したあとにメモリ空間名を省略した場合は，エラーとなります。

【機　能】

① メモリ領域名で指定した名前を持つメモリ領域を，指定したメモリ空間に定義します。
② 1つのメモリ・ディレクティブで1つのメモリ領域を定義できます。
③ メモリ・ディレクティブ自体は，複数の記述が可能です。このとき，指定した順番に複数回定義された場合は，エラーとなります。
④ ディフォールトのメモリ領域は，メモリ・ディレクティブで同一のメモリ領域を再定義しないかぎり有効です。メモリ・ディレクティブの記述を省略した場合，リンカが持つ各デバイスごとのディフォールトのメモリ領域のみを指定したものとします。

【使 用 例】

① ディフォールトのメモリ空間(REGULAR)のアドレス0Hから1FFHまでをメモリ領域ROMAとして定義します。

```
MEMORY  ROMA : (0, 200H)
```

② メモリ領域　RAMAとして定義します。

```
MEMORY  RAMA : (1F00H, 100H) /EX1
```

保守／廃止

5.4.3 セグメント配置ディレクティブ

セグメント配置ディレクティブは，指定したセグメントを指定したメモリ領域上か，特定番地に配置するディレクティブです。

【構　文】

MERGE△セグメント名▲：▲［AT▲（▲スタート・アドレス▲）］［▲＝▲メモリ領域名］
［▲/▲メモリ空間名］

(1) セグメント名

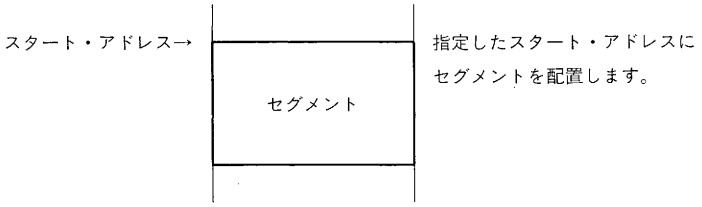
リンカに入力するオブジェクト・モジュール・ファイル中に含まれるセグメント名です。

① セグメント名として入力セグメント以外は指定できません。

(2) スタート・アドレス

セグメントを“スタート・アドレス”で指定した領域に配置します。

① 予約語 AT は，大文字または小文字のいずれか一方で記述しなければなりません。なお，大文字と小文字を混同してはなりません。
② スタート・アドレスには，数値定数を記述します。



注意 1. 指定したスタート・アドレスによって配置を行うと，セグメントが配置されるメモリ領域の範囲を越えてしまう場合はエラーとなります。

2. セグメント疑似命令の AT 指定または ORG 疑似命令によって配置アドレスを指定したセグメントに対して，リンク・ディレクティブでスタート・アドレスは指定できません。

保守／廃止

(3) メモリ空間名

メモリ空間名は，セグメントを配置するメモリ空間を指定します。

① メモリ空間名として指定できるのは，次の16種類のうちのいずれかです。

REGULAR, EX1, EX2, EX3, EX4, EX5, EX6, EX7, EX8, EX9, EX10, EX11, EX12, EX13, EX14, EX15

② メモリ空間名は，すべて大文字で記述します。

③ メモリ空間名を省略した場合，REGULARを指定したものとみなします。

次にセグメントの配置先を示します。

表5-4 メモリ領域名指定とメモリ空間の組み合わせによるセグメントの配置

メモリ領域名指定	メモリ空間	セグメントの配置先
×	×	REGULAR空間中のディフォールト状態のとき配置されるメモリ領域
×	○	指定されたメモリ空間中の任意のメモリ領域
メモリ領域名	×	REGULAR空間の指定されたメモリ領域
メモリ領域名	○	指定されたメモリ空間の指定されたメモリ領域

この表では，セグメントの配置の対象となるメモリ領域を定義するということを中心として述べています。なお，実際の配置アドレス決定時には，「AT（スタート・アドレス）」が指定されていれば，そのアドレスからセグメントを配置します。

たとえば，再配置属性が 'CSEG FIXED' であるセグメントに，メモリ名 'EX1' が指定された場合，セグメントが800H～FFFHの中に納まるように配置します。

【注　意】

① セグメント配置ディレクティブが指定されなかった入力セグメントは，アセンブル時にセグメント疑似命令で指定した再配置属性に従って配置アドレスを決定します。

② セグメント名として指定したセグメントが存在しない場合はエラーです。

③ 同一のセグメントに対して，セグメント配置ディレクティブを複数回指定した場合エラーとなります。

保守／廃止

【使 用 例】

セグメント・タイプ，再配置属性が 'CSEG FIXED' であるセグメント SEG1 に対して，アドレスを割り付けます。

① 入力セグメント SEG1 を ROM 領域中の 2000H に割り付けます。

```
MERGE SEG1 : AT (2000H)
```

② 入力セグメント SEG1 をメモリ領域 MEM1 中に割り付けます。

```
MERGE SEG1 : =MEM1
```

③ 入力セグメント SEG1 をメモリ領域 MEM1 中の 2000H に割り付けます。

```
MERGE SEG1 : AT (2000H) =MEM1
```

保守／廃止

5.5 リンカの起動方法

5.5.1 リンカの起動

リンカの起動には，次の２つの方法があります。

(1) コマンド行での起動

X＞［パス名］　lk78k1　［△オプション］…△オブジェクト・モジュール・ファイル名
①　②　③　④　⑤

［△オプション］…［△］
④

① OSのプロンプト。

② リンカのコマンド・ファイルを格納しているパス名。

③ リンカのコマンド・ファイル名。

MS-DOS ver2.11の場合，すべてのプログラム・ファイルはカレント・ディレクトリになければなりません。

④ リンカに対して動作の詳細を指示します。

複数のリンカ・オプションを指定する場合は，それぞれのリンカ・オプション間を空白で区切ってください。リンカ・オプションの詳細については，“**5.6 リンカ・オプション**”をお読みください。

⑤ リンクするオブジェクト・モジュール・ファイル名です。

複数のファイル名を指定する場合は，それぞれのファイル名間を空白で区切ってください。入力ファイルは，－Bオプションでの指定ファイルと合わせて最大128個入力できます。

例 A＞lk78k1 78k1main.rel 78k1sub.rel -o78k1.lnk -g

保守／廃止

(2) パラメータ・ファイルによる起動

パラメータ・ファイルは，起動に必要な情報がコマンド行に指定しきれない場合やリンクするたびに同じオプションを繰り返し指定するような場合に使用します。

パラメータ・ファイルを使用する場合には，コマンド行にパラメータ・ファイル指定オプション（－F）を指定します。

パラメータ・ファイルによる起動方法は，次のようになります。

X>lk78k1 [△オブジェクト・モジュール・ファイル] △-fパラメータ・ファイル名
①　②

① パラメータ・ファイル指定オプション

② リンカの起動に必要な情報を含んだファイル。

備考 パラメータ・ファイルは，エディタなどで作成します。

パラメータ・ファイル内での記述規則を次に示します。

[[[△] オプション [△オプション] … [△] △]] …

① コマンド行でオブジェクト・モジュール・ファイル名を省略した場合は，パラメータ・ファイル内でオブジェクト・モジュール・ファイル名を指定します。

② オブジェクト・モジュール・ファイル名は，オプションの後に記述することも可能です。

③ パラメータ・ファイルには，コマンド行で指定すべきすべてのリンク・オプション，出力ファイル名を記述します。

例 パラメータ・ファイル（78K1. PLK）をエディタなどで作成します。

78K1. PLKの内容

```
; parameter file
78k1main. rel 78k1sub. rel -o78k1. lnk -p78k1. map -e
-ta：￥tmp -g
```

パラメータ・ファイル78K1. PLKを使用してリンカを起動します。

```
A>lk78k1 -f78k1. plk
```

保守／廃止

5.5.2 実行開始，終了メッセージ

(1) 実行開始メッセージ

リンカが起動すると，ディスプレイに次の実行開始メッセージを表示します。

```
78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989
```

(2) 実行終了メッセージ

リンクの結果，リンク・エラーが検出されなかった場合，リンカは次のメッセージをディスプレイに出力して制約をOSに戻します。

```
Link complete,     0 error (s) and     0 warning (s) found.
```

リンクの結果，リンク・エラーが検出された場合，リンカはエラーの数をディスプレイに出力して制御をOSに戻します。

```
Link complete,     2 error (s) and     0 warning (s) found.
```

リンク中にリンカの処理継続が不可能な致命的エラーが検出された場合，リンカはメッセージをディスプレイに出力してリンクを中止し，制御をOSに戻します。

例1. 存在しないオブジェクト・モジュール・ファイルを指定した場合

```
A>lk78k1 samp1.rel samp2.rel

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

A006 File not found 'SAMP1.REL'
A006 File not found 'SAMP2.REL'
Program Aborted.
```

上記の例では，存在しないオブジェクト・モジュール・ファイルを指定したためにエラーとなり，リンクが中止されました。

例2. 存在しないリンカ・オプションを指定した場合

```
A>lk78k1 78k1main.rel 78k1sub.rel -a

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

A018 Option is not recognized '-c'
Program Aborted.
```

上記の例では，存在しないリンカ・オプションを指定したためにエラーとなり，リンクが中止されました。

リンカがエラー・メッセージを出力してリンクを中止した場合，そのエラー・メッセージの原因を“**第11章　エラー・メッセージ**”で調べて対処してください。

保守／廃止

5.6 リンカ・オプション

5.6.1 リンカ・オプションの種類

リンカ・オプションはリンカの動作に細かい指示を与えるものです。リンカ・オプションは，14種類のオプションに分類できます。

表5-5 リンカ・オプション

項番	分類	オプション	説明
1	ロード・モジュール・ファイル出力指定	－O	ロード・モジュール・ファイルの出力を指定します。
		－NO	
2	ロード・モジュール・ファイル強制出力指定	－J	強制的にロード・モジュール・ファイルを出力します。
		－NJ	
3	ディバグ情報出力指定	－G	ディバグ情報をロード・モジュール・ファイルへ出力します。
		－NG	
4	スタック解決用シンボル生成指定	－S	スタック解決用のパブリック・シンボルを自動生成します。
		－NS	
5	ディレクティブ・ファイル指定	－D	指定のファイルをディレクティブ・ファイルとして入力します。
6	リンク・リスト・ファイル出力指定	－P	リンク・リスト・ファイルの出力を指定します。
		－NP	
7	リンク・リスト・ファイル情報指定	－KM	リンク・リスト・ファイル中に，マップ・リストを出力します。
		－NKM	
		－KD	リンク・リスト・ファイル中に，リンク・ディレクティブ・ファイルを出力します。
		－NKD	
		－KP	リンク・リスト・ファイル中に，パブリック・シンボル・リストを出力します。
		－NKP	
		－KL	リンク・リスト・ファイル中にローカル・シンボル・リストを出力します。
		－NKL	
8	リンク・リスト・ファイル形式指定	－LL	リストの1頁に印字する行数を変更します。
		－LF	リスト・ファイルの最後に改頁コードを付加します。
		－NLF	
9	エラー・リスト・ファイル出力指定	－E	エラー・リスト・ファイルを出力します。
		－NE	
10	ライブラリ・ファイル指定	－B	指定のファイルをライブラリ・ファイルとして入力します。
11	ライブラリ・ファイル読み込みパス指定	－I	ライブラリ・ファイルを指定したパスから読み込みます。
12	パラメータ・ファイル指定	－F	入力ファイル名，オプションを指定したファイルより入力します。
13	テンポラリ・ファイル作成パス指定	－T	テンポラリ・ファイルを指定したパスに作成します。
14	ヘルプ指定	－－	ディスプレイにヘルプ・メッセージを出力します。

この表は，リンカ・オプションを紹介するための表です。実際にリンカ・オプションを使用する場合には，“**付録C. 2 リンカ・オプション一覧**”をお読みください。

保守／廃止

5.6.2 リンカ・オプションの優先度

次の表に示すアセンブラ・オプションの内，縦軸のものと横軸のものを同時に２つ以上指定した場合の優先度について説明します。

表５－６ リンカ・オプションの優先度

	－NO	－NG	－NP	－NKM	－NKP	－NKL	－－
－J	×						×
－G	×						×
－P				△	△	△	×
－KM			×				×
－KD			×	×			×
－KP		×	×				×
－KL		×	×				×
－LL			×				×
－LF			×				×

← 横軸

↑
縦軸

【×で記した箇所】

横軸に示したオプションを指定した場合，縦軸に示したオプションは無効となります。

例 A>lK78k1 78k1main. rel 78k1sub. rel -np -km

－KMオプションは，無効となります。

【△で記した箇所】

横軸に示したオプション３つをすべて指定した場合，縦軸に示したオプションは無効となります。

例 A>lk78k1 78k1main. rel 78k1sub. rel -p -nkm -nkp -nkl

－NKM，－NKP および－NKL が同時に指定されたので，－Pオプションは無効となります。

また，－O/－NOオプションのようにオプション名の前に'N'を付加できるオプションを同時に指定した場合，あとで指定した方が有効となります。

例　A>lk78k1 78k1main. rel 78k1sub. rel -o -no

－NOオプションがあとに指定されているので，－Oオプションは無効となり，－NOオプションが有効となります。

"**表5-6　リンカ・オプションの優先度**"に記述されていないオプションは，他のオプションの影響を特に受けません。しかし，ヘルプ指定オプション'－－'が指定された場合には，すべてのオプション指定が無効となります。

保守／廃止

5.6.3 リンカ・オプションの説明

以降に，各リンカ・オプションの詳細について説明します。

(1) ロード・モジュール・ファイル出力指定 (－O/－NO)

記述形式 ：－O［出力ファイル名］

－NO

省略時解釈：－O 入力ファイル名．LNK

【機 能】

① －O オプションは，ロード・モジュール・ファイルの出力を指定します。
また，その出力先や出力ファイル名を指定します。

② －NO オプションは，ロード・モジュール・ファイルを出力しない指定をします。

【用 途】

① ロード・モジュール・ファイルの出力先や出力ファイル名を変更したいときに－O オプションを指定します。

② リンク・リスト・ファイルの出力のみが目的でリンクする場合などに，－NO オプションを指定します。この指定によってリンク時間が短縮されます。

【説 明】

① 出力ファイルとしてディスク型ファイル名とデバイス型ファイル名の NUL, AUX を指定できます。

② －O オプションを指定してもフェイタル・エラーがある場合は，ロード・モジュール・ファイルは出力されません。

③ －O オプションを指定する際に‘出力ファイル名’を省略すると，カレント・ディレクトリにロード・モジュール・ファイル‘入力ファイル名．LNK’が出力されます。

④ ‘出力ファイル名’にパス名のみを指定すると，指定したパスに‘入力ファイル名．LNK’が出力されます。

⑤ －O と－NO の両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例 1. ロード・モジュール・ファイル78K1. LNK を出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -o78k1.lnk

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

2. －NO と－ O の両オプションを指定します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -no -o

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

－NO オプションは無効となり，－ O オプションが有効となります。

保守／廃止

(2) ロード・モジュール・ファイル強制出力指定 (－J/－NJ)

記述形式 ：－J
　　－NJ
省略時解釈：－NJ

【機　能】

① －Jオプションは,フェイタル・エラーの場合にでもロード・モジュール・ファイルを出力するよう指示します。

② －NJオプションは，－Jオプションを無効にします。

【用　途】

通常フェイタル・エラーがある場合には,ロード・モジュール・ファイルは出力されません。したがって,フェイタル・エラーがあるのを承知でプログラムを実行させたい場合には,－Jオプションを指定しロード・モジュール・ファイルを出力します。

【説　明】

① －Jオプションを指定すると,フェイタル・エラーがある場合でも,ロード・モジュール・ファイルが出力されます。

② －Jと－NJの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例　ロード・モジュール・ファイルを強制的に出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -j

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(3) ディバグ情報出力指定 (－G/－NG)

記述形式 ：－G
　　－NG
省略時解釈：－G

【機　能】

① －Gオプションは，ロード・モジュール・ファイル中にディバグ情報（ローカル・シンボル情報）を付加する指定をします。

② －NGオプションは，－Gオプションを無効にします。

【用　途】

ソース・ディバッガでシンボリック・ディバグを行うときには，必ず，－Gオプションを指定します。

【説　明】

① －NOオプションを指定した場合，－Gオプションは無効となります。

② －Gと－NGの両オプションを同時に指定した場合は，あとで指定した方を優先とします。

③ －NGオプションが指定された場合，－KP，－KLオプションにかかわらず，パブリック・シンボル・リストおよびローカル・シンボル・リストは出力しません。

【使 用 例】

例　ロード・モジュール・ファイルにディバグ情報を付加します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -g

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(4) スタック解決用シンボル生成指定 (－S/－NS)

記述形式 ：－S［領域名］
－NS

省略時解釈：－NS

【機 能】

① －Sオプションは，スタック解決用のパブリック・シンボル '_@STBEG' と '_@STEND' を生成します。

② －NSオプションは，－Sオプションを無効にします。

【用 途】

スタック領域を確保するために－Sオプションを指定します。

【説 明】

① '領域名' には，利用者が定義したメモリ領域名，またはディフォールトで定義されているメモリ領域名を指定します。

② '領域名' は，大文字と小文字を区別します。

③ リンカは，－Sオプションで指定したメモリ領域の中で，セグメントが配置されていない領域のうち，最大のアドレスを探します。そして，見つかった最大のアドレス領域の先頭アドレスを値として持つパブリック・シンボル '@STEND' と，最終アドレス＋1を値として持つパブリック・シンボル '_@STBEG' を生成します。

これらのシンボルは，パブリック宣言されたNUMBER属性のシンボルとして扱われ，リンカのシンボル・テーブルの最後に登録されます。また，リンク・リスト・ファイルに出力される際，モジュール名欄は空白となります。

④ 最大の空き領域のサイズが10バイト以下の場合，ワーニング・メッセージが出力されます。

⑤ 空き領域が存在しない場合，ワーニング・メッセージが出力され，'_@STEND' と '_@STBEG' は指定したメモリ領域の最終アドレス＋1を持ちます。

⑥ '領域名' が省略された場合，'RAM' が指定されたものとします。

⑦ －Sと－NSの両オプションを同時に指定した場合は，あとで指定した方を優先します。

保守／廃止

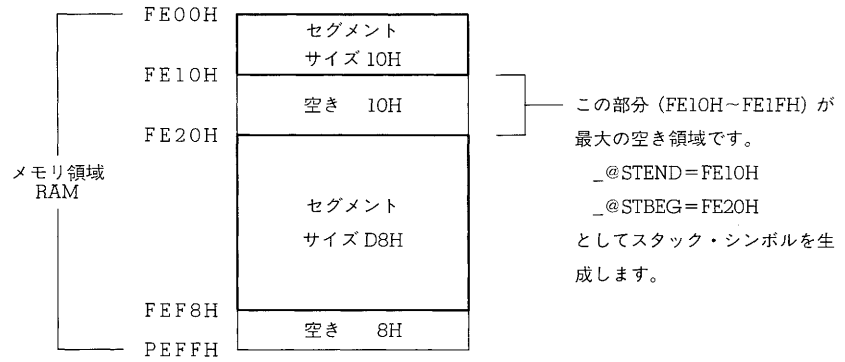
【使 用 例】

例 メモリ領域RAMにスタック領域を確保します。
(ただし，RAM領域にサイズ10Hのセグメントとsaddr領域に配置するサイズD8Hのセグメントを入力すると仮定します)

```
A>lk78k1 78k1main.rel 78k1sub.rel -s

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```



保守／廃止

(5) ディレクティブ・ファイル指定 (－D)

記述形式 ：－D ファイル名

省略時解釈：なし

【機 能】

－Dオプションは，指定ファイルをディレクティブ・ファイルとして入力することを指定します。

【用 途】

メモリ領域を新たに定義したり，ディフォールトのメモリ領域を再定義する場合，またはセグメントを特定のアドレスやメモリ領域に配置したい場合，ディレクティブ・ファイルを作成します。このディレクティブ・ファイルをリンカに入力するために，－Dオプションを指定します。

【説 明】

① 'ファイル名' として指定できるのは，ディスク型ファイル名のみです。デバイス型ファイル名を指定するとアボート・エラーとなります。

② ファイル名を省略するとアボート・エラーとなります。

③ ディレクティブ・ファイルのネストは許されません。

④ ディレクティブ・ファイル中に記述できる文字数の制限はありません。

⑤ －Dオプションを複数指定したり，ファイル名を複数指定するとアボート・エラーとなります。

⑥ ディレクティブ・ファイルの詳細については，"**5.4 リンク・ディレクティブ**" をお読みください。

【使 用 例】

例 ディフォールトのメモリ領域ROM/RAMを再定義します。

●ディレクティブ・ファイル78K1. DRの内容

```
memory ROM: (0000h, 3FFFh)
memory RAM : (D000h, 2EFFh)
```

●78K1. DR を使用してリンクします。

```
A>lk78k1 78k1main.rel 78k1sub.rel -d78k1.dr

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(6) リンク・リスト・ファイル出力指定 (－P/－NP)

記述形式 ：－P［出力ファイル名］
－NP

省略時解釈：－P入力ファイル名．MAP

【機 能】

① －Pオプションは，リンク・リスト・ファイルの出力を指定します。また，その出力先や出力ファイル名を指定します。

② －NPオプションは，－Pオプションを無効にします。

【用 途】

① リンク・リスト・ファイルの出力先や出力ファイル名を変更する場合，－Pオプションを指定します。

② ロード・モジュール・ファイルの出力だけを目的でリンクする場合などに，－NPオプションを指定します。これによりリンク時間が短縮されます。

【説 明】

① ファイル名としてディスク型ファイル名とデバイス型ファイル名を指定できます。ただし，指定できるデバイス型ファイル名はCON，PRN，NULおよびAUXです。CLOCKを指定した場合，アボート・エラーとなります。

② －Pオプションを指定する際に，‘出力ファイル名’を省略すると，カレント・ディレクトリにリンク・リスト・ファイル‘入力ファイル名．MAP’を出力します。

③ ‘出力ファイル名’にパス名のみを指定すると，指定したパスに‘入力ファイル名．MAP’を出力します。

④ －Pと－NPの両オプションを同時に指定した場合は，あとで指定した方が優先となります。

【使 用 例】

例1. リンク・リスト・ファイル（78K1．MAP）を作成します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -p78k1.map

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

例2. リンク・リスト・ファイルをプリンタに出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -pprn

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(7) リンク・リスト・ファイル情報指定

(－KM/－NKM, －KD/－NKD, －KP/－NKP, －KL/－NKL)

(a) －KM/－NKM

記述形式 ：－KM
　　－NKM
省略時解釈：－KM

【機　能】

① －KMオプションは,リンク・リスト・ファイル中にマップ・リストを出力します。

② －NKMオプションは，－KMオプションを無効にします。

【用　途】

リンク・リスト・ファイル中にマップ・リストを出力したいときに，－KMオプションを指定します。

【説　明】

① －KMと－NKMの両オプションを同時に指定した場合は，あとで指定した方を優先します。

② －NKMオプションを指定した場合,－KDオプションを指定してもリンク・リスト・ファイル中にリンク・ディレクティブ・ファイルは出力されません。

③ －NKM,－NKPおよび－NKLオプションが,すべて指定された場合,－Pオプションを指定してもリンク・リスト・ファイルは出力されません。

【使 用 例】

例　リンク・リスト・ファイル78K1.MAPにマップ・リストを出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -d78k1.map -km

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,     0 error(s) and     0 warning(s) found.
```

●78K1.MAPを参照します。

```
A>type 78k1.map

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   1

Command:   78k1main.rel 78k1sub.rel -p78k1.map -km
Para-file:
Out-file:  78K1MAIN.LNK
Map-file:  78K1.MAP
Direc-file:
Directive:


*** Link information ***

     4 output segment(s)
   38H byte(s) real data
    18 symbol(s) defined


*** Memory map ***


  SPACE=REGULAR

  MEMORY=ROM
  BASE ADDRESS=0000H    SIZE=8000H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?ASEG1                       0000H     0002H   CSEG AT
                  ?ASEG1   SAMPM      0000H     0002H
* gap *                               0002H     007EH
         ?CSEG                        0080H     0035H   CSEG
                  ?CSEG    SAMPM      0080H     0019H
                  ?CSEG    SAMPS      0099H     001CH
* gap *                               00B5H     7F4BH

  MEMORY=RAM
  BASE ADDRESS=FC80H    SIZE=0280H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?DSEG                        FC80H     0002H   DSEG
                  ?DSEG    SAMPM      FC80H     0002H
* gap *                               FC82H     017EH
         DATA                         FE00H     0001H   DSEG AT
                  DATA     SAMPM      FE00H     0001H
* gap *                               FE01H     00FFH
```

保守／廃止

(b) －KD/－NKD

記述形式　：－KD
　　－NKD
省略時解釈：－KD

【機　能】

① －KDオプションは，リンク・リスト・ファイル中にリンク・ディレクティブ・ファイルを出力します。

② －NKDオプションは，－KDオプションを無効にします。

【用　途】

リンク・リスト・ファイル中に，リンク・ディレクティブ・ファイルを出力したい場合は，－KDオプションを指定します。

【説　明】

① －KDと－NKDの両オプションを同時に指定した場合は，あとで指定した方を優先します。

② －NKMオプションを指定した場合，－KDオプションを指定してもリンク・リスト・ファイル中にリンク・ディレクティブ・ファイルは出力されません。

③ －NKM，－NKPおよび－NKLオプションが，すべて指定された場合，－Pオプションを指定してもリンク・リスト・ファイルは出力されません。

【使 用 例】

例　リンク・リスト・ファイル（78K1.MAP）にリンク・ディレクティブ・ファイルを出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -d78k1.dr -p78k1.map -kd

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

●78K1.MAPを参照します。

```
A>type 78k1.map

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   1

Command:   78k1main.rel 78k1sub.rel -d78k1.dr -p78k1.map -kd
Para-file:
Out-file:  78K1MAIN.LNK
Map-file:  78K1.MAP
Direc-file:78K1.DR
Directive: memory ROM:(00000H,3FFFH)
           memory RAM:(0D000H,2EFFH)


*** Link information ***

     4 output segment(s)
   38H byte(s) real data
    18 symbol(s) defined


*** Memory map ***


  SPACE=REGULAR

  MEMORY=ROM
  BASE ADDRESS=0000H    SIZE=3FFFH
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?ASEG1                       0000H     0002H   CSEG AT
                  ?ASEG1   SAMPM      0000H     0002H
* gap *                               0002H     007EH
         ?CSEG                        0080H     0035H   CSEG
                  ?CSEG    SAMPM      0080H     0019H
                  ?CSEG    SAMPS      0099H     001CH
* gap *                               00B5H     3F4AH

  MEMORY=RAM
  BASE ADDRESS=D000H    SIZE=2EFFH
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?DSEG                        D000H     0002H   DSEG
                  ?DSEG    SAMPM      D000H     0002H
* gap *                               D002H     2DFEH
         DATA                         FE00H     0001H   DSEG AT
                  DATA     SAMPM      FE00H     0001H
* gap *                               FE01H     00FEH
```

保守／廃止

(c) －KP/－NKP

記述形式　：－KP
　　－NKP
省略時解釈：－NKP

【機　能】

① －KPオプションは，リンク・リスト・ファイル中にパブリック・シンボル・リストを出力します。

② －NKPオプションは，－KPオプションを無効にします。

【用　途】

リンク・リスト・ファイル中にパブリック・シンボル・リストを出力する場合に－KPオプションを指定します。

【説　明】

① －KPと－NKPの両オプションを同時に指定した場合は，あとで指定した方を優先します。

② －NKM，－NKPおよび－NKLオプションが，すべて指定された場合，－Pオプションを指定してもリンク・リスト・ファイルは出力されません。

③ －NGオプションを指定した場合，－KPオプションを指定してもパブリック・シンボル・リストは出力されません。

【使 用 例】

例　リンク・リスト・ファイル（78K1.MAP）に，パブリック・シンボル・リストを出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -g -p78k1.map -kp

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

●78K1. MAP を参照します。

```
A>type 78k1.map

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   1

Command:   78k1main.rel 78k1sub.rel -g -p78k1.map -kp
Para-file:
Out-file:  78K1MAIN.LNK
Map-file:  78K1.MAP
Direc-file:
Directive:

*** Link information ***

     4 output segment(s)
   38H byte(s) real data
    18 symbol(s) defined

*** Memory map ***

  SPACE=REGULAR

  MEMORY=ROM
  BASE ADDRESS=0000H    SIZE=8000H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?ASEG1                       0000H     0002H   CSEG AT
                  ?ASEG1   SAMPM      0000H     0002H
* gap *                               0002H     007EH
         ?CSEG                        0080H     0035H   CSEG
                  ?CSEG    SAMPM      0080H     0019H
                  ?CSEG    SAMPS      0099H     001CH
* gap *                               00B5H     7F4BH

  MEMORY=RAM
  BASE ADDRESS=FC80H    SIZE=0280H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?DSEG                        FC80H     0002H   DSEG
                  ?DSEG    SAMPM      FC80H     0002H
* gap *                               FC82H     017EH
         DATA                         FE00H     0001H   DSEG AT
                  DATA     SAMPM      FE00H     0001H
* gap *                               FE01H     00FFH
```

```
78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   2

*** Public symbol list ***

MODULE    ATTR       VALUE    NAME

SAMPM     ADDR       0000H    MAIN
SAMPM     ADDR       0080H    START
SAMPS     ADDR       0099H    CONVAH
```

保守／廃止

(d) －KL/－NKL

記述形式　：－KL
　　－NKL
省略時解釈：－NKL

【機　能】

① －KLオプションは，リンク・リスト・ファイル中にローカル・シンボル・リストを出力します。

② －NKLオプションは，－KLオプションを無効にします。

【用　途】

リンク・リスト・ファイル中にローカル・シンボル・リストを出力する場合に，－KLオプションを指定します。

【説　明】

① －KLと－NKLの両オプションを同時に指定した場合は，あとで指定した方を優先します。

② －NKM，－NKPおよび－NKLオプションが，すべて指定された場合，－Pオプションを指定してもリンク・リスト・ファイルは出力されません。

③ －NGオプションを指定した場合は，－KLオプションを指定してもローカル・シンボル・リストは出力されません。

【使 用 例】

例　リンク・リスト・ファイル（78K1.MAP）に，ローカル・シンボル・リストを出力します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -g -p78k1.map -kl

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

●78K1. MAP を参照します。

```
A>type 78k1.map

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   1

Command:   78k1main.rel 78k1sub.rel -g -p78k1.map -kl
Para-file:
Out-file:  78K1MAIN.LNK
Map-file:  78K1.MAP
Direc-file:
Directive:

*** Link information ***

     4 output segment(s)
   38H byte(s) real data
    18 symbol(s) defined

*** Memory map ***

  SPACE=REGULAR

  MEMORY=ROM
  BASE ADDRESS=0000H    SIZE=8000H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?ASEG1                       0000H     0002H   CSEG AT
                  ?ASEG1   SAMPM      0000H     0002H
* gap *                               0002H     007EH
         ?CSEG                        0080H     0035H   CSEG
                  ?CSEG    SAMPM      0080H     0019H
                  ?CSEG    SAMPS      0099H     001CH
* gap *                               00B5H     7F4BH

  MEMORY=RAM
  BASE ADDRESS=FC80H    SIZE=0280H
         OUTPUT   INPUT    INPUT      BASE      SIZE
                                             ⋮
```

```
78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   2

*** Local symbol list ***

MODULE    ATTR       VALUE    NAME

SAMPM     MOD                 SAMPM
SAMPM     DSEG                ?DSEG
SAMPM     ADDR       FC80H    STASC
SAMPM     CSEG                ?CSEG
SAMPM     CSEG                ?ASEG1
SAMPM     ADDR       FE00H    HDTSA
SAMPM     DSEG                DATA
SAMPS     MOD                 SAMPS
SAMPS     CSEG                ?CSEG
SAMPS     ADDR       00ACH    SASC
SAMPS     ADDR       00B2H    SASC1
```

保守／廃止

(8) リンク・リスト・ファイル形式指定 (－LL, －LF/－NLF)

(a) －LL

記述形式 ：－LL [行数]

省略時解釈：－LL66 (ディスプレイ出力の場合は改頁しません)

【機 能】

－LLオプションは，リンク・リスト・ファイルの1頁の行数を指定します。

【用 途】

リンク・リスト・ファイルの1頁の行数を変更したいときに，－LLオプションで指定します。

【説 明】

① －LLオプションで指定できる行数の範囲は次のとおりです。

20 ≦ 1頁に印字する行数 ≦ 32767

範囲外の数値や数値以外のものが指定された場合には，アボート・エラーとなります。

② 行数が省略された場合，66が指定されたものとみなします。

③ 行数に0を指定した場合は改頁しません。

④ －NPオプションが指定された場合，－LLオプションは無効となります。

【使 用 例】

例 リンク・リスト・ファイルの1頁の行数を20行に指定します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -p78k1.map -ll20

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

●78K1. MAP を参照します。

```
A>type 78k1.map

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   1

Command:   78k1main.rel 78k1sub.rel -p78k1.map -l120
Para-file:
Out-file:  78K1MAIN.LNK
Map-file:  78K1.MAP
Direc-file:
Directive:

*** Link information ***

     4 output segment(s)
   38H byte(s) real data

-------------------------------------------------------------------------------------------------

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   2

    18 symbol(s) defined

*** Memory map ***

  SPACE=REGULAR

  MEMORY=ROM
  BASE ADDRESS=0000H    SIZE=8000H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS

-------------------------------------------------------------------------------------------------

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   3

         ?ASEG1                        0000H     0002H   CSEG AT
                  ?ASEG1   SAMPM       0000H     0002H
* gap *                                0002H     007EH
         ?CSEG                         0080H     0035H   CSEG
                  ?CSEG    SAMPM       0080H     0019H
                  ?CSEG    SAMPS       0099H     001CH
* gap *                                00B5H     7F4BH

  MEMORY=RAM
  BASE ADDRESS=FC80H    SIZE=0280H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS

-------------------------------------------------------------------------------------------------

78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   4

         ?DSEG                         FC80H     0002H   DSEG
                  ?DSEG    SAMPM       FC80H     0002H
* gap *                                FC82H     017EH
         DATA                          FE00H     0001H   DSEG AT
                  DATA     SAMPM       FE00H     0001H
* gap *                                FE01H     00FFH
```

保守／廃止

(b) －LF/－NLF

記述形式　：－LF
　　－NLF
省略時解釈：－NLF

【機　能】

① －LFオプションは，リンク・リスト・ファイルの最後に，改頁コード（FF）を付加する指定をします。

② －NLFオプションは，－LFオプションを無効にします。

【用　途】

リンク・リスト・ファイルの内容を印字したあとで改頁しておきたい場合，－LFオプションを指定して改頁コードを付加します。

【説　明】

① －NPオプションを指定した場合，－LFオプションは無効となります。

② －LFと－NLFの両オプションを同時に指定した場合，あとで指定した方を優先します。

【使 用 例】

例　リンク・リスト・ファイルの最後に改頁コードを付加します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -p78k1.map -lf

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(9) エラー・リスト・ファイル出力指定 (－E/－NE)

記述形式 ：－E［出力ファイル名］
－NE

省略時解釈：－NE

【機 能】

① －Eオプションは，エラー・リスト・ファイルの出力を指定します。また，その出力先や出力ファイル名を指定します。

② －NEオプションは，－Eオプションを無効にします。

【用 途】

エラー・リスト・ファイルの出力先や出力ファイル名を変更したいときに，－Eオプションを指定します。

【説 明】

① ファイル名として，ディスク型ファイル名とデバイス型ファイル名を指定できます。ただし，デバイス型ファイル名としてCLOCKを指定した場合は，アボート・エラーとなります。

② －Eオプションを指定する際に出力ファイル名を省略すると，エラー・リスト・ファイル名は，‘入力ファイル名．ELK’となります。

③ －Eオプションを指定する際にドライブ名を省略すると，カレント・ドライブにエラー・リスト・ファイルが出力されます。

④ －Eと－NEの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例 エラー・リスト・ファイル（78K1．ELK）を作成します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -dsamp.dr -e78k1.elk

78K/I Series Linker Vx.xx [xx xxx xx]
   Copyright (C) NEC Corporation 1989

SAMP.DR(2) : F111 Memory area 'RAM' redefinition out of range
SAMP.DR(3) : F102 Directive syntax error
Program Aborted.
```

保守／廃止

●ディレクティブ・ファイルの内容に誤りがあったので，78K1. ELK を参照します。

```
A>type 78k1.elk

SAMP.DR(2) : F111 Memory area 'RAM' redefinition out of range
SAMP.DR(3) : F102 Directive syntax error
```

保守／廃止

(10) ライブラリ・ファイル指定 (－B)

記述形式 ：－B ファイル名

省略時解釈：なし

【機 能】

－Bオプションは,指定ファイルをライブラリ・ファイルとして入力することを指定します。

【用 途】

リンカは入力モジュールが参照しているモジュールを,ライブラリ・ファイルから探し出し,そのモジュールだけを入力モジュールと結合します。

ライブラリ・ファイルは，あらかじめ複数のモジュールを登録して1つのファイルにまとめておくためのものです。

共通的なモジュールをライブラリ・ファイルとして作成しておけば，ファイル管理の面でも操作性の面においても効率がよくなります。ライブラリ・ファイルをリンカに入力するには,－Bオプションを指定します。

【説 明】

① ファイル名としてディスク型ファイル名以外は指定できません。

② ファイル名は省略できません。

③ ファイル名としてパス名を含めて指定した場合は,指定されたパスからライブラリ・ファイルを入力します。指定されたパスにライブラリ・ファイルが存在しない場合は，エラーとなります。

④ ファイル名としてパス名を含まずに指定した場合には,－Iオプションの指定およびディフォールトのサーチ・パスからライブラリ・ファイルを入力します。

⑤ －Bオプションが複数指定された場合は，指定順にライブラリ・ファイルの入力を行います。－Bオプションは最大10個まで指定できます。

⑥ ライブラリ・ファイルの作成方法の詳細は,“**第7章 ライブラリアン**”をお読みください。

【使 用 例】

例 ライブラリ・ファイル（78K1.LIB）を入力します。
（ライブラリ・ファイルには，78K1SUB.RELが登録されています。）

```
A>lk78k1 78k1main.rel -b78k1.lib

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(11) ライブラリ・ファイル読み込みパス指定 (－I)

記述形式 ：－I パス名 [, パス名] … (複数指定可能)

省略時解釈：環境変数 'LIB78K1' により指定されたパス

指定されていない場合は，カレント・パス

【機 能】

ライブラリ・ファイルを指定したパスから入力するよう指示します。

【用 途】

ライブラリ・ファイルをあるパスから検索したいときに指定します。

【説 明】

① －Iオプションは，－Bオプションでパス名を含まないライブラリ・ファイル名が指定された場合にのみ有効です。

② －Iは複数指定できます。また，‘，’で区切ることにより，1度に複数のパスを指定できます。この際‘，’の前後には空白を入れることはできません。

③ パス名は，1回のリンクで最大10個まで指定できます。パス名が複数指定された場合，リンカは指定順にライブラリ・ファイルのサーチを行います。

④ 指定したパスにライブラリ・ファイルが存在しない場合でもエラーとはなりません。

⑤ パス名が省略された場合，アボート・エラーとなります。

⑥ －Bオプションでパス名を含まずにライブラリ・ファイルを指定した場合，リンカは以下に示すパスをサーチします。

1. －Iオプションで指定したパス
2. 環境変数 'LIB78K1' で指定されたパス
3. カレント・パス

いずれのパスにも指定された名前のライブラリ・ファイルが存在しない場合にはエラーとなります。

保守／廃止

【使 用 例】

例 ライブラリ・ファイルをディレクトリLIBから検索します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -b78k1.lib -i¥lib

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(12) パラメータ・ファイル指定 （－F）

記述形式 ：－F ファイル名

省略時解釈：起動行上からのみオプション，入力ファイル名の入力が可能

【機 能】

－Fオプションは，オプションあるいは入力ファイル名を指定のファイルから入力する指定をします。

【用 途】

① コマンド行では，リンカの起動に必要な情報を指定しきれないときに，－Fオプションを指定します。

② リンクするたび繰り返し同じようにオプションを指定する場合には，それらをパラメータ・ファイルに記述しておき，－Fオプションで指定します。

【説 明】

① ‘ファイル名’として指定できるのは，ディスク型ファイル名のみです。デバイス型ファイル名を指定するとアボート・エラーとなります。

② ファイル名を省略するとアボート・エラーとなります。

③ パラメータ・ファイルのネストは許されません。パラメータ・ファイル中で，－Fオプションを指定するとアボート・エラーとなります。

④ パラメータ・ファイル中に記述できる文字数の制限はありません。

⑤ 空白とタブおよび‘↵’をオプションあるいは入力ファイル名の区切りとします。

⑥ パラメータ・ファイル中に記述したオプションあるいは入力ファイル名は，コマンド行上のパラメータ・ファイル指定のあった位置に展開されます。

⑦ 展開されたオプションは，あとで指定したものが優先です。

⑧ ‘;’以降に記述された文字は‘↵’またはEOFの前まですべてコメントと解釈します。

⑨ －Fオプションを複数指定するとアボート・エラーとなります。

【使 用 例】

例 パラメータ・ファイルを使用してリンクします。

●パラメータ・ファイル（78K1.PLK）の内容

```
;parameter file
78k1main.rel 78k1sub.rel -o78k1.lnk -p78k1.map -e
-ta:¥tmp -g
```

●コマンド行には，次のように入力します。

```
A>lk78k1 -f78k1.plk

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(13) テンポラリ・ファイル作成パス指定 (－T)

記述形式 ：－T パス名

省略時解釈：環境変数 'TMP' により指定されたパスに作成します。
指定されていない場合は，カレント・パスに作成します。

【機 能】

－Tオプションは，テンポラリ・ファイルを作成するパスを指示します。

【用 途】

テンポラリ・ファイルの作成場所を指定できます。

【説 明】

① パス名として，パス以外のものは指定できません。

② パス名は省略できません。

③ 以前に作成されたテンポラリ・ファイルが存在している場合でも，ファイル保護がされていなければ，上書きします。

④ 必要とするメモリ・サイズがある間は，テンポラリ・ファイルをメモリに展開します。メモリが足りなくなった時点で，メモリに展開していたテンポラリ・ファイルの内容をディスクに書き出します。以降のテンポラリ・ファイルへのアクセスは，セーブしたディスク・ファイルに対して行います。

⑤ テンポラリ・ファイルは，リンク終了時に削除されます。また，キー入力（CTRL－C）によってリンクが中止されたときも，削除されます。

⑥ テンポラリ・ファイルの作成パスは，次の順番で決定されます。

1. －Tオプションで指定されたパス
2. 環境変数TMPに設定されているパス（－Tオプション省略の場合）
3. カレント・パス（TMPが設定されていない場合）

なお，1または2を指定した場合，指定されたパスにテンポラリ・ファイルが作成できなければアボート・エラーとなります。

【使 用 例】

例 テンポラリ・ファイルをディレクトリ 'TMP' に作成するよう指定します。

```
A>lk78k1 78k1main.rel 78k1sub.rel -t¥tmp

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.
```

保守／廃止

(14) ヘルプ指定 (--)

記述形式 : --

省略時解釈 : 表示しない

【機 能】

--オプションは，ヘルプ・メッセージをディスプレイに表示します。

【用 途】

ヘルプ・メッセージは，リンカ・オプションとその説明の一覧です。リンカを実行するときに参照してください。

【説 明】

--オプションを指定すると，他のリンカ・オプションはすべて無効となります。

【使 用 例】

例 --オプションを指定するとヘルプ・メッセージがディスプレイに出力されます。

```
A>lk78k1 --

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

usage : LK78K1 [option[ ...]] input-file[ ...] [option[ ...]]
The option is as follows ([] means omissible).
-Ffile           :Input option or input-file name from specified file.
-Dfile           :Read directive file from specified file.
-Bfile           :Read library file from specified file.
-Idirectory[,directory..]  :Set library file search path.
-O[file]/-NO     :Create load module file [with specified name] / Not.
-P[file]/-NP     :Create link map file [with specified name] / Not.
-E[file]/-NE     :Create error list file [with specified name] / Not.
-Tdirectory      :Set temporary directory.
-KM/-NKM         :Output map list to link map file / Not.
-KD/-NKD         :Output directive file image to link map file / Not.
-KP/-NKP         :Output public symbol list to link map file / Not.
-KL/-NKL         :Output local symbol list to link map file / Not.
-LL[page length] :Specify link map file lines per page.
-LF/-NLF         :Add Form Feed at end of the link map file / Not.
-S[memory area]/-NS  :Create stack symbol [in specified memory area] / Not.
-G/-NG           :Output symbol information to load module file / Not.
-J/-NJ           :Create load module file if fatal error occurred / Not.
--               :Show this message.
DEFAULT ASSIGNMENT: -O -P -NE -KM -KD -NKP -NKL -LL66 -NLF -NS -G -NJ
                                    ⋮
```

保守／廃止

第６章　オブジェクト・コンバータ

オブジェクト・コンバータは，RA78K/Iのリンカが出力したロード・モジュール・ファイル（すべての参照アドレス情報が解決されていなければなりません）を入力し，それをHEX形式オブジェクト・モジュール・ファイルとして出力します。

さらに，シンボリック・ディバグ時に必要となるシンボル情報をシンボル・テーブル・ファイルとして出力します。

オブジェクト・コンバータ・エラーがある場合は，エラー・メッセージをディスプレイに出力し，エラーの原因を明示します。

保守／廃止

6.1　オブジェクト・コンバータの入出力ファイル

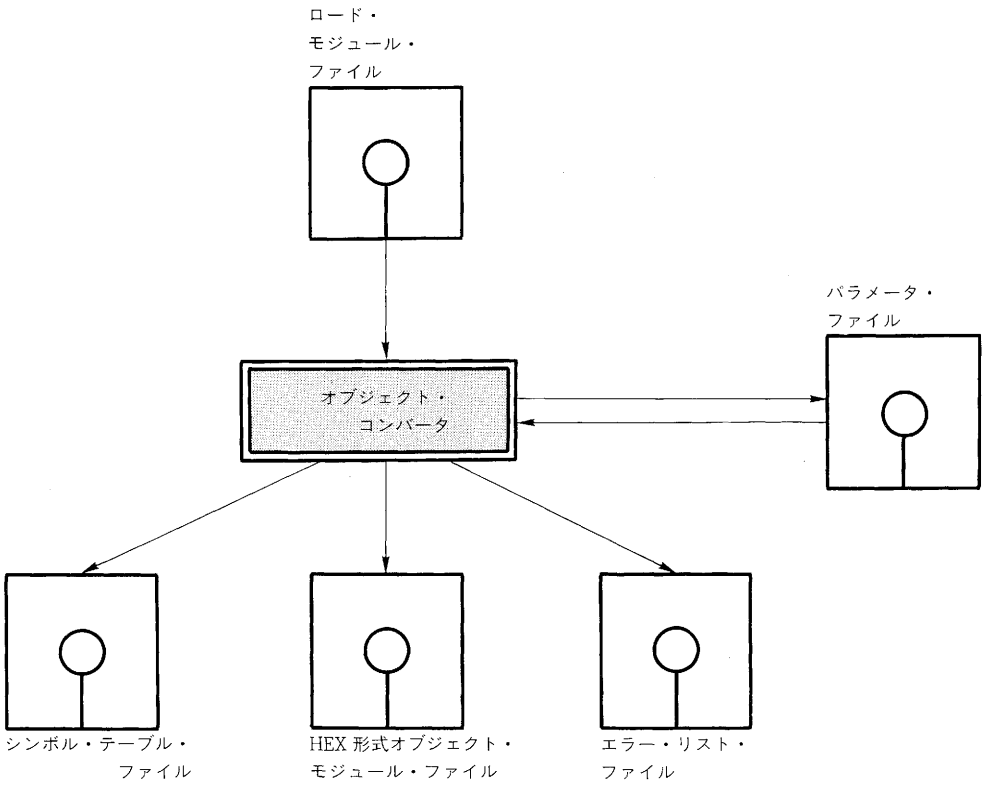
オブジェクト・コンバータの入出力ファイルについて以下に示します。

表6－1　オブジェクト・コンバータの入出力ファイル

種類	ファイル名	説　　明	ディフォールト・ファイル・タイプ
入力ファイル	ロード・モジュール・ファイル	○リンクの結果のオブジェクト・コードのバイナリ・イメージ・ファイルです。 ○リンカの出力したファイルです。	.LNK
	パラメータ・ファイル	○実行プログラムのパラメータを内容とするファイルです。 ○ユーザ作成ファイルです。	.POC
出力ファイル	HEX形式オブジェクト・モジュール・ファイル	○ロード・モジュール・ファイルを，インテル標準HEX形式オブジェクト・フォーマットで変換したファイルです。 ○マスクROM発注時またはPROMプログラマ使用時に使います。 ○インサーキット・エミュレータにロードするファイルです。	.HEX
	シンボル・テーブル・ファイル	○入力ファイルの各モジュールに含まれるシンボルの情報を持つファイルです。	.SYM
	エラー・リスト・ファイル	○オブジェクト・コンバート時のエラー情報を持つファイルです。	.EOC

保守／廃止

図6-1　オブジェクト・コンバータの入出力ファイル



保守／廃止

6.2　オブジェクト・コンバータの機能

(1) 拡張空間への対応

オブジェクト・コンバータは，拡張空間に配置されたセグメントにコードが出力されているときには，空間ごとに別々のHEX形式オブジェクト・モジュール・ファイルを生成します。

また，拡張空間に配置されたセグメントにADDRESS属性または，BIT属性を持つシンボルが定義されているときには，空間ごとに別々のシンボル・テーブル・ファイルを生成します。NUMBER属性を持つシンボルは，すべて通常空間に対するシンボル・テーブル・ファイルに出力します。

次に拡張空間に対する，HEX形式オブジェクト・モジュール・ファイルとシンボル・テーブル・ファイルのファイル・タイプを示します。

表6-2　拡張空間に対する出力ファイルのファイル・タイプ

ファイル	通常空間	拡　張　空　間								
	REGULAR	EX1	EX2	EX3	EX4	EX5	…	EX13	EX14	EX15
HEX	.HEX	.H1	.H2	.H3	.H4	.H5	…	.H13	.H14	.H15
シンボル	.SYM	.S1	.S2	.S3	.S4	.S5	…	.S13	.S14	.S15

(2) HEX形式オブジェクト・モジュール・ファイル

オブジェクト・コンバータの出力するHEX形式オブジェクト・モジュール・ファイルは，PROMプログラマやディバッガなどのHEXローダに入力可能です。

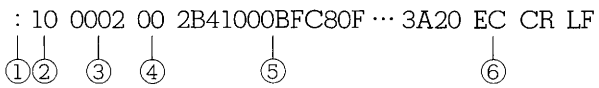
次にサンプル・プログラムのHEX形式オブジェクト・モジュール・ファイルを示します。

```
:0200000080007E
:100080000BFCE0FE2BC40009C0FF00BC0028990057
:100090006480FCD350D25014FEB900059928AC00FE
:1000A0002431B900059928AC00242156AF0A8302F7
:0500B000A807A830566E
:01FE00001AE7
:00000001FF
```

1～5行がオブジェクト・コードのレコードで，最終行が最終レコードです。

保守／廃止

【HEX 形式オブジェクト・モジュール・ファイルのフォーマット】



① レコード・マーク

レコードの始まりを示します。

② コード数（2桁）

レコードに納められるコードのバイト数です。最大16バイトになります。

最終レコードは，00Hで示します。

③ ロケーション・アドレス（4桁）

そのレコードで表すコードの先頭アドレスを示します。

最終レコードは0000Hで示します。

④ レコード・タイプ（2桁）

00Hは，データ・レコードを表し，01Hは最終レコードを表します。

⑤ コード（最大32桁）

オブジェクト・コードを1バイトずつ上位4ビット，下位4ビットに分けて示します。

コードは，最大16バイトまで表現されます。

最終レコードには，本フィードは存在しません。

⑥ チェック・サム（2桁）

コード数からコードまでのデータを0から順に減算した値が入ります。

(3) シンボル・テーブル・ファイル

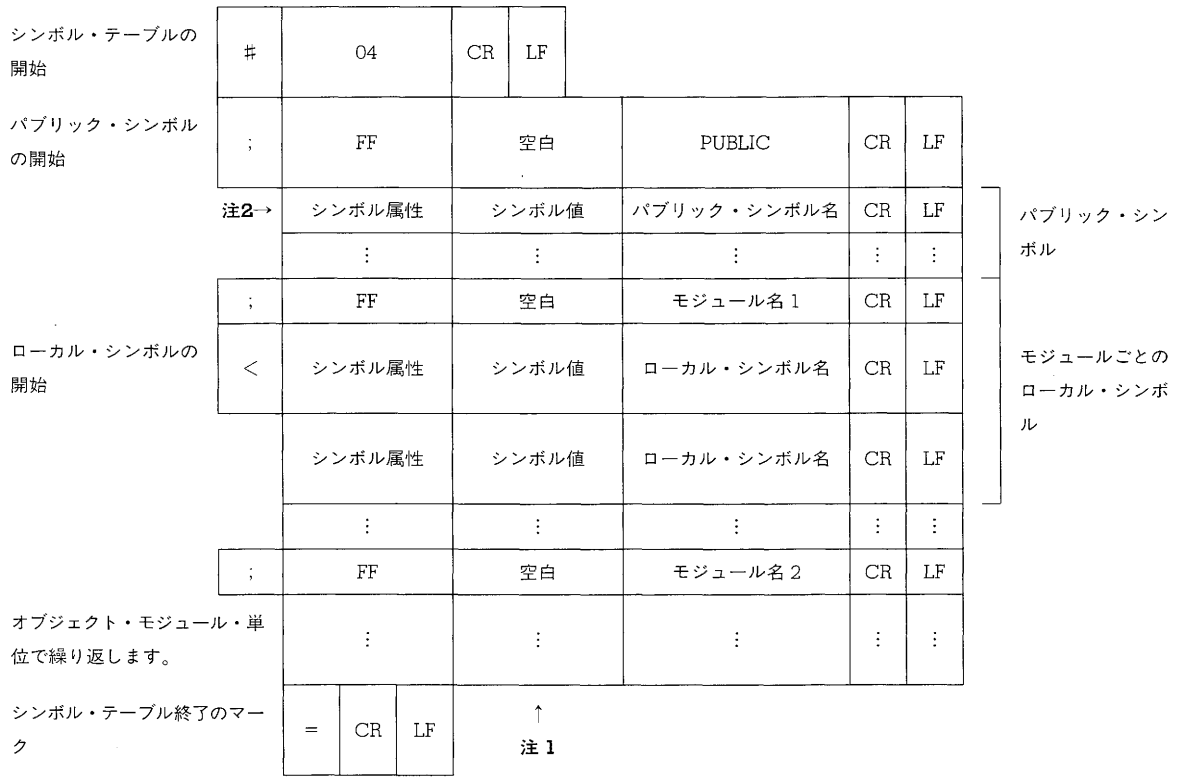
オブジェクト・コンバータの出力するシンボル・テーブル・ファイルは，ディバッガへの入力となります。

次にサンプル・プログラムのシンボル・テーブル・ファイルを示します。

```
#04
;FF    PUBLIC
010099CONVAH
010000MAIN
010080START
;FF    SAMPM
<01FE00HDTSA
02FC80STASC
;FF    SAMPS
<0100ACSASC
0100B2SASC1
=
```

保守／廃止

【シンボル・テーブル・ファイルのフォーマット】



注 1. この欄は，4 文字固定です。

2. シンボル属性は，次の値をとります。

値	シンボル属性
00	EQU 疑似命令で定義した定数
01	コード・セグメント内のレーベル
02	データ・セグメント内のレーベル
03	ビット・シンボル
FF	モジュール名

保守／廃止

6.3 オブジェクト・コンバータの起動方法

6.3.1 オブジェクト・コンバータの起動

オブジェクト・コンバータの起動には，次の２つの方法があります。

(1) コマンド行での起動

```
X > [パス名] oc78k1 [△オプション] …△ロード・モジュール・ファイル名 [△オプション] … [△]
|     |        |        |                      |                          |
①    ②       ③       ④                     ⑤                         ④
```

① OSのプロンプト。

② オブジェクト・コンバータのコマンド・ファイルを格納しているパス名。

③ オブジェクト・コンバータのコマンド・ファイル名。

MS-DOS ver2.11の場合，すべてのプログラム・ファイルはカレント・ディレクトリになければなりません。

④ オブジェクト・コンバータに対して動作の詳細を指定します。

複数のオブジェクト・コンバータ・オプションを指定する場合は，おのおののオブジェクト・コンバータ・オプション間を空白で区切ってください。オブジェクト・コンバータ・オプションの詳細については，“**6.4 オブジェクト・コンバータ・オプション**”をお読みください。

⑤ コンバートするロード・モジュール・ファイル名。

例　A>oc78k1 78k1.lnk -osample.hex

(2) パラメータ・ファイルによる起動

パラメータ・ファイルは，起動に必要な情報がコマンド行に指定しきれない場合やオブジェクト・コンバートするたびに同じオプションを繰り返し指定するような場合に使用します。

パラメータ・ファイルを使用する場合には，コマンド行にパラメータ・ファイル指定オプション（－F）を指定します。

パラメータ・ファイルによる起動方法を以下に示します。

```
X > oc78k1 [△ロード・モジュール・ファイル] △-f パラメータ・ファイル名
                                              |        |
                                              ①       ②
```

① パラメータ・ファイル指定オプション

② オブジェクト・コンバータの起動に必要な情報を含んだファイル

6

備考　パラメータ・ファイルはエディタなどで作成します。

パラメータ・ファイル内での記述規則を次に示します。

```
[[[△] オプション [△オプション] … [△] △]] …
```

備考 1. コマンド行でロード・モジュール・ファイル名を省略した場合，パラメータ・ファイル内でロード・モジュール・ファイル名を 1 つだけ指定できます。

2. ロード・モジュール・ファイル名は，オプションのあとでも記述できます。

3. パラメータ・ファイルには，コマンド行で指定すべきすべてのオブジェクト・コンバータ・オプション，出力ファイル名を記述します。

例　パラメータ・ファイル（78K1.POC）をエディタで作成します。

○78K1.POC の内容

```
: parameter file
78k1.lnk -osample.hex
-ssample.sym -r
```

○パラメータ・ファイル（78K1.POC）を使用してオブジェクト・コンバータを起動します。

```
A > oc78k1 -f78k1.poc
```

保守／廃止

6.3.2　実行開始，終了メッセージ

(1) 実行開始メッセージ

オブジェクト・コンバータが起動すると，ディスプレイに以下の実行開始メッセージを表示します。

```
78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990
```

(2) 実行終了メッセージ

オブジェクト・コンバートの結果，エラーが検出されなかった場合には，オブジェクト・コンバータは，次のメッセージをディスプレイに出力して制御をOSに戻します。

```
Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```


オブジェクト・コンバートの結果，エラーが検出された場合は，オブジェクト・コンバータはエラーの数をディスプレイに出力して制御をOSに戻します。

```
1  Object Conversion Complete,       2 error(s) and      0 warning(s) found.
```

オブジェクト・コンバート中に，オブジェクト・コンバータの処理継続が不可能な致命的エラーが検出された場合，オブジェクト・コンバータはメッセージをディスプレイに出力してオブジェクト・コンバートを中止し，制御をOSに戻します。

例1. 存在しないロード・モジュール・ファイル名を指定

```
A>oc78k1 sample.lnk

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

A006 file not found 'SAMPLE.LNK'
Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

この例では，存在しないロード・モジュール・ファイルを指定したためにエラーとなり，オブジェクト・コンバートが中止されました。

例２. 存在しないオブジェクト・コンバータ・オプションを指定

```
A>oc78k1 78k1.lnk -a

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

A018 Option is not recognized '-a'
Program aborted.
```

この例では，存在しないオブジェクト・コンバータ・オプションを指定したために，エラーとなり，オブジェクト・コンバートが中止されました。

オブジェクト・コンバータがエラー・メッセージを出力して，オブジェクト・コンバートを中止した場合，そのエラー・メッセージの原因を“**第11章　エラー・メッセージ**”で調べて対処してください。

保守／廃止

6.4　オブジェクト・コンバータ・オプション

6.4.1　オブジェクト・コンバータ・オプションの種類

オブジェクト・コンバータ・オプションは，オブジェクト・コンバータの動作に細かい指示を与えるものです。オブジェクト・コンバータ・オプションは，7種類のオプションに分類できます。

以下にオブジェクト・コンバータ・オプションの分類と説明を示します。

表6-3　オブジェクト・コンバータ・オプション

項　番	分　　類	オプション	説　　明
1	HEX形式 オブジェクト・モジュール・ ファイル出力指定	－O	HEX形式オブジェクト・モジュール・ファイルの出力を指定します。
		－NO	
2	シンボル・テーブル・ファイル出力指定	－S	シンボル・テーブル・ファイルの出力を指定します。
		－NS	
3	オブジェクト・アドレス順 ソート指定	－R	HEX形式オブジェクトをアドレス順にソートします。
		－NR	
4	オブジェクト充填指定	－U	HEX形式オブジェクトが出力されないアドレス領域に対して，指定した充填値をオブジェクト・コードとして出力します。
5	エラー・リスト・ファイル 出力指定	－E	エラー・リスト・ファイルを出力します。
		－NE	
6	パラメータ・ファイル指定	－F	入力ファイル名，オプションを指定したファイルより入力します。
7	ヘルプ指定	－－	ディスプレイへヘルプ・メッセージを出力します。

備考　オブジェクト・コンバータ・オプションの詳細については，“**付録C.3　オブジェクト・コンバータ・オプション一覧**”を参照してください。

保守／廃止

6.4.2　オブジェクト・コンバータ・オプションの説明

以下に，各オブジェクト・コンバータ・オプションの詳細を説明します。

(1) HEX 形式オブジェクト・モジュール・ファイル出力指定　(－O/－NO)

記述形式　：－O［出力ファイル名］
　　：－NO
省略時解釈：－O 入力ファイル名．HEX
　　（拡張空間に対してのファイル・タイプは，'.H1' ～ '.H15'）

【機　　能】

① －O オプションは，HEX 形式オブジェクト・モジュール・ファイルの出力を指定します。また，その出力先や出力ファイル名を指定します。

② －NO オプションは，HEX 形式オブジェクト・モジュール・ファイルを出力しないことを指定します。

【用　　途】

① HEX 形式オブジェクト・モジュール・ファイルの出力先や出力ファイル名を変更したいときに，－O オプションを指定します。

② シンボル・テーブル・ファイルの出力のみが目的でオブジェクト・コンバートする場合などに，－NO オプションを指定します。これによりオブジェクト・コンバート時間が短縮されます。

【説　　明】

① '出力ファイル名' には，ディスク型ファイル名を指定してください。
　デバイス型ファイル名を指定すると，アボート・エラーとなります。

② －O オプションを指定する際に '出力ファイル名' を省略すると，カレント・ディレクトリに HEX 形式オブジェクト・モジュール・ファイル('入力ファイル名.HEX')が出力されます。

③ '出力ファイル名' にパス名のみを指定すると，指定したパスに '入力ファイル名.HEX' が出力されます。

④ －O と－NO の両オプションを同時に指定した場合は，あとで指定した方を優先します。

オブジェクト・コンバータは，拡張空間に配置されたセグメントにコードが出力されているときには，空間ごとに別々の HEX 形式オブジェクト・モジュール・ファイルを生成します。

次に拡張空間に対する HEX 形式オブジェクト・モジュール・ファイルのファイル・タイプを示します。

保守／廃止

ファイル	通常空間	拡張空間								
	REGULAR	EX1	EX2	EX3	EX4	EX5	…	EX13	EX14	EX15
HEX	.HEX	.H1	.H2	.H3	.H4	.H5	…	.H13	.H14	.H15

【使 用 例】

例 1. HEX 形式オブジェクト・モジュール・ファイル（SAMPLE.HEX）を出力します。

```
A>oc78k1 78k1.lnk -osample.hex

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

2. －NO と－O の両オプションを指定します。

```
A>oc78k1 78k1.lnk -no -o

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

－NO オプションは無効となり，－O オプションが有効となります。

保守／廃止

(2) シンボル・テーブル・ファイル出力指定　(－S/－NS)

記述形式　：－S［出力ファイル名］
　　：－NS
省略時解釈：－S 入力ファイル名.SYM
　　（拡張空間に対してのファイル・タイプは,'.S1'～'.S15'）

【機　能】

① －Sオプションは，シンボル・テーブル・ファイルの出力を指定します。また，その出力先や出力ファイル名を指定します。

② －NSオプションは，シンボル・テーブル・ファイルを出力しないことを指定します。

【用　途】

① シンボル・テーブル・ファイルの出力先や出力ファイル名を変更したいときに，－Sオプションを指定します。

② HEX形式オブジェクト・モジュール・ファイルの出力のみが目的でオブジェクト・コンバートする場合などに，－NSオプションを指定します。
　こうすることによってオブジェクト・コンバート時間が短縮されます。

【説　明】

① '出力ファイル名'には，ディスク型ファイル名を指定してください。
　デバイス型ファイル名を指定した場合，アボート・エラーとなります。

② －Sオプションを指定する際に'出力ファイル名'を省略すると,カレント・ディレクトリにシンボル・テーブル・ファイル（'入力ファイル名.SYM'）が出力されます。

③ '出力ファイル名'にパス名のみを指定すると，指定したパスに'入力ファイル名.SYM'が出力されます。

④ －Sと－NSの両オプションを同時に指定した場合は,あとで指定した方を優先します。

拡張空間に配置されたセグメントにADDRESS属性または,BIT属性を持つシンボルが定義されている場合，空間ごとに別々のシンボル・テーブル・ファイルを生成します。

NUMBER属性を持つシンボルは,すべて通常空間に対するシンボル・テーブル・ファイルに出力します。

次に拡張空間に対するシンボル・テーブル・ファイルの，ファイル・タイプを示します。

ファイル	通常空間	拡　張　空　間								
	REGULAR	EX1	EX2	EX3	EX4	EX5	…	EX13	EX14	EX15
シンボル	.SYM	.S1	.S2	.S3	.S4	.S5	…	.S13	.S14	.S15

【使 用 例】

例 1. シンボル・テーブル・ファイル（SAMPLE.SYM）を出力します。

```
A>oc78k1 78k1.lnk -ssample.sym

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

2. －NSと－Sの両オプションを指定します。

```
A>oc78k1 78k1.lnk -ns -s

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

－NSオプションは無効となり，－Sオプションが有効となります。

保守／廃止

(3) オブジェクト・アドレス順ソート指定　(－R/－NR)

記述形式　：－R
　　：－NR
省略時解釈：－NR

【機　　能】

① －Rオプションは，HEX形式オブジェクトをアドレス順にソートして出力します。

② －NRオプションは，HEX形式オブジェクトをロード・モジュール・ファイルに格納されている順に出力します。

【用　　途】

HEX形式オブジェクトがアドレス順にソートされている必要のある場合に，－Rオプションを指定します。

【説　　明】

① －Rと－NRの両オプションを同時に指定した場合は，あとで指定した方を優先します。

② －NOオプションを指定した場合，－R/－NRオプションは無効となります。

【使 用 例】

例　HEX形式オブジェクトをアドレス順にソートします。

```
A>oc78k1 78k1.lnk -r

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

保守／廃止

(4) オブジェクト充塡指定　(－U)

記述形式　：－U　充塡値 [, [スタート], サイズ]
省略時解釈：なし

【機　　能】

－Uオプションは，HEX形式オブジェクトが出力されないアドレス領域に対して，指定した充塡値をオブジェクト・コードとして出力します。

【用　　途】

HEX形式オブジェクトが出力されていないアドレス領域には，不要なコードが書き込まれてしまうことがあります。このようなアドレスをプログラムが何らかの原因でアクセスした場合，プログラムの動作は予測できません。このため，あらかじめ，－Uオプションを指定してHEX形式オブジェクトが出力されていないアドレス領域にコードを書き込んでおきます。

【説　　明】

① 充塡値として指定できる値の範囲は，次のとおりです。

0H ≦ 充塡値 ≦ 0FFH

16進数字または10進数字で指定します。範囲外の数値または数値以外を指定した場合は，アボート・エラーとなります。

② スタートには，充塡を行うアドレス領域の先頭アドレスを指定します。
指定できる値の範囲は，次のとおりです。

0H ≦ スタート ≦ 0FF00H

16進数字または10進数字で指定します。範囲外の数値または数値以外を指定した場合には，アボート・エラーとなります。また，スタートを省略した場合には，0を指定したものとみなします。

保守／廃止

③　サイズには，充塡を行うアドレス領域のサイズを指定します。指定できる値の範囲は，次のとおりです。

0H　≦　サイズ　≦　0FF00H

16進数字または10進数字で指定します。範囲外の数値または数値以外を指定した場合には，アボート・エラーとなります。また，スタートを指定している場合には，サイズを省略することはできません。

④　－Uオプションを複数指定した場合は，最後に指定したものを優先します。

⑤　－Uオプションでの，スタート，サイズの指定形式とその解釈は次のようになります。

(a) －U　充塡値

：対象デバイスに内蔵ROMがある場合は，内蔵ROMの範囲

：対象デバイスに内蔵ROMのない場合は，アボート・エラー

(b) －U　充塡値，，サイズ

：0番地からサイズ－1番地まで

(c) －U　充塡値，スタート，サイズ

：スタート番地から（スタート＋サイズ－1）番地まで

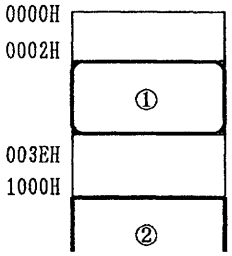
保守／廃止

【使 用 例】

例　HEX 形式オブジェクトが出力されていないアドレス領域にコードを充填します。
次のような HEX 形式オブジェクト・モジュール・ファイルがあると仮定します。この場合，003EH-0FFFH のアドレス領域にはコードが書き込まれていません。

```
:0200000080007E
:100080000BFCE0FE2BC40009C0FF00BC0028990057
:100090006480FCD350D25014FEB900059928AC00FE
:1000A0002431B900059928AC00242156AF0A8302F7
:0500B000A807A830566E
:01FE00001AE7
                ⋮
:00000001FF
```

①
②



この部分にコードを充填します。

●003EH-0FFFHのアドレス領域に、00Hを充填します。

```
A>oc78k1 78k1.ink -u00h,003eh,0fc2h

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

保守／廃止

(5) エラー・リスト・ファイル出力指定　(－E/－NE)

記述形式　：－E［出力ファイル名］
　　：－NE
省略時解釈：－NE

【機　　能】

① －Eオプションは，エラー・リスト・ファイルの出力を指定します。
　また，その出力先や出力ファイル名を指定します。
② －NEオプションは，－Eオプションを無効にします。

【用　　途】

エラー・リスト・ファイルの出力先や出力ファイル名を変更したいときに，－Eオプションを指定します。

【説　　明】

① ファイル名として，ディスク型ファイル名とデバイス型ファイル名を指定できます。ただし，デバイス型ファイル名としてCLOCKを指定した場合は，アボート・エラーとなります。
② －Eオプションを指定する際に出力ファイル名を省略するとエラー・リスト・ファイル名は，'入力ファイル名.EOC'となります。
③ －Eオプションを指定する際にドライブ名を省略するとカレント・ドライブにエラー・リスト・ファイルが出力されます。
④ －Eと－NEの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例　エラー・リスト・ファイル（78K1.EOC）を作成します。

```
A>oc78k1 78k1.lnk -e78k1.eoc

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

F200 Undefined symbol : CONVAH
Object Conversion Complete,      0 error(s) and      0 warning(s) found.
```

●78K1.EOCを参照します。

```
F200 Undefined symbol : CONVAH
```

保守／廃止

(6) パラメータ・ファイル指定　(－F)

記述形式　：－Ｆ　ファイル名

省略時解釈：起動行上からのみオプションまたは入力ファイル名の入力が可能

【機　　能】

－Ｆオプションは，オプションまたは入力ファイル名を指定のファイルから入力する指定をします。

【用　　途】

① コマンド行ではオブジェクト・コンバータの起動に必要な情報を指定しきれないときに，－Ｆオプションを指定します。

② オブジェクト・コンバートするたびに繰り返し同じようにオプションを指定する場合には，それらをパラメータ・ファイルに記述しておいて，－Ｆオプションで指定します。

【説　　明】

① 'ファイル名' として指定できるのは，ディスク型ファイル名のみです。
デバイス型ファイル名を指定するとアボート・エラーとなります。

② ファイル名を省略するとアボート・エラーとなります。

③ パラメータ・ファイルのネストは許されません。パラメータ・ファイル中で，－Ｆオプションを指定するとアボート・エラーとなります。

④ パラメータ・ファイル中に記述できる文字数の制限はありません。

⑤ 空白とタブおよび '↵' をオプションあるいは入力ファイル名の区切りとします。

⑥ パラメータ・ファイル中に記述したオプションまたは入力ファイル名は，コマンド行上のパラメータ・ファイル指定のあった位置に展開されます。

⑦ 展開されたオプションは，あとで指定したものを優先します。

⑧ ';' または '#' 以降に記述された文字は '↵' または EOF の前まですべてコメントと解釈します。

⑨ －Ｆオプションを複数指定するとアボート・エラーとなります。

【使 用 例】

例　パラメータ・ファイルを使用してオブジェクト・コンバートします。

●パラメータ・ファイル 78K1.POC の内容

```
;parameter file
78k1.lnk -osample.hex
-ssample.sym -r
```

●コマンド行には，次のように入力します。

```
A>oc78k1 -f78k1.poc

78K/I Series Object Converter Vx.xx  [ xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.
```

保守／廃止

(7) ヘルプ指定　(－－)

記述形式　：－－

省略時解釈：表示しない

【機　　能】

－－オプションは，ヘルプ・メッセージをディスプレイに表示します。

【用　　途】

ヘルプ・メッセージは，オブジェクト・コンバータ・オプションとその説明の一覧です。オブジェクト・コンバータを実行するときに参照してください。

【説　　明】

－－オプションを指定すると，他のオブジェクト・コンバータ・オプションは，すべて無効となります。

【使 用 例】

例　－－オプションを指定するとヘルプ・メッセージがディスプレイに出力されます。

```
A>oc78k1 --

78K/I Series Object Converter Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

usage : OC78K1 [option[ ...]] input-file [option[ ...]]
The option is as follows ([] means ommisible).
-Ffile           :Input option or input-file name from specified file.
-O[file]/-NO     :Create HEX module file [with specified name] / Not.
-S[file]/-NS     :Create symbol table file [with specified name] / Not.
-R/-NR           :Sort HEX object by address / Not.
-Uvalue[,[start],size] :Fill up HEX object with specified value.
--               :Show this message.
DEFAULT ASSIGNMENT: -O -S -NR
```

保守／廃止

保守／廃止

第7章　ライブラリアン

ライブラリアンは，RA78K/Iのオブジェクト・モジュール・ファイルまたはライブラリ・ファイルに対してモジュール単位で編集を行います。

さらに，リスト・ファイルを出力します。

ライブラリアン・エラーがある場合は，エラー・メッセージをディスプレイに出力し，エラーの原因を明示します。

保守／廃止

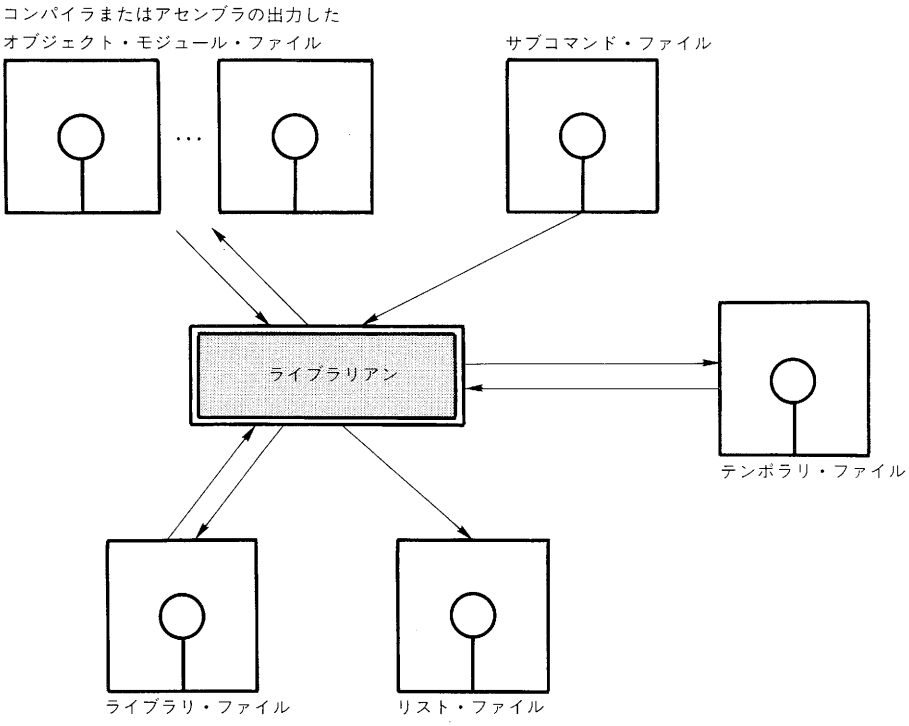
7.1　ライブラリアンの入出力ファイル

ライブラリアンの入出力ファイルを以下に示します。

表7-1　ライブラリアンの入出力ファイル

種類	ファイル名	説　　明	ディフォールト・ファイル・タイプ
入力ファイル	サブコマンド・ファイル	○実行プログラムのコマンドとパラメータを内容とするファイルです。 ○ユーザ作成ファイルです。	なし
出力ファイル	リスト・ファイル	○ライブラリ・ファイルの情報を出力した結果のファイルです。	.LST
入出力ファイル	オブジェクト・モジュール・ファイル	○アセンブラまたはコンパイラの出力したオブジェクト・モジュール・ファイルです。	.REL
	ライブラリ・ファイル	○ライブラリアンが出力したライブラリ・ファイルを入力し，内容を更新します。	.LIB
	テンポラリ・ファイル	○ライブラリ化のためにライブラリアンが自動生成するファイルです。ライブラリアンの実行終了時には消去されます。	LB78K1.$$x x＝1～6

図7-1　ライブラリアンの入出力ファイル



保守／廃止

7.2　ライブラリアンの機能

(1) モジュールのライブラリ化

アセンブラおよびリンカは，１個の出力モジュールを１個のファイルに作成します。

したがって，モジュールの数が多い場合，ファイルの数も増加します。このため，複数のモジュールを１個のファイルにまとめる機能を用意しました。これをモジュールのライブラリ化と呼び，ライブラリ化されたファイルをライブラリ・ファイルと呼びます。

ライブラリ・ファイルは，リンカに入力できます。したがって，モジュラ・プログラミングを行った場合，共通的なモジュールをライブラリ・ファイルとして作成しておけば，ファイル管理の面でも操作性の面においても効率がよくなります。

(2) ライブラリ・ファイルに対する編集

ライブラリアンには，ライブラリ・ファイルに対して次に示す編集機能があります。

① ライブラリ・ファイルへのモジュールの追加
② ライブラリ・ファイル内のモジュールの削除
③ ライブラリ・ファイル内のモジュールの置換
④ ライブラリ・ファイル内のモジュールの抽出

（機能の詳細については，“**7.5　サブコマンド**”をお読みください。）

(3) ライブラリ・ファイル情報の出力

ライブラリアンには，ライブラリ・ファイルの中に格納されている情報のうち，次に示すものを編集し，出力する機能があります。

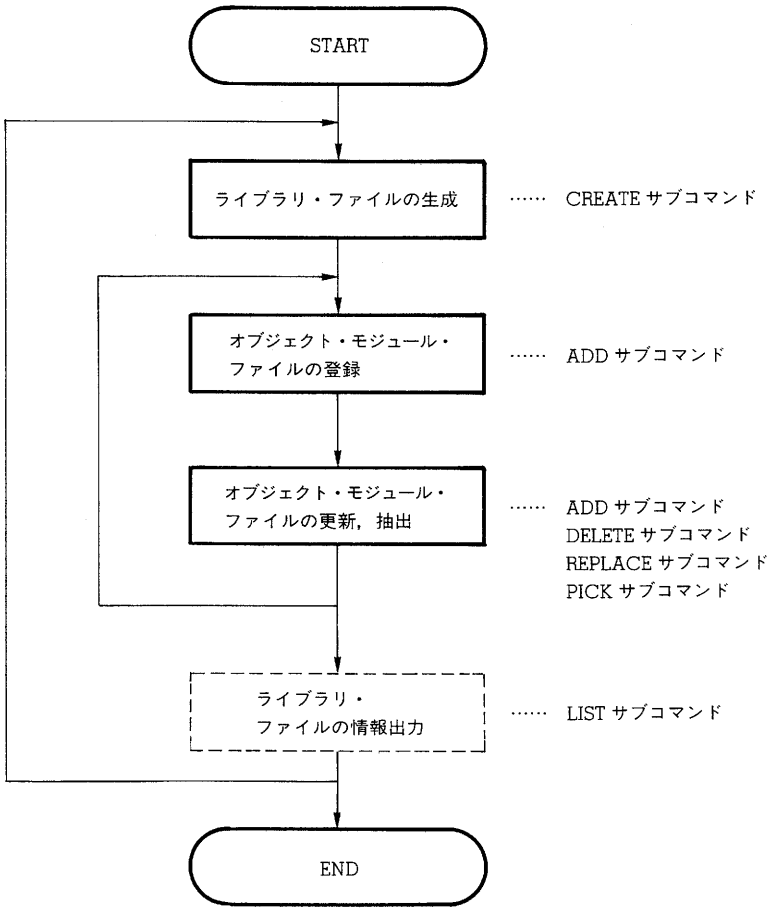
① モジュール名
② 作成プログラム
③ 登録日付
④ 更新日付
⑤ PUBLIC シンボル情報

注意　ライブラリアンでは，(2)，(3) で述べた機能をそれぞれサブコマンドとして提供しています。ライブラリアンは，各サブコマンドを順次解決しながら処理を行います。
サブコマンドの操作については，“7.5　サブコマンド”をお読みください。

(4) ライブラリ・ファイルの作成手順

一般的なライブラリ・ファイルの作成手順を以下に示します。

図７-２　ライブラリ・ファイルの作成手順



保守／廃止

7.3　ライブラリアンの起動方法

7.3.1　ライブラリアンの起動

ライブラリアンの二とおりの起動方法について説明します。

(1) コマンド行での起動

```
X>［パス名］　lb78k1　［△オプション］…
①　　②　　③　　④
```

① OSのプロンプト。

② ライブラリアンのコマンド・ファイルを格納しているパス名。

③ ライブラリアンのコマンド・ファイル名。

MS-DOS ver2.11の場合，すべてのプログラム・ファイルはカレント・ディレクトリになければなりません。

④ ライブラリアンに対して動作の詳細を指示します。

複数のライブラリアン・オプションを指定する場合は，おのおののライブラリアン・オプション間を空白で区切ります。

ライブラリアン・オプションの詳細については，“**7.4　ライブラリアン・オプション**”をお読みください。

例　A>lb78k1 -1120 -lw80

例　起動メッセージ

ライブラリアンが起動すると，ディスプレイに以下の起動メッセージが表示されます。

```
78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*
```

‘*’に続いて，ライブラリアンのサブコマンドを指定します。

```
*create 78k1.lib
*add 78k1.lib 78k1main.rel 78k1sub.rel
*exit
```

サブコマンドの入力を終了すると，各サブコマンドの処理を開始します。1つのサブコマンドの処理が完了すると再び‘*’がディスプレイに出力され，次のサブコマンドの入力待ち状態となります。終了サブコマンド（EXIT）が入力されるまで，この動作は繰り返されます。

```
＊サブコマンド指定
    サブコマンド処理
＊サブコマンド指定
    サブコマンド処理
        ⋮
＊サブコマンド（EXIT）指定
    ライブラリアンの終了
```

1行で指定可能な文字数は，128文字です。

1行に必要なオペランド情報をすべて入力できない場合は，‘&’を用いて継続行の指定ができます。なお，継続できる行数は15行です。

(2) サブコマンド・ファイルによる起動

サブコマンド・ファイルとはライブラリアンへのコマンドを格納しているファイルです。

ライブラリアンの起動時にサブコマンド・ファイルを指定しない場合，‘*’のあとに複数のサブコマンドを入力しなければなりません。しかし，サブコマンド・ファイルを作成することにより，これら複数のサブコマンドの処理が一度に行えます。

また，ライブラリ化のたびに同じサブコマンドを繰り返し指定することがあります。このような場合にサブコマンド・ファイルを使用してライブラリ化を行います。

サブコマンド・ファイルを使用する場合には，ファイル名の前に‘<’を記述します。

サブコマンド・ファイルによる起動方法を以下に示します。

```
X>lb78k1△　<サブコマンド・ファイル名　[△オプション] …
           ①          ②
```

① サブコマンド・ファイルを指定する場合は，必ず付加してください。

② サブコマンドが格納されているファイル

(a) サブコマンド・ファイルは，エディタなどで作成してください。

(b) サブコマンド・ファイル内での記述規則を以下に示します。

保守／廃止

```
サブコマンド名　オペランド情報
サブコマンド名　オペランド情報
                ⋮
EXIT
```

(c) 1つのサブコマンドが複数行にわたる場合，各行の行末に行の継続を示す'&'を記述します。

(d) ';'（セミコロン）から行末までは，ライブラリアンのコマンドとしては解釈されず，コメントとしてみなされます。

(e) サブコマンド・ファイルの最後のサブコマンドがEXITサブコマンドでない場合には，自動的にEXITサブコマンドがあったものと解釈されます。

(f) ライブラリアンは，サブコマンドをサブコマンド・ファイルから読み込んで処理を行います。サブコマンド・ファイル内のすべてのサブコマンドの処理を終了するとライブラリアンは終了します。

例　サブコマンド・ファイル（78K1.SLB）をエディタなどで作成します。

●78K1. SLBの内容

```
;
;library creation command
;
create 78k1.lib
;
add 78k1.lib 78k1main.rel &
78k1sub.rel
;
exit
```

●サブコマンド・ファイル（78K1.SLB）を使用してライブラリアンを起動します。

```
A>lb78k1 <78k1.slb
```

保守／廃止

7.3.2　実行開始，終了メッセージ

(1) 実行開始メッセージ

ライブラリアンが起動すると，ディスプレイに以下の実行開始メッセージが表示されます。

```
78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*program aborted
```

(2) 実行終了メッセージ

ライブラリアンは，実行終了メッセージを出力しません。各処理を終了したあとにユーザがEXITコマンドを入力することにより，ライブラリアンは制御をOSに戻します。

```
*create 78k1.lib
*add 78k1.lib 78k1main.rel 78k1sub.rel
*exit
```

ライブラリアンの処理継続が不可能な致命的エラーが検出された場合，ライブラリアンはメッセージをディスプレイに出力して，制御をOSに戻します。

例 1. ライブラリ・ファイルの先頭に‘<’を指定しなかった場合

```
A>lb78k1 78k1.slb

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
A003 Unrecognized string '78k1.slb'
Usage: LB78K1 [options]
```

この例では，ライブラリ・ファイルの先頭に‘<’を指定しなかったためにエラーとなり，ライブラリアンの実行が中止されました。

2. 存在しないライブラリアン・オプションを指定した場合

```
A>lb78k1 -a

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
A018 Option is not recognized '-a'
Usage: LB78K1 [options]
```

保守／廃止

この例では，存在しないライブラリアン・オプションを指定したためにエラーとなり，
ライブラリアンの実行が中止されました。

ライブラリアンがエラー・メッセージを出力してライブラリ化を中止した場合には，そのエラー・メッセージの原因を“**第11章　エラー・メッセージ**”で調べて対処してください。

保守／廃止

7.4　ライブラリアン・オプション

7.4.1　ライブラリアン・オプションの種類

ライブラリアン・オプションは，リスト・ファイルの形式やテンポラリ・ファイルの作成パスなどの指定を行います。ライブラリアン・オプションは，３種類のオプションに分類できます。

表7-2　ライブラリアン・オプション

項番	分　類	オプション	説　明
1	リスト・ファイル形式指定	－LW	リスト・ファイルの１行に印字する文字数を変更します。
		－LL	リスト・ファイルの１頁に印字する行数を変更します。
		－LF －NLF	リスト・ファイルの最後に改頁コードを付加します。
2	テンポラリ・ファイル作成パス指定	－T	テンポラリ・ファイルを指定したパスに作成します。
3	ヘルプ指定	－－	ディスプレイにヘルプ・メッセージを出力します。

備考　ライブラリアン・オプションの詳細については“**付録C.4　ライブラリアン・オプション一覧**”をお読みください。

保守／廃止

7.4.2　ライブラリアン・オプションの説明

以下に，各ライブラリアン・オプションの詳細を説明します。

(1) リスト・ファイル形式指定　(－LW，－LL，－LF/－NLF)

(a) －LW

記述形式　：－LW［文字数］

省略時解釈：－LW132（ディスプレイ出力の場合は80文字とします）

【機　　能】

－LW オプションは，リスト・ファイルの１行の文字数を指示します。

【用　　途】

リスト・ファイルの１行の文字数を変更したいときに，－LW オプションで指定します。

【説　　明】

① －LW オプションで指定できる文字数の範囲は次のとおりです（ディスプレイ出力の場合は，80文字までです）。

72　≦　1行に印字する文字数　≦　132

範囲外の数値や数値以外を指定した場合には，アボート・エラーとなります。

② 文字数を省略した場合は，132を指定したものとみなします。ただし，リスト・ファイルの出力先がディスプレイの場合は，80となります。

③ 指定する文字数には，ターミネータ（CR，LF）は含みません。

④ LIST サブコマンドが指定されない場合，－LW オプションは無視されます。

⑤ －LW オプションを複数指定した場合は，あとで指定した方を優先します。

【使 用 例】

リスト・ファイルの１行の文字数を80文字に指定します。

```
A>lb78k1 -lw80

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*create 78k1.lib 78k1sub.rel
*list 78k1.lib
```

保守／廃止

(b) －LL

記述形式　：－LL［行数］

省略時解釈：－LL66（ディスプレイ出力の場合は改頁しません）

【機　　能】

－LLオプションは，リスト・ファイルの1頁の行数を指定します。

【用　　途】

リスト・ファイルの1頁の行数を変更したいときに，－LLオプションを指定します。

【説　　明】

① －LLオプションで指定できる行数の範囲は次のとおりです。

20　≦　1頁に印字する行数　≦　32767

範囲外の数値や数値以外を指定した場合は，アボート・エラーとなります。

② 行数が省略された場合，66が指定されたものとみなします。

③ 行数の0を指定した場合は改頁しません。

④ LISTサブコマンドが指定されない場合，－LLオプションは無視されます。

⑤ －LLオプションを複数指定した場合は，あとで指定した方を優先します。

【使 用 例】

例　リスト・ファイルの1頁の行数を20行に指定します。

```
A>lb78k1 -l120

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*create 78k1.lib 78k1sub.rel
*list 78k1.lib
```

保守／廃止

(c) －LF/－NLF

記述形式　：－LF
　　－NLF
省略時解釈：－NLF

【機　　能】

① －LFオプションは，リスト・ファイルの最後に改頁コード（FF）を付加する指定をします。

② －NLFオプションは，－LFオプションを無効にします。

【用　　途】

リスト・ファイルの内容を印字したあとで改頁しておきたい場合に，－LFオプションを指定して改頁コードを付加します。

【説　　明】

① LISTサブコマンドが指定されない場合，－LFオプションは無視されます。

② －LFと－NLFの両オプションを同時に指定した場合は，あとで指定した方を優先します。

【使 用 例】

例　リスト・ファイルに改頁コードを付加します。

```
A>lb78k1 -lf

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*create 78k1.lib 78k1sub.rel
*list 78k1.lib
```

保守／廃止

(2) テンポラリ・ファイル作成パス指定　(-T)

記述形式　：－T　パス名

省略時解釈：環境変数 TMP により指定されたパスに作成します。
　　指定されていない場合は，カレント・パスに作成します。

【機　　能】

－Tオプションは，テンポラリ・ファイルを作成するパスを指定します。

【用　　途】

テンポラリ・ファイルの作成場所を指定できます。

【説　　明】

① パス名としてパス以外のものは指定できません。

② パス名は省略できません。

③ 以前に作成されたテンポラリ・ファイルが存在している場合でも，ファイル保護をしていなければ，上書きします。

④ 必要とするメモリ・サイズがある間は，テンポラリ・ファイルをメモリに展開します。なお，メモリが足りなくなった時点で，メモリに展開していたテンポラリ・ファイルの内容をディスクに書き出します。以降のテンポラリ・ファイルへのアクセスは，セーブしたディスク・ファイルに対して行います。

⑤ テンポラリ・ファイルは，ライブラリ化終了時に削除されます。また，キー入力(CTRL-C) によってライブラリ化が中止されたときも，削除されます。

⑥ テンポラリ・ファイルの作成パスは，次の順番で決定されます。

1. －Tオプションで指定されたパス
2. 環境変数 TMP に設定されているパス（－Tオプション省略の場合）
3. カレント・パス（TMP が設定されていない場合）

なお，1.または2.を指定した場合，指定されたパスにテンポラリ・ファイルが作成できなければアボート・エラーとなります。

保守／廃止

【使 用 例】

例　テンポラリ・ファイルをディレクトリ TMP に出力するよう指定します。

```
A>lb78k1 -t¥tmp

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*
```

(3) ヘルプ指定　(－－)

記述形式　：－－

省略時形式：表示しない

【機　　能】

－－オプションは，ヘルプ・メッセージをディスプレイに表示します。

【用　　途】

ヘルプ・メッセージは，サブコマンドとその説明の一覧です。ライブラリアンを実行するときに参照してください。

【説　　明】

－－オプションを指定すると他のライブラリアン・オプションは，すべて無効となります。

【使 用 例】

例　－－オプションを指定するとヘルプ・メッセージがディスプレイに出力されます。

```
A>lb78k1 --

78K/I Series Librarian Vx.xx  [xx xxx xx]
  Copyright (C) NEC Corporation 1990
+-------------------------------------------------------------------------+
| Subcommands : create, add, delete, replace, pick, list, help, exit      |
|                                                                         |
|   Usage : subcommand[ option] masterLBF[ option] transaction[ option]   |
|                                                                         |
|              transaction :== OMFname                                    |
|                               LBFname[(modulename[,...])]               |
|                                                                         |
|   <create > : create masterLBF[ transaction]                            |
|   <add    > : add masterLBF transaction                                 |
|   <delete > : delete masterLBF(modulename[,...])                        |
|   <replace> : replace masterLBF transaction                             |
|   <pick   > : pick masterLBF (modulename[,...])                         |
|   <list   > : list[ option] masterLBF[(modulename[,...])                |
|                   option : -P  = output public symbol                   |
|                            -NP = no output public symbol                |
|                            -O filename = specify output file name       |
|   <help   > : help                                                      |
|   <exit   > : exit                                                      |
|                                                                         |
+-------------------------------------------------------------------------+
```

保守／廃止

7.5　サブコマンド

7.5.1　サブコマンドの種類

サブコマンドは，ライブラリアンの動作に細かい指示を与えます。サブコマンドは，８種類に分類できます。

表7-3　サブコマンド

項番	サブコマンド名	短縮形	説　明
1	CREATE	C	ライブラリ・ファイルを新規に作成します。
2	ADD	A	ライブラリ・ファイルにモジュールを追加します。
3	DELETE	D	ライブラリ・ファイル内のモジュールを削除します。
4	REPLACE	R	ライブラリ・ファイル内のモジュールを他のモジュールと置き換えます。
5	PICK	P	ライブラリ・ファイル内のモジュールを取り出します。
6	LIST	L	ライブラリ・ファイル内のモジュール情報を出力します。
7	HELP	H	ディスプレイにヘルプ・メッセージを出力します。
8	EXIT	E	ライブラリアンを終了します。

備考　サブコマンドの詳細については，“**付録Ｄ　サブコマンド一覧**”をお読みください。

7.5.2　サブコマンドの説明

各サブコマンドの詳細について説明します。

●コマンド・ファイルの一般形式

＊サブコマンド［△オプション］△ライブラリ・ファイル名［△オプション］　トランザクション［△オプション］
　　①　　②

①　直前に指定されたライブラリ・ファイル名は‘.’で置き換えできます。

②　トランザクション　＝　△オブジェクト・モジュール・ファイル名
　　△ライブラリ・ファイル名［▲(▲モジュール名［▲，…])］

保守／廃止

(1) CREATE

記述形式：CREATE△ライブラリ・ファイル名［△トランザクション］

短 縮 形：C

【機　　能】

CREATE サブコマンドは，ライブラリ・ファイルを新規に作成します。

【説　　明】

① 作成されたライブラリ・ファイルのサイズは 0 となります。

② トランザクションを指定した場合は，ライブラリ・ファイルの作成と同時にモジュールを登録します。

③ ライブラリ・ファイル名：

指定したファイルがすでに存在している場合には，上書きします。

④ トランザクション：

ライブラリ・ファイル中に PUBLIC シンボルと同一の PUBLIC シンボルを持つオブジェクト・モジュール・ファイルは，登録できません。

またライブラリ・ファイル中にあるモジュールと同一の名前のモジュールは登録できません。

⑤ エラーが発生した場合は処理を中断し，ライブラリ・ファイルは作成されません。

【使 用 例】

例 1. 新規にライブラリ・ファイル（78K1.LIB）を作成します。

```
*create 78k1.lib
```

＜作成前＞

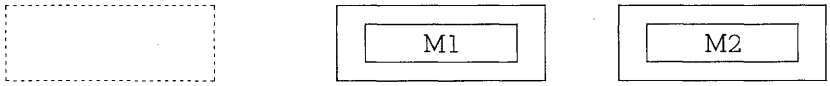
＜作成後＞ 78K1.LIB

保守／廃止

例 2. ライブラリ・ファイルを作成し同時にモジュール M1 と M2 を登録します。

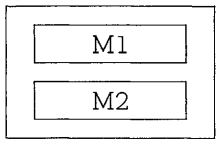
*create 78k1.lib m1.rel m2.rel

＜作成前＞



＜作成後＞

78K1.LIB



保守／廃止

(2) ADD

記述形式：ADD△ライブラリ・ファイル名△トランザクション

短 縮 形：A

【機　　能】

既存のライブラリ・ファイルに対してモジュールを追加します。

【説　　明】

① 追加するライブラリ・ファイル中には，モジュールが存在していなくてもかまいません。

② 追加するモジュールと同名のモジュールがライブラリ・ファイル内に存在する場合，エラーとなります。

③ 追加するモジュール中の PUBLIC シンボルがライブラリ・ファイル内に存在する場合，エラーとなります。

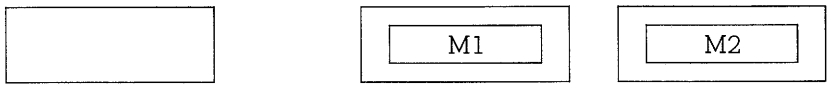
【使 用 例】

例 1. ライブラリ・ファイル（78K1.LIB）にモジュール（M1，M2）を追加します。

```
*add 78k1.lib m1.rel m2.rel
```

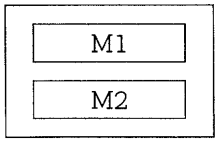
＜追加前＞

78K1.LIB



＜追加後＞

78K1.LIB



保守／廃止

例 2. ライブラリ・ファイル（78K1.LIB）にモジュール（M3）を追加します。

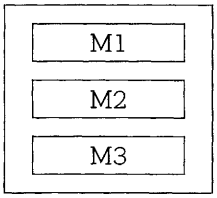
```
*add 78k1.lib m3.rel
```

＜追加前＞

78K1.LIB

＜追加後＞

78K1.LIB



保守／廃止

(3) DELETE

記述形式：DELETE△ライブラリ・ファイル名▲(▲モジュール名［▲，…］▲)

短 縮 形：D

【機　　能】

既存のライブラリ・ファイルからモジュールを削除します。

【説　　明】

① 指定したモジュールがライブラリ・ファイルに存在しない場合，エラーとなります。

② エラーが発生した場合は処理を中断し，ライブラリ・ファイルの状態は変化しません。

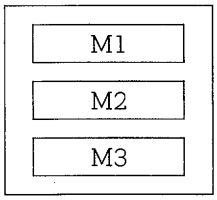
【使 用 例】

例 ライブラリ・ファイル（78K1.LIB）からモジュール（M1，M3）を削除します。

```
*delete 78k1.lib  (m1.rel m3.rel)
```

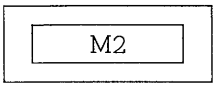
＜削除前＞

78K1.LIB



＜削除後＞

78K1.LIB



保守／廃止

(4) REPLACE

記述形式：REPLACE△ライブラリ・ファイル名△トランザクション

短 縮 形：R

【機　　能】

既存のライブラリ・ファイルのモジュールを，他のオブジェクト・モジュール・ファイルのモジュールと置き換えます。

【説　　明】

① 置換するモジュール名と同名のモジュールが，更新するライブラリ・ファイル内に存在しない場合はエラーとなります。

② 置換するモジュール中の PUBLIC シンボルが，ライブラリ・ファイル内に存在する場合はエラーとなります。

③ 置換するオブジェクト・モジュール・ファイル名は，登録時と同じファイル名でなければなりません。

④ エラーが発生した場合は処理を中断し，ライブラリ・ファイルの状態は変化しません。

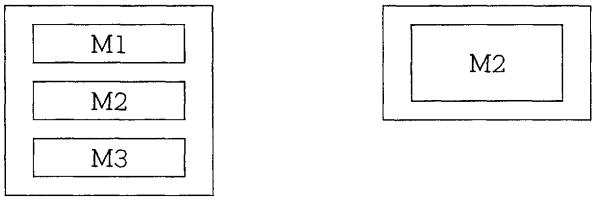
【使 用 例】

例　ライブラリ・ファイル（78K1.LIB）中のモジュール（M2）を置換します。

```
*replace 78k1.lib m2.rel
```

＜置換前＞

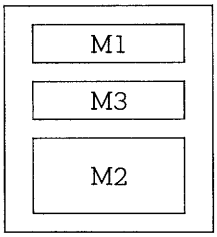
78K1.LIB



保守／廃止

＜置換後＞

78K1.LIB



ライブラリ・ファイル中のモジュール（M2）を削除したあと，新たにモジュール（M2）を登録するため，ライブラリ・ファイル中のモジュール（M2）の順序は最後となります。

保守／廃止

(5) PICK

記述形式：PICK△ライブラリ・ファイル名▲（▲モジュール名［▲，…］▲）

短 縮 形：P

【機　　能】

既存のライブラリ・ファイルのモジュールから指定したモジュールを取り出します。

【説　　明】

① 取り出したモジュールは，登録時のファイル名を持つオブジェクト・モジュール・ファイルとなります。

② 指定したモジュール名が，ライブラリ・ファイル内に存在しない場合はエラーとなります。

③ エラーの場合は処理を中断します。ただし，複数のモジュールが指定されているときにエラーが発生した場合には，エラーとなったモジュールの直前までに取り出されたモジュールは有効となりディスク上に保存されます。

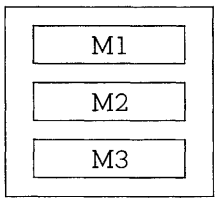
【使 用 例】

例　ライブラリ・ファイル（78K1.LIB）中のモジュール（M2）を取り出します。

```
*pick 78k1.lib  (m2.rel)
```

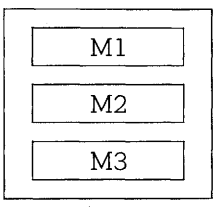
＜抽出前＞

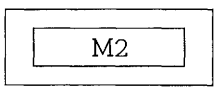
78K1.LIB



＜抽出後＞

78K1. LIB





保守／廃止

(6) LIST

記述形式：LIST [△オプション] △ライブラリ・ファイル名 [▲ (▲モジュール名 [▲, …] ▲)]

●オプション：－ PUBLIC/－ NOPUBLIC
　　：－ ○▲ファイル名

短 縮 形：L

【機　　能】

ライブラリ・ファイル内のモジュール情報を出力します。

【説　　明】

① オプションは，複数指定できます。

② －○：
出力ファイル名には，デバイス型ファイル名を指定できます。
出力ファイル名を省略した場合は，エラーとなります。
ファイル・タイプを省略した場合は，'入力ファイル名.LST' が入力されたものとします。

③ － PUBLIC/－ NOPUBLIC：
下線部のみの指定も可能です。
－ PUBLIC は，パブリック・シンボル情報の出力を指示します。
－ NOPUBLIC は，－ PUBLIC を無効にします。
－ PUBLIC と － NOPUBLIC の両方を指定した場合は，あとで指定した方を優先します。

保守／廃止

【使 用 例】

例　ライブラリ・ファイル（78K1.LIB）のモジュール情報をリスト・ファイル（78K1.LST）に出力します。この際，パブリック・シンボル情報が出力されるように，－Pオプションを指定します。

```
*list -p -o78k1.lst  (78k1.lib)
```

●リスト・ファイル（78K1.LST）を参照します。

```
A>type 78k1.lst
78K/I Series librarian Vx.xx    DATE : xx xxx xx                    PAGE   1
LIB-FILE NAME : 78K1.LIB        (xx xxx xx)
0001  78K1MAIN.REL    (xx xxx xx)
  MAIN                          START
  NUMBER OF PUBLIC SYMBOLS :   2
0002  78K1SUB.REL     (xx xxx xx)
  CONVAH
  NUMBER OF PUBLIC SYMBOLS :   1
```

保守／廃止

(7) HELP

記述形式：HELP

短 縮 形：H

【機　　能】

ヘルプ・メッセージをディスプレイに出力します。

【説　　明】

ヘルプ・メッセージは，サブコマンドとその説明の一覧です。HELP コマンドまたは，－－オプションを指定してライブラリアンを実行するときに参照してください。

【使 用 例】

例　HELP コマンドを指定するとヘルプ・メッセージが出力されます。

```
A>lb78k1

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*;
*help

+-------------------------------------------------------------------------+
| Subcommands : create, add, delete, replace, pick, list, help, exit      |
|                                                                         |
|   Usage : subcommand[ option] masterLBF[ option] transaction[ option]   |
|                                                                         |
|              transaction :== OMFname                                    |
|                              LBFname[(modulename[,...])]                |
|                                                                         |
|   <create > : create masterLBF[ transaction]                            |
|   <add    > : add masterLBF transaction                                 |
|   <delete > : delete masterLBF(modulename[,...])                        |
|   <replace> : replace masterLBF transaction                             |
|   <pick   > : pick masterLBF (modulename[,...])                         |
|   <list   > : list[ option] masterLBF[(modulename[,...])                |
|                  option : -P  = output public symbol                    |
|                           -NP = no output public symbol                 |
|                           -O filename = specify output file name        |
|   <help   > : help                                                      |
|   <exit   > : exit                                                      |
|                                                                         |
+-------------------------------------------------------------------------+
*
```

保守／廃止

(8) EXIT

記述形式：EXIT

短 縮 形：E

【機　　能】

ライブラリアンを終了します。

【説　　明】

ライブラリアンを終了するときに使用します。

【使 用 例】

例　ライブラリアンを終了します。

```
*exit
```

保守／廃止

保守／廃止

第8章　リスト・コンバータ

リスト・コンバータは，アセンブラが出力するアセンブル・リスト・ファイル，オブジェクト・モジュール・ファイルと，リンカが出力するロード・モジュール・ファイルを入力します。

そして，入力ファイル中のリロケータブルなアドレスやシンボルに実際のアドレスを埋め込んで，アブソリュート・アセンブル・リスト・ファイルとして出力します。こうすることによって，リンク・マップを参照しながらアセンブル・リストを見るというわずらわしさが軽減されます。

保守／廃止

8.1　リスト・コンバータの入出力ファイル

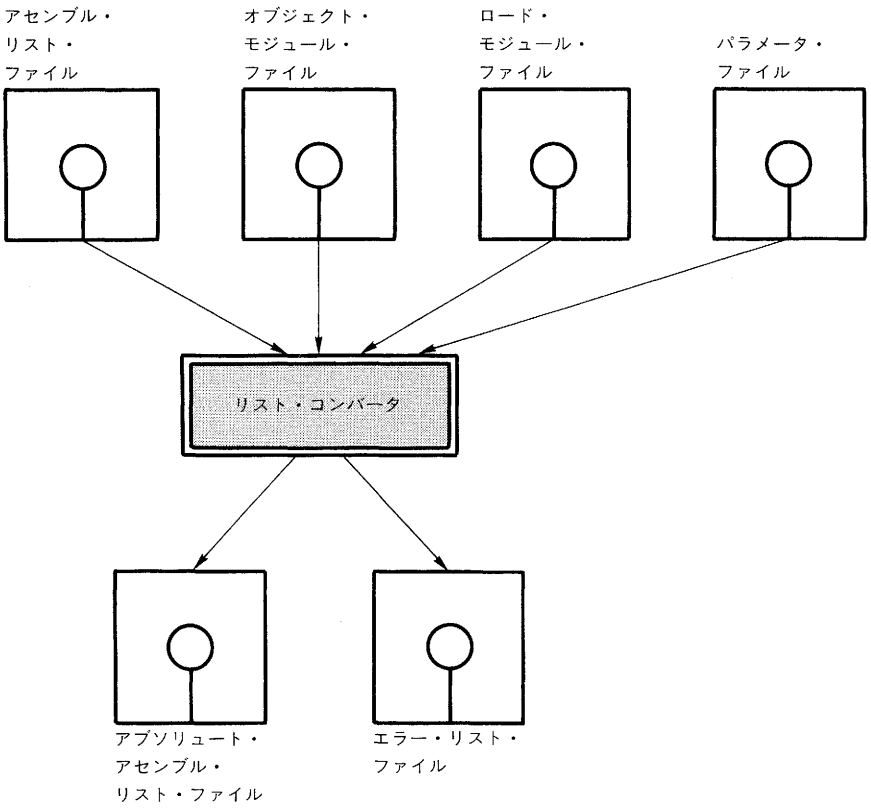
リスト・コンバータの入出力ファイルを以下に示します。

表8-1　リスト・コンバータの入出力ファイル

種類	ファイル名	説　明	ディフォールト・ファイル・タイプ
入力ファイル	オブジェクト・モジュール・ファイル	○機械語情報と機械語の配置アドレスに関する再配置情報およびシンボル情報を含んだバイナリ・ファイルです。	.REL
	アセンブル・リスト・ファイル	○アセンブル・リスト，クロスレファレンス・リストなどのアセンブル情報を持つファイルです。	.PRN
	ロード・モジュール・ファイル	○リンク結果のオブジェクト・コードのバイナリ・イメージ・ファイルです。	.LNK
	パラメータ・ファイル	○実行プログラムのパラメータを内容とするファイルです。 ○ユーザ作成ファイルです。	.PLV
出力ファイル	アブソリュート・アセンブル・リスト・ファイル	○入力ファイル中のリロケータブルなアドレスやシンボルに実際のアドレスを埋め込んだリスト・ファイルです。	.P
	エラー・リスト・ファイル	○リスト・コンバート時のエラー情報を持つファイルです。	.ELV

保守／廃止

図8-1　リスト・コンバータの入出力ファイル



保守／廃止

8.2　リスト・コンバータの機能

リロケータブル・アセンブラをアブソリュート・アセンブラと比較した場合の長所，短所を以下に示します。

［長所］

① 複数の人数で開発可能。
② モジュール分割により開発，保守が容易。
③ ライブラリ管理が容易。
④ 大規模プログラムの開発に適している。

［短所］

① アセンブル・リストのアドレスが実際の物理アドレスと一致しない。
② 外部シンボルの値がアセンブル・リスト中では０になっており，実際の値は，リンク・マップを参照しなければならない。
③ アセンブル・リスト中のリロケータブルな値は，実際の値と異なる。

上記の短所は，特にディバグ時や保守のためのドキュメント面で生産性の低下を招きます。
リスト・コンバータは，リロケータブル・アセンブラ・パッケージの上記の欠点を解決できます。

① リスト・コンバータの出力したアブソリュート・アセンブル・リストは，動作時のアドレスと完全に一致しています。
② 外部シンボルの実際の値がリスト上に埋め込まれます。
③ リロケータブルな値がリスト上に実際の値として埋め込まれます。
④ シンボル・テーブルあるいはクロスレファレンス・リスト上のシンボル値に対しても，実際の値が埋め込まれます。

例 1. ロケーションの埋め込み

●アセンブル・リスト

```
16    16   0000                    STASC:  DS      2
17    17
18    18   ----                            CSEG
19    19   ----                            ORG     0H
20    20   0000   R0000            MAIN:   DW      START
21    21
22    22   ----                            CSEG
23    23   0000    0BFCE0FE        START:  MOVW    SP, #0FEE0H
24    24   0004    2BC400                  MOV     MM, #00
25    25   0007    09C0FF00                MOV     STBC, #00
```

●アブソリュート・アセンブル・リスト

```
16    16   FC80                    STASC:  DS      2
17    17
18    18   ----                            CSEG
19    19   ----                            ORG     0H
20    20   0000   R8000            MAIN:   DW      START
21    21
22    22   ----                            CSEG
23    23   0080    0BFCE0FE        START:  MOVW    SP, #0FEE0H
24    24   0084    2BC400                  MOV     MM, #00
25    25   0087    09C0FF00                MOV     STBC, #00
```

2. オブジェクト・コードの埋め込み

●アセンブル・リスト

```
27    27 000B    BC00                    MOV     E, #LOW HDTSA
28    28

29    29 000D    R280000                 CALL    !CONVAH
30    30
31    31 0010    R640000                 MOVW    DE, #STASC

32    32 0013    D3                      MOV     A, B
33    33 0014    50                      MOV     [DE+], A
34    34 0015    D2                      MOV     A, C
35    35 0016    50                      MOV     [DE+], A
```

●アブソリュート・アセンブル・リスト

```
27    27 008B     BC00              MOV     E,#LOW HDTSA
28    28
29    29 008D     R289900           CALL    !CONVAH
30    30
31    31 0090     R6480FC           MOVW    DE,#STASC
32    32 0093     D3                MOV     A,B
33    33 0094     50                MOV     [DE+],A
34    34 0095     D2                MOV     A,C
35    35 0096     50                MOV     [DE+],A
```

保守／廃止

8.3　リスト・コンバータの起動方法

8.3.1　リスト・コンバータの起動

リスト・コンバータの2種類の起動方法を以下に示します。

(1) コマンド行での起動

```
X>  [パス名] lcnv78k1 [△オプション] … △入力ファイル名 [△オプション] … [△]
①     ②        ③          ④                ⑤            ④
```

① OSのプロンプト。

② リスト・コンバータのコマンド・ファイルを格納しているパス名。

③ リスト・コンバータのコマンド・ファイル名。

MS-DOS ver2.11の場合，すべてのプログラム・ファイルはカレント・ディレクトリになければなりません。

④ リスト・コンバータに対して動作の詳細を指示します。

複数のリスト・コンバータ・オプションを指定する場合には，それぞれのリスト・コンバータ・オプション間を空白で区切ってください。なお，リスト・コンバータ・オプションの詳細については，“**8.4　リスト・コンバータ・オプション**”をお読みください。

⑤ アセンブル・リスト・ファイル名のプライマリ・ネームです（このファイルの拡張子は‘.PRN’にしてください）。

例　A＞lcnc78k1 78k1main -178k1.lnk

注意　上記⑤で，コマンド行にアセンブル・リストのプライマリ・ネームのみを指定する場合は，オブジェクト・モジュール・ファイル，ロード・モジュール・ファイルのプライマリ・ネームは，アセンブル・リスト・ファイルのプライマリ・ネームと同一でなければなりません。また，ファイル・タイプは次のものでなければなりません。

ファイル名	ファイル・タイプ
オブジェクト・モジュール・ファイル	.REL
ロード・モジュール・ファイル	.LNK

プライマリ・ネームの異なるファイルを指定する場合は，オプションを使用してください。

保守／廃止

(2) パラメータ・ファイルによる起動

パラメータ・ファイルは，起動に必要な情報がコマンド行に指定しきれない場合や，リスト・コンバートするたびに同じオプションを繰り返し指定するような場合に使用します。

パラメータ・ファイルを使用する場合には，コマンド行にパラメータ・ファイル指定オプション（－F）を指定します。

パラメータ・ファイルによる起動方法は，次のようになります。

X＞lcnv78k1［△入力ファイル名］△ －f パラメータ・ファイル名
　　①　　②

① パラメータ・ファイル指定オプション

② リスト・コンバータの起動に必要な情報を含んだファイル

備考　パラメータ・ファイルはエディタなどで作成してください。

パラメータ・ファイルでの生成規則を次に示します。

［［［△］オプション［△オプション］…［△］△］］…

① コマンド行で入力ファイル名を省略した場合，パラメータ・ファイル内に入力ファイル名を記述します。

② 入力ファイル名は，オプションのあとにも記述できます。

③ パラメータ・ファイルには，コマンド行で指定すべきすべてのリスト・コンバータ・オプション，出力ファイル名を記述します。

例　パラメータ・ファイル（78K1.PLV）をエディタで作成します。

●78K1.PLV の内容

```
: parameter file
78k1main -l78k1.lnk
-e78k1.elv
```

●パラメータ・ファイル（78K1.PLV）を使用してリスト・コンバータを起動します。

```
A＞lcnv78k1 -f78k1.plv
```

保守／廃止

8.3.2　実行開始，終了メッセージ

(1) 実行開始メッセージ

リスト・コンバータが起動すると，ディスプレイに実行開始メッセージが表示されます。

```
List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

(2) 実行終了メッセージ

リスト・コンバートの結果，リスト・コンバート・エラーが検出されなかった場合，リスト・コンバータは次のメッセージをディスプレイに出力して制御をOSに戻します。

```
Conversion complete.
```

リスト・コンバート中に，リスト・コンバータ処理継続が不可能な致命的エラーが検出された場合，リスト・コンバータはメッセージをディスプレイに出力して処理を中止し，制御をOSに戻します。

例 1. 存在しないアセンブル・リスト・ファイルを指定した場合。

```
A>lcnv78k1 sample

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

A006 File not found '-lSAMPLE.LNK'
Program aborted.
```

上記の例では，存在しないアセンブル・リスト・ファイルを指定したためにエラーとなり，処理を中止しました。

保守／廃止

例 2. 存在しないリスト・コンバータ・オプションを指定した場合。

```
A>lcnv78k1 78k1main -a

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

A018 Option is not recognized '-a'
Program aborted.
```

上記の例では，存在しないリスト・コンバータ・オプションを指定したためにエラーとなり，処理を中止しました。

リスト・コンバータがエラー・メッセージを出力して処理を中止した場合は，そのエラー・メッセージの原因を“**第11章　エラー・メッセージ**”で調べて対処してください。

保守／廃止

8.4　リスト・コンバータ・オプション

8.4.1　リスト・コンバータ・オプションの種類

リスト・コンバータ・オプションは，リスト・コンバータの動作に細かい指示を与えるものです。リスト・コンバータ・オプションは，６種類のオプションに分類できます。

表８-２　リスト・コンバータ・オプション

項番	分　類	オプション	説　明
1	オブジェクト・モジュール・ファイル入力指定	－R	オブジェクト・モジュール・ファイルを入力します。
2	ロード・モジュール・ファイル入力指定	－L	ロード・モジュール・ファイルを入力します。
3	アブソリュート・アセンブル・リスト・ファイル出力指定	－O	アブソリュート・アセンブル・リスト・ファイルを出力します。
4	エラー・リスト・ファイル出力指定	－E －NE	エラー・リスト・ファイルを出力します。
5	パラメータ・ファイル指定	－F	入力ファイル名，オプションを指定したファイルより入力します。
6	ヘルプ指定	－－	ディスプレイにヘルプ・メッセージを表示します。

以上の表は，リスト・コンバータ・オプションを紹介するための表です。実際に，リスト・コンバータ・オプションを使用する場合には，“**付録C.5　リスト・コンバータ・オプション一覧**”をお読みください。

保守／廃止

8.4.2　リスト・コンバータ・オプションの説明

以下に，各リスト・コンバータ・オプションの詳細を説明します。

(1) オブジェクト・モジュール・ファイル入力指定　(－R)

記述形式　：－R入力ファイル名

省略時解釈：－Rアセンブル・リスト・ファイル名 .REL

【機　　能】

－Rオプションは，入力するオブジェクト・モジュール・ファイルを指定します。

【用　　途】

オブジェクト・モジュール・ファイルのプライマリ・ネームが，アセンブル・リスト・ファイルのプライマリ・ネームと異なる場合，またはファイル・タイプが“.REL”でない場合は，－Rオプションを指定します。

【説　　明】

① フェイタル・エラーがある場合は，アブソリュート・アセンブル・リスト・ファイルは出力されません。

② 入力ファイル名のプライマリ・ネームのみを指定した場合は，ファイル・タイプとして‘.REL’を付加してファイルを入力します。

【使 用 例】

例　アセンブル・リスト・ファイル名が78K1MAIN.PRNで，オブジェクト・モジュール・ファイル名がSAMPLE.RELの場合

```
A>lcnv78k1 78k1main -rsample.rel

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

保守／廃止

(2) ロード・モジュール・ファイル入力指定　(－L)

記述形式　：－L入力ファイル名

省略時解釈：－Lアセンブル・リスト・ファイル名 .LNK

【機　　能】

－Lオプションは，入力するロード・モジュール・ファイルを指定します。

【用　　途】

ロード・モジュール・ファイルのプライマリ・ネームがアセンブル・リスト・ファイルのプライマリ・ネームと異なる場合またはファイル・タイプが“.LNK”でない場合に，－Lオプションを指定します。

【説　　明】

① フェイタル・エラーがある場合は，アブソリュート・アセンブル・リスト・ファイルは出力されません。

② 入力ファイル名のプライマリ・ネームのみを指定した場合は，ファイル・タイプとして‘.LNK’を付加してファイルを入力します。

8

【使 用 例】

例　アセンブル・リスト・ファイル名が78K1MAIN.PRNで，ロード・モジュール・ファイル名がSAMPLE.LNKの場合

```
A>lcnv78k1 78k1main -lsample.lnk

List Conversion Program for RA78K/1 Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

保守／廃止

(3) アブソリュート・アセンブル・リスト・ファイル出力指定　(－O)

記述形式　：－O出力ファイル名

省略時解釈：－Oアセンブル・リスト・ファイル名.P

【機　　能】

－Oオプションは，アブソリュート・アセンブル・リスト・ファイルの出力を指定します。また，その出力先や出力ファイル名も指定できます。

【用　　途】

アブソリュート・アセンブル・リスト・ファイルの出力先や出力ファイル名を変更したいときに，－Oオプションを指定します。

【説　　明】

① ファイル名として，ディスク型ファイル名とデバイス型ファイル名を指定できます。指定できるデバイス型ファイル名はCON，PRN，NULおよびAUXです。CLOCKを指定した場合，アボート・エラーとなります。

② ファイル名にエラー・ファイルと同一のデバイスが指定された場合，アボート・エラーとなります。

③ －Oオプションを指定する際に出力ファイル名を省略するとアブソリュート・アセンブル・リスト・ファイル名は，'アセンブル・リスト・ファイル名.P'となります。

④ 出力ファイル名のプライマリ・ネームのみを指定した場合は，ファイル・タイプとして'.P'を付加してファイルを出力します。

⑤ －Oオプションを指定する際にドライブ名を省略すると，カレント・ドライブにアブソリュート・アセンブル・リスト・ファイルが出力されます。

【使 用 例】

例 1. アブソリュート・アセンブル・リスト・ファイル（SAMPLE.P）を作成します。

```
A>lcnv78k1 78k1main -osample.p

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

保守／廃止

例 2. アブソリュート・アセンブル・リスト・ファイルをプリンタに出力します。

```
A>lcnv78k1 78k1main -oprn

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

保守／廃止

(4) エラー・リスト・ファイル出力指定　(－E/－NE)

記述形式　：－E［出力ファイル名］
　　－NE
省略時解釈：－NE

【機　　能】

① －Eオプションは，エラー・リスト・ファイルの出力を指定します。また，その出力先や出力ファイル名も指定できます。

② －NEオプションは，－Eオプションを無効にします。

【用　　途】

エラー・メッセージをファイルに保存しておきたい場合には，－Eオプションを指定します。

【説　　明】

① ファイル名として，ディスク型ファイル名とデバイス型ファイル名を指定できます。ただし，デバイス型ファイル名として，CLOCKを指定した場合は，アボート・エラーとなります。

② ファイル名にアブソリュート・アセンブル・リスト・ファイルと同一のデバイスが指定された場合，アボート・エラーとなります。

③ －Eオプションを指定する際に出力ファイル名を省略するとエラー・リスト・ファイル名は，'アセンブル・リスト・ファイル名.ELV'となります。

④ 出力ファイル名のプライマリ・ネームのみを指定した場合は，ファイル・タイプとして'.ELV'を付加してファイルを出力します。

⑤ －Eオプションを指定する際にドライブ名を省略するとカレント・ドライブにエラー・リスト・ファイルが出力されます。

⑥ －Eと－NEオプションを同時に指定した場合は，あとで指定した方を優先します。

保守／廃止

【使 用 例】

例　エラー・リスト・ファイル（SAMPLE.ELV）を作成します。

```
A>lcnv78k1 78k1main -esample.elv

List Conversion Program for RA78K/1 Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

●エラー・リスト・ファイル（SAMPLE.ELV）を参照します。

```
Pass1: start
***WARNING W101 Load module file is older than object module file '78K1MAIN.REL'
***WARNING W102 Load module file is older than assemble list file '78K1MAIN.LNK,
78K1MAIN.PRN'

Pass2: start
```

保守／廃止

(5) パラメータ・ファイル指定　(－F)

記述形式　：－Fファイル名

省略時解釈：起動行上からのみオプション，入力ファイル名の入力が可能

【機　　能】

－Fオプションは，オプションあるいは入力ファイル名を指定のファイルから入力する指定をします。

【用　　途】

① コマンド行ではリスト・コンバータの起動に必要な情報を指定しきれないときに，－Fオプションを指定します。

② リスト・コンバートするたびに繰り返し同じようにオプションを指定し，リスト・コンバートする場合には，それらをパラメータ・ファイルに記述しておき，－Fオプションを指定します。

【説　　明】

① 'ファイル名'として指定できるのは，ディスク型ファイル名のみです。デバイス型ファイル名を指定するとアボート・エラーとなります。

② ファイル名を省略するとアボート・エラーとなります。

③ ファイル名のプライマリ・ネームのみを指定した場合は，ファイル・タイプとして'.PLV'を付加してファイルをオープンします。

④ パラメータ・ファイルのネストは許されません。パラメータ・ファイル中で，－Fオプションを指定するとアボート・エラーとなります。

⑤ パラメータ・ファイル中に記述できる文字数の制限はありません。

⑥ 空白とタブおよび'↵'をオプションあるいは入力ファイル名の区切りとします。

⑦ パラメータ・ファイル中に記述したオプションあるいは入力ファイル名はコマンド行上のパラメータ・ファイル指定のあった位置に展開されます。

⑧ 展開されたオプションは，あとで指定したものが優先です。

⑨ －Fオプションを複数指定するとアボート・エラーとなります。

保守／廃止

【使 用 例】

例　パラメータ・ファイルを使用してリスト・コンバータを起動させます。

●パラメータ・ファイル（78K1.PLV）の内容

```
: parameter file
78k1main -l78k1.lnk
-e78k1.elv
```

●コマンド行には，次のように入力します。

```
A>lcnv78k1 -f78k1.plv

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

保守／廃止

(6) ヘルプ指定　(－－)

記述形式　：－－

省略時解釈：表示しない

【機　　能】

－－オプションは，ヘルプ・メッセージをディスプレイに表示します。

【用　　途】

ヘルプ・メッセージは，リスト・コンバータ・オプションとその説明の一覧です。リスト・コンバータを実行するときに参照してください。

【説　　明】

－－オプションを指定すると他のリスト・コンバータ・オプションは，すべて無効となります。

【使 用 例】

例　－－オプションを指定するとヘルプ・メッセージがディスプレイに出力されます。

```
A>lcnv78k1 --

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

usage : LCNV78K1 [option[...]] input-file [option[...]]
The option is as follows([]means omissible).
-R[file]:Specify object module file.
-L[file]:Specify load module file.
-O[file]:Specify output list file (absolute assemble list file).
-Ffile  :Input option or input-file name from specified file.
-E[file]:Create error list file.
--      :Show this message.
```

保守／廃止

第９章　プログラムの出力リスト

以下に示す各プログラムが出力する各種リストのフォーマットなどについて説明します。

○アセンブルの出力リスト

アセンブル・リスト・ファイルのヘッダ

アセンブル・リスト

シンボル・リスト

クロスレファレンス・リスト

エラー・リスト

○リンカの出力リスト

リンク・リスト・ファイルのヘッダ

マップ・リスト

パブリック・シンボル・リスト

ローカル・シンボル・リスト

エラー・リスト

○オブジェクト・コンバータの出力リスト

エラー・リスト

○ライブラリアンの出力リスト

ライブラリ情報出力リスト

○リスト・コンバータの出力リスト

アブソリュート・アセンブル・リスト

エラー・リスト

保守／廃止

9.1　アセンブラの出力リスト

アセンブラは，次のリストを出力します。

出力リスト・ファイル名	出力リスト名
アセンブル・リスト・ファイル	アセンブル・リスト
	シンボル・リスト
	クロスレファレンス・リスト
エラー・リスト・ファイル	エラー・リスト

9.1.1　アセンブル・リスト・ファイルのヘッダ

ヘッダ部は，常にアセンブル・リスト・ファイルの先頭に出力されます。

【出力形式】

```
78K/I Series Assembler ①Vx.xx      ②                Date:③xx xxx xxxx Page:④ 1

⑤

Command:  ⑥-C138 78K1MAIN.ASM -lw80
Para-file:⑦
In-file:  ⑧78K1MAIN.ASM
Obj-file: ⑨78K1MAIN.REL
Prn-file: ⑩78K1MAIN.PRN
```

【出力項目の説明】

項目	内　　容
①	アセンブラのバージョン番号
②	タイトル文字列 －LH オプションまたは TITLE 制御命令によって指定された文字列
③	アセンブル・リストの作成年月日
④	ページ番号
⑤	サブタイトル文字列 SUBTITLE 制御命令によって指定された文字列
⑥	コマンド行のイメージ
⑦	パラメータ・ファイルの内容
⑧	入力ソース・モジュール・ファイル名
⑨	出力オブジェクト・モジュール・ファイル名
⑩	アセンブル・リスト・ファイル名

保守／廃止

9.1.2　アセンブル・リスト

アセンブル・リストは，アセンブル結果をエラー・メッセージ（エラーがある場合のみ）とともに出力します。

【出力形式】

```
        Assemble list

 ALNO  STNO   ADRS    OBJECT   M I  SOURCE STATEMENT

 ① 1  ② 1                    ③ ④  $      PROCESSOR(138)
    2     2
    3     3                           NAME    SAMPM
    4     4                       ;*********************************************
    5     5                       ;*                                           *
    6     6                       ;*      HEX -> ASCII Conversion Program      *
    7     7                       ;*                                           *
    8     8                       ;*             main-routine                  *
    9     9                       ;*                                           *
   10    10                       ;*********************************************
   11    11
   12    12                              PUBLIC  MAIN,START
   13    13                              EXTRN   CONVAH
   14    14
   15    15 ⑥----                        DSEG
   16    16   0000               ⑤STASC:  DS      2
   17    17
   18    18   ----                       CSEG
   19    19   ----                       ORG     0H
   20    20   0000 ⑧R0000          MAIN:  DW      START
   21    21
   22    22   ----                       CSEG
   23    23   0000    0BFCE0FE     START: MOVW    SP,#0FEE0H
⑦***   ERROR F201, STNO    23 (    0) Syntax error
   24    24   0004    2BC400             MOV     MM,#00
   25    25   0007    09C0FF00           MOV     STBC,#00
   26    26
   27    27   000B    BC00               MOV     E,#LOW HDTSA
   28    28
   29    29   000D   R280000             CALL    !CONVAH
   30    30
   31    31   0010   R640000             MOVW    DE,#STASC
   32    32   0013    D3                 MOV     A,B
   33    33   0014    50                 MOV     [DE+],A
   34    34   0015    D2                 MOV     A,C

                                  ⋮

  Segment informations:

  ADRS    LEN      NAME

 ⑨0000  ⑩0002H  ⑪?DSEG
  0000    0015H    ?CSEG
  0000    0002H    ?ASEG1
  FE00    0001H    DATA

  Target chip: ⑫ uPD78138
 Assembly complete,    ⑬ 1 error(s) and ⑭ 0 warning(s) found. ( ⑮ 23)
```

【出力項目の説明】

項目	内　　容
①	ソース・モジュールのイメージの行番号
②	行番号（INCLUDEファイルの展開，マクロ展開も含みます）
③	マクロ表示 M　：マクロ定義行です。 #n　：マクロ展開行です。nはネスト・レベルです。 空白：マクロ定義行/マクロ展開行ではありません。
④	INCLUDE表示 In　：INCLUDEファイル中です。nはネスト・レベルです。 空白：INCLUDEファイル未使用
⑤	ソース・プログラム・ステートメント
⑥	ロケーション・カウンタ値
⑦	フェイタル・エラー/ワーニング発生行
⑧	リロケーション情報 R　：リンカによってオブジェクト・コードまたはシンボル値が変更されます。 空白：オブジェクト・コードまたはシンボル値が変更されません。
	オブジェクト・コード
	EQU，SET疑似命令で設定されたシンボル値
⑨	セグメント・アドレス
⑩	セグメント・サイズ
⑪	セグメント名
⑫	RA78K/Iの対象デバイス
⑬	フェイタル・エラーの個数
⑭	ワーニングの個数
⑮	最終エラー行

保守／廃止

9.1.3　シンボル・リスト

ソース・モジュール内で定義されているシンボル（ローカル・シンボルを含む）の情報を出力します。

【出力形式】

```
                Symbol Table List

VALUE    ATTR    RTYP   NAME            VALUE    ATTR    RTYP   NAME

        ②DSEG          ④?ASEG1                  ②CSEG          ④?CSEG
          DSEG            ?DSEG      ①---H          ③EXT     CONVAH
          DSEG            DATA        FE00H    ADDR           HDTSA
①0H       ADDR  ③PUB     MAIN                 MOD            SAMPM
  0H      ADDR    PUB     START          0H    ADDR           STASC
```

【出力項目の説明】

項目	内　　容
①	シンボル値
②	シンボル属性 CSEG　：コード・セグメント名 DSEG　：データ・セグメント名 BSEG　：ビット・セグメント名 MOD　：モジュール名 SET　：SET 疑似命令によって定義されたシンボル NUM　：NUMBER 属性シンボル DNUM　：DNUMBER 属性シンボル ADDR　：ADDRESS 属性シンボル ABIT　：BIT 属性シンボル（addr.bit） SABIT　：BIT 属性シンボル（saddr.bit） SFBIT　：BIT 属性シンボル（sfr.bit） RBIT　：BIT 属性シンボル（A.bit, X.bit, PSW.bit, PSWL.bit, PSWH.bit） RBBIT　：BIT 属性シンボル（br.bit） RWBIT　：BIT 属性シンボル（wr.bit） 空白　：EXTRN または EXTBIT 宣言された外部参照シンボル ＊＊＊＊＊：未定義シンボル
③	シンボル参照形式 EXT　：EXTRN 宣言された外部参照シンボル EXTB　：EXTBIT 宣言された外部参照ビット・シンボル PUB　：PUBLIC 宣言された外部定義シンボル 空白　：ローカル・シンボル，セグメント名，マクロ名，モジュール名 ＊＊＊＊＊：未定義シンボル
④	定義されたシンボル名

9.1.4　クロスレファレンス・リスト

ソース・モジュール内で定義されたシンボルがソース・モジュールのどこで（行番号）参照されているかという情報が出力されます。

【出力形式】

```
                Cross-Reference List

  NAME       VALUE  R   ATTR    RTYP   SEGNAME  XREFS

①?ASEG1              ④CSEG          ⑥?ASEG1   ⑦19#
  ?CSEG                CSEG            ?CSEG     18#    22#
  ?DSEG                DSEG            ?DSEG     15#
  CONVAH ②----H ③E             ⑤EXT              13@    29
  DATA                 DSEG            DATA      39#
  HDTSA     FE00H      ADDR            DATA      27     40#
  MAIN        0H       ADDR    PUB     ?ASEG1    12@    20#
  SAMPM                MOD              3#
  START       0H   R   ADDR    PUB     ?CSEG     12@    20     23#
  STASC       0H   R   ADDR            ?DSEG     16#    31
```

【出力項目の説明】（1/2）

項目	内　　容
①	定義されたシンボル名
②	シンボル値
③	リロケーション属性 R　：リロケータブルなシンボル E　：external なシンボル 空白　：アブソリュートなシンボル ＊　：未定義シンボル
④	シンボル属性 CSEG　：コード・セグメント名 DSEG　：データ・セグメント名 BSEG　：ビット・セグメント名 MOD　：モジュール名 SET　：SET 疑似命令によって定義されたシンボル NUM　：NUMBER 属性シンボル DNUM　：DNUMBER 属性シンボル ADDR　：ADDRESS 属性シンボル ABIT　：BIT 属性シンボル（addr.bit） SABIT　：BIT 属性シンボル（saddr.bit） SFBIT　：BIT 属性シンボル（sfr.bit） RBIT　：BIT 属性シンボル（A.bit, X.bit, PSW.bit, PSWL.bit, PSWH.bit） RBBIT　：BIT 属性シンボル（br.bit） RWBIT　：BIT 属性シンボル（wr.bit） 空白　：EXTRN または EXTBIT 宣言された外部参照シンボル ＊＊＊＊＊：未定義シンボル

【出力項目の説明】(2/2)

項目	内　　容
⑤	シンボル参照形式 EXT　　：EXTRN 宣言された外部参照シンボル EXTN　　：EXTBIT 宣言された外部参照ビット・シンボル PUB　　：PUBLIC 宣言された外部定義シンボル 空白　　：ローカル・シンボル，セグメント名，マクロ名，モジュール名 ＊＊＊＊＊：未定義シンボル
⑥	定義されたセグメント名
⑦	定義・参照行番号 定義行：××××××＃ 参照行：××××××▲　（▲は空白1つ） EXTRN 宣言，EXTBIT 宣言，PUBLIC 宣言：××××××@

9.1.5　エラー・リスト

アセンブラ起動時に出力されたエラー・メッセージが格納されています。

【出力形式】

```
  Pass1 Start
①78K1MAIN.ASM(②20) :③ F201 ④Syntax error
  78K1MAIN.ASM(23)   :   F201   Syntax error
  Pass2 Start
①78K1MAIN.ASM(②12) :③ F404 ④Public symbol is undefined 'MMAIN'
  78K1MAIN.ASM(20)   :   F201   Syntax error
  78K1MAIN.ASM(23)   :   F201   Syntax error
```

【出力項目の説明】

項目	内　　容
①	エラーの発生したソース・モジュール・ファイル名
②	エラーの発生行
③	エラー番号
④	エラー・メッセージ

保守／廃止

9.2　リンカの出力リスト

リンカは，次のリストを出力します。

出力リスト・ファイル名	出力リスト名
リンク・リスト・ファイル	マップ・リスト
	パブリック・シンボル・リスト
	ローカル・シンボル・リスト
エラー・リスト・ファイル	エラー・リスト

9.2.1　リンク・リスト・ファイルのヘッダ

ヘッダ部は常にリンク・リスト・ファイルの先頭に出力されます。

【出力形式】

```
78K/I Series Linker ①Vx.xx                Date:②xx xxx xxxx Page:③ 1

Command:   ④ 78k1main.rel 78k1sub.rel -d78k1.dr -p78k1.map -kd
Para-file: ⑤
Out-file:  ⑥ 78K1MAIN.LNK
Map-file:  ⑦ 78K1.MAP
Direc-file:⑧ 78K1.DR
Directive: ⑨ memory ROM:(00000H,3FFFH)
             memory RAM:(0D000H,2EFFH)

*** Link information ***

⑩   4 output segment(s)
⑪ 38H byte(s) real data
⑫  18 symbol(s) defined
```

保守／廃止

【出力項目の説明】

項目	内　　容
①	リンカのバージョン番号
②	リンク・リスト・ファイルの作成年月日
③	ページ番号
④	コマンド行のイメージ
⑤	パラメータ・ファイルの内容
⑥	出力ロード・モジュール・ファイル名
⑦	リンク・リスト・ファイル名
⑧	ディレクティブ・ファイル名
⑨	ディレクティブ・ファイルの内容
⑩	ロード・モジュール・ファイルに出力されるセグメント数
⑪	ロード・モジュール・ファイルに出力されるデータの大きさ
⑫	ロード・モジュール・ファイルに出力されるシンボル数

9.2.2　マップ・リスト

セグメントの配置に関する情報を出力します。

【出力形式】

```
*** Memory map ***


①SPACE=REGULAR

  MEMORY=②ROM
  BASE ADDRESS=③0000H    SIZE=④8000H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
        ⑥?ASEG1                      ⑨0000H   ⑩0002H ⑪CSEG AT
                 ⑦?ASEG1 ⑧SAMPM      0000H     0002H
⑤* gap *                             0002H     007EH
         ?CSEG                       0080H     0035H   CSEG
                  ?CSEG    SAMPM     0080H     0019H
                  ?CSEG    SAMPS     0099H     001CH
  * gap *                            00B5H     7F4BH

  MEMORY=②RAM
  BASE ADDRESS=③FC80H    SIZE=④0280H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
        ⑥?DSEG                       ⑨FC80H   ⑩0002H ⑪CSEG
                 ⑦?DSEG  ⑧SAMPM      FC80H     0002H
⑤* gap *                             FC82H     017EH
         DATA                        FE00H     0001H   DSEG AT
                  DATA     SAMPM     FE00H     0001H
  * gap *                            FE01H     00FFH
```

【出力項目の説明】

項目	内　　容
①	メモリ空間名
②	メモリ領域名
③	メモリ領域の先頭アドレス
④	メモリ領域のサイズ
⑤	出力グループ名 何も配置されないエリアがある場合 'gap' を表示します。
⑥	ロード・モジュール・ファイルに出力されるセグメント名
⑦	オブジェクト・モジュール・ファイルから読み込まれたセグメント名
⑧	入力モジュール名
⑨	セグメントの先頭アドレス
⑩	出力セグメント/入力セグメントのサイズ
⑪	セグメント・タイプ，再配置属性

9.2.3　パブリック・シンボル・リスト

入力モジュール内で定義されているパブリック・シンボルの情報を出力します。

【出力形式】

```
*** Public symbol list ***

MODULE     ATTR       VALUE     NAME

①SAMPM    ②ADDR     ③0000H   ④MAIN
 SAMPM      ADDR       0080H     START
 SAMPS      ADDR       0099H     CONVAH
```

【出力項目の説明】

項目	内　　容	
①	パブリック・シンボルが定義されたモジュール名	
②	シンボル属性	ABIT　：BIT 属性シンボル（addr.bit）
	CSEG　：コード・セグメント名	SABIT　：BIT 属性シンボル（saddr.bit）
	DSEG　：データ・セグメント名	SFBIT　：BIT 属性シンボル（sfr.bit）
	BSEG　：ビット・セグメント名	RBIT　：BIT 属性シンボル（A.bit,
	MOD　：モジュール名	X.bit, PSW.bit, PSWL.bit, PSWH.bit）
	SET　：SET 疑似命令によって定義	RBBIT　：BIT 属性シンボル（br.bit）
	されたシンボル	RWBIT　：BIT 属性シンボル（wr.bit）
	NUM　：NUMBER 属性シンボル	空白　：EXTRN または EXTBIT 宣言された
	DNUM　：DNUMBER 属性シンボル	外部参照シンボル
	ADDR　：ADDRESS 属性シンボル	＊＊＊＊＊：未定義シンボル
③	シンボル値	
④	パブリック・シンボル名	

9.2.4　ローカル・シンボル・リスト

入力モジュール内で定義されているローカル・シンボルの情報を出力します。

【出力形式】

```
*** Local symbol list ***

MODULE    ATTR       VALUE    NAME

①SAMPM   ②MOD                ④SAMPM
 SAMPM     DSEG                 ?DSEG
 SAMPM     ADDR     ③FC80H     STASC
 SAMPM     CSEG                 ?CSEG
 SAMPM     CSEG                 ?ASEG1
 SAMPM     ADDR      FE00H      HDTSA
 SAMPM     DSEG                 DATA
 SAMPS     MOD                  SAMPS
 SAMPS     CSEG                 ?CSEG
 SAMPS     ADDR      00ACH      SASC
 SAMPS     ADDR      00B2H      SASC1
```

【出力項目の説明】

項目	内　　容	
①	ローカル・シンボルが定義されたモジュール名	
②	シンボル属性	ABIT　：BIT 属性シンボル（addr.bit）
	CSEG　：コード・セグメント名	SABIT　：BIT 属性シンボル（saddr.bit）
	DSEG　：データ・セグメント名	SFBIT　：BIT 属性シンボル（sfr.bit）
	BSEG　：ビット・セグメント名	RBIT　：BIT 属性シンボル（A.bit,
	MOD　：モジュール名	X.bit, PSW.bit, PSWL.bit, PSWH.bit）
	SET　：SET 疑似命令によって定義	RBBIT　：BIT 属性シンボル（br.bit）
	されたシンボル	RWBIT　：BIT 属性シンボル（wr.bit）
	NUM　：NUMBER 属性シンボル	空白　：EXTRN または EXTBIT 宣言された
	DNUM　：DNUMBER 属性シンボル	外部参照シンボル
	ADDR　：ADDRESS 属性シンボル	＊＊＊＊＊：未定義シンボル
③	シンボル値	
④	ローカル・シンボル名	

9.2.5　エラー・リスト

リンカ起動時に出力されたエラー・メッセージが格納されています。

【出力形式】

```
SAMP.DR(2) : ①F111 ②Memory area 'RAM' redefinition out of range
SAMP.DR(3) : ①F102 ②Directive syntax error
```

【出力項目の説明】

項目	内　　容
①	エラー番号
②	エラー・メッセージ

9.3　オブジェクト・コンバータの出力リスト

オブジェクト・コンバータは，次のリストを出力します。

出力リスト・ファイル名	出力リスト名
エラー・リスト・ファイル	エラー・リスト

9.3.1　エラー・リスト

オブジェクト・コンバータ起動時に出力されたエラー・メッセージが格納されています。

【出力形式】

リンカの出力するエラー・リストと同一です。

保守／廃止

9.4　ライブラリアンの出力リスト

ライブラリアンは，次のリストを出力します。

出力リスト・ファイル名	出力リスト名
リスト・ファイル	ライブラリ情報出力リスト

9.4.1　ライブラリ情報出力リスト

ライブラリ・ファイル内のモジュールに関する情報を出力します。

【出力形式】

```
  78K/I Series librarian Vx.xx    DATE : ①xx xxx xx                    PAGE ② 1

  LIB-FILE NAME :③ 78K1.LIB        (④xx xxx xx)

⑤0001  ⑥78K1MAIN.REL     (⑦xx xxx xx)
    ⑧MAIN                              ⑧START
    NUMBER OF PUBLIC SYMBOLS : ⑨ 2

⑤0002  ⑥78K1SUB.REL      (⑦xx xxx xx)
    ⑧CONVAH
    NUMBER OF PUBLIC SYMBOLS : ⑨ 1
```

【出力項目の説明】

項目	内　　容
①	リストの作成年月日
②	ページ数
③	ライブラリ・ファイル名
④	ライブラリ・ファイルの作成年月日
⑤	モジュールの通番（0001から番号を付けます）
⑥	モジュール名
⑦	モジュール作成年月日
⑧	パブリック・シンボル名
⑨	モジュール内で定義されているパブリック・シンボル数

保守／廃止

9.5　リスト・コンバータの出力リスト

リスト・コンバータは，次のリストを出力します。

出力リスト・ファイル名	出力リスト名
アブソリュート・アセンブル・リスト・ファイル	アブソリュート・アセンブル・リスト
エラー・リスト・ファイル	エラー・リスト

9.5.1　アブソリュート・アセンブル・リスト

アセンブル・リストにアブソリュートな値を埋め込んで出力します。

【出力形式】

アセンブラの出力するアセンブル・リストと同一です。

9.5.2　エラー・リスト

リスト・コンバータ起動時に出力されたエラー・メッセージが格納されています。

【出力形式】

アセンブラの出力するエラー・リストと同一です。

保守／廃止

保守／廃止

第10章　RA78K/Iの活用法

RA78K/Iを効率よく使用するための方法を紹介します。

保守／廃止

10.1　作業の効率化（EXITステータス機能）

RA78K/Iの各プログラムは，処理終了時に，処理中に発生した最大のエラー・レベルをEXITステータスとして，OSに返します。

EXITステータスを次に示します。

- 正常終了時　　：0
- WARNINGあり　　：0
- FATAL　ERORRあり：1
- ABORT時　　：2

これらを利用してバッチ・ファイルを作成することで効率良く作業を行えます。

【使 用 例】

例　バッチ・ファイル（RA.BAT）の内容

```
ra78k1 -c138 %1.asm -g -e
echo off
IF ERRORLEVEL 1 GOTO ERR
echo¥
echo on
ra78k1 -c138 %2.asm -g -e
echo off
IF ERRORLEVEL 1 GOTO ERR
echo¥
echo on
lk78k1 %1.rel %2.rel -o%3.lnk -g
echo off
IF ERRORLEVEL 1 GOTO ERR
echo¥
echo on
cc78k1 %3.lnk
echo off
IF ERRORLEVEL 1 GOTO ERR
GOTO EXIT
:ERR
echo エラーが発生しました。
echo off
:EXIT
```

●バッチ・ファイル（RA.BAT）を使用して処理を行います。

```
A>ra.bat
```

保守／廃止

10.2　開発環境の整備（環境変数）

RA78K/Iでは開発環境を整えるため，次の環境変数をサポートしています。

PATH	：実行形式のサーチ・パス
INC78K1	：インクルード・ファイルのサーチ・パス（アセンブラのみ）
LIB78K1	：ライブラリ・ファイルのサーチ・パス（リンカのみ）
TMP	：テンポラリ・ファイルを作成するパス

プログラム開発を行う場合，関連ファイル別にサブディレクトリを作成し，関連ファイルをひとまとめにしておくと作業が円滑に行えるようになります。

【使 用 例】

例　AUTOEXEC．BATの内容

```
;AUTOEXEC.BAT
verify on
break on
PATH A:¥BIN;A:¥BAT;A:¥RA78K1;      ←①
SET INC78K1=A:¥RA78K1¥INCLUDE      ←②
SET LIB78K1=A:¥RA78K1¥LIB          ←③
SET TMP=A:¥TMP                     ←④
```

① パス指定により，A：¥BIN，A：¥BAT，A：¥RA78K1という順に実行形式ファイルを検索します。

② アセンブラは，インクルード・ファイルをA：¥RA78K1¥INCLUDEから検索します。

③ リンカは，ライブラリ・ファイルをA：¥RA78K1¥LIBから検索します。

④ 各プログラムは，テンポラリ・ファイルをA：¥TMPに作成します。

10.3　プログラム実行の中断

キー入力（CTRL-C）により各プログラムの実行を中断できます。

AUTOEXEC．BAT中で‘break on’を指定した場合は，キー入力のタイミングに関係なしに制御をOSに戻し，‘break off’を指定した場合には画面表示中にのみ制御をOSに戻します。そして，オープン中のすべてのテンポラリ・ファイル，出力ファイルを削除します。

保守／廃止

10.4　アセンブル・リストを見やすくする

－LHオプションやTITLE制御命令を使用してアセンブル・リストのヘッダにタイトルを表示します。アセンブル・リストの内容を端的に表すようなタイトルを表示しておくことでアセンブル・リストの内容がわかりやすくなります。

また，SUBTITLE制御命令を使用すればサブタイトルが表示できます。制御命令については，言語編の“**第4章　制御命令**”をお読みください。

【使 用 例】

例　アセンブル・リスト・ファイルのヘッダにタイトルを印字します。

```
A>ra78k1 -c138 78k1main.asm -lhRA78K1-MAINROUTINE

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.
```

78K1MAIN．PRNを参照します。

```
A>type 78k1main.prn

78K/I Series Assembler Vx.xx   RA78K1-MAINROUTINE            Date:xx xxx xxxx Page:   1
                                  └ ｺﾏﾝﾄﾞ･ﾗｲﾝで指定したﾀｲﾄﾙ

Command:  -c138 78k1main.asm -lhRA78K1-MAINROUTINE
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                    $       PROCESSOR(138)
    2     2
    3     3                            NAME    SAMPM
    4     4                    ;*************************************************
    5     5                    ;*                                               *
    6     6                    ;*      HEX -> ASCII Conversion Program          *
    7     7                    ;*                                               *
    8     8                    ;*              main-routine                     *
    9     9                    ;*                                               *
   10    10                    ;*************************************************
   11    11
   12    12                            PUBLIC  MAIN,START
   13    13                            EXTRN   CONVAH

                                        :
```

10.5　プログラム起動時の手間を省く

10.5.1　ソース・プログラムに制御命令を記述する

アセンブラ起動時に常に指定するオプションと同一の機能を持つ制御命令は，あらかじめソース・プログラム中で指定しておきます。これにより，アセンブラを起動するたびにオプションを指定する必要がなくなります。

【使 用 例】

```
$       PROCESSOR(138)      ┐
$       DEBUG               ├ 制御命令
$       XREF                ┘

        NAME    SAMPM
;************************************************
;*                                              *
;*      HEX -> ASCII Conversion Program         *
;*                                              *
;*              main-routine                    *
;*                                              *
;************************************************

        PUBLIC  MAIN,START
        EXTRN   CONVAH
                  ⋮
```

10.5.2　パラメータ・ファイルやサブコマンド・ファイルを作成する

プログラム（アセンブラ，リンカ，オブジェクト・コンバータおよびリスト・コンバータ）の起動時にコマンド行に起動に必要な情報を指定しきれない場合やプログラムを起動するたびに同じオプションを指定するような場合にはパラメータ・ファイルを使用します。

また，ライブラリアンにおいては，サブコマンド・ファイルにサブコマンドを登録指定してオブジェクト・モジュールのライブラリ化が容易にできます。

【使 用 例】

例 1. パラメータ・ファイルを使用してアセンブルします。

パラメータ・ファイル 78K1MAIN．PRA の内容

```
;parameter file
78k1main.asm -osample.rel -g
-psample.prn
```

コマンド行には，次のように入力します。

```
A>ra78k1 -f78k1main.pra
```

例 2. サブコマンド・ファイル使用してライブラリアンを起動します。

●78K1．SLBの内容

```
;
;library creation command
;
create 78k1.lib
;
add 78k1.lib 78k1main.rel &
78k1sub.rel
;
exit
```

●コマンド行には，次のように入力します。

```
A>lb78k1 <78k1.slb
```

10.6　オブジェクト・モジュールのライブラリ化

アセンブラおよびリンカは，1つの出力モジュールを1つのファイルに作成します。したがって，モジュールの数が多い場合はファイルの数も増加します。このため，複数のモジュールを1つのファイルにまとめる機能を用意しました。これをモジュールのライブラリ化と呼び，ライブラリ化されたファイルをライブラリ・ファイルと呼びます。

ライブラリ・ファイルは，リンカに入力できます。したがって，モジュラ・プログラミングを行った場合，共通的なモジュールをライブラリ・ファイルとして作成しておけば，ファイル管理の面でも操作性の面においても効率がよくなります。

保守／廃止

第11章　エラー・メッセージ

この章では，RA78K/I（アセンブラ，リンカ，オブジェクト・コンバータ，ライブラリアン，リスト・コンバータ）の出力するエラー・メッセージの原因，ユーザの処置などについて説明します。

保守／廃止

11.1　エラー・メッセージの概要

RA78K/Iのエラー・メッセージは以下の３種類にレベル分けしています。

(1) アボート・エラー（A×××）

処理続行が不可能なエラーが発生したため，ただちに処理を終了（中断）します。

なお，起動行のアボート・エラーに関しては，他の起動行のエラーを検出してから処理を終了します。

(2) フェイタル・エラー（F×××）

実行プログラム・エラーが発生したため，他のエラーを検出後，出力オブジェクトを生成せずに処理を終了（中断）します。

なお，フェイタル・エラー時に出力オブジェクトを生成しないことを明示するため，同名のオブジェクトが存在する場合は消去して終了します。

(3) ワーニング・エラー（W×××）

コンパイラ：ユーザが意図したものとは異なる可能性がありますが，正常に動作する出力オブジェクトを生成します。

アセンブラ：コード生成と無関係の場所にエラーはありますが，出力オブジェクトはユーザの意図するものを生成します。

備考　会話形式の実行プログラムでは，アボート・エラーの発生以外はすべて正常終了とします。

アセンブラのエラー・メッセージは次のように分類されています。
次頁よりアセンブラのエラー・メッセージについて説明します。

- A0××…コマンド行解析部のエラー
- A9××…ファイル，システムに関するエラー
- A1××…その他のアボート・エラー
- F2××…文の記述に関するエラー
- F3××…式に関するエラー
- F4××…シンボルに関するエラー
- F5××…セグメントに関するエラー
- F6××…制御命令，マクロに関するエラー
- W7××…各種ワーニング・エラー

保守／廃止

11.2　RA78K/I 共通のエラー・メッセージ

表 11 - 1　RA78K/I 共通のエラー・メッセージ（1/2）

A001	メッセージ	Missing input file
	原　　因	オプションのみの指定で，入力ファイルが１つも指定されていません。
A002	メッセージ	Too many input files
	原　　因	入力ファイルの総数が制限を越えて指定されました。
A003	メッセージ	Unrecognized string '???'
	原　　因	会話形式のコマンド行にオプション以外のものが指定されました。
A004	メッセージ	Illegal file name 'ファイル名'
	原　　因	ファイル名に OS で許されない文字があるか，文字数が制限を越えています。
A005	メッセージ	Illegal file specification 'ファイル名'
	原　　因	ファイル名に不当なものが指定されました。
A006	メッセージ	File not found 'ファイル名'
	原　　因	指定された入力ファイルが存在しません。
A007	メッセージ	Input file specification overlapped 'ファイル名'
	原　　因	入力ファイル名が重複して指定されました。
A008	メッセージ	File specification conflicted 'ファイル名'
	原　　因	入出力ファイル名が重複して指定されました。
A009	メッセージ	Unable to make file 'ファイル名'
	原　　因	指定された出力ファイルが作成できません。
A010	メッセージ	Directory not found 'ファイル名'
	原　　因	出力ファイル名中に存在しないドライブ，またはディレクトリが含まれています。
A011	メッセージ	Illegal path 'ファイル名'
	原　　因	パラメータにパス名を指定するオプションで，パス名以外が指定されました。
A012	メッセージ	Missing parameter 'オプション'
	原　　因	必要なパラメータが指定されていません。
A013	メッセージ	Parameter not needed 'オプション'
	原　　因	不要なパラメータが指定されました。
A014	メッセージ	Out of range 'オプション'
	原　　因	指定数値が範囲外です。
A015	メッセージ	Parameter is too long 'オプション'
	原　　因	パラメータの文字数が制限を越えて指定されました。
A016	メッセージ	Illegal parameter 'オプション'
	原　　因	パラメータの文法に誤りがあります。

表 11 - 1　RA78K/I 共通のエラー・メッセージ（2/2）

A017	メッセージ	Too many parameters 'オプション'
	原　　因	パラメータの総数が制限を越えました。
A018	メッセージ	Option is not recognized 'オプション'
	原　　因	誤ったオプションが指定されました。
A019	メッセージ	Parameter file nested
	原　　因	パラメータ・ファイル中に −F オプションが，指定されました。
A020	メッセージ	Parameter file read error 'ファイル名'
	原　　因	パラメータ・ファイルの読み込みに，失敗しました。
A021	メッセージ	Memory allocation failed
	原　　因	メモリ・アロックに失敗しました。

保守／廃止

11.3　アセンブラのエラー・メッセージ

A0××のエラー・メッセージについては，“**11.2　RA78K/I共通のエラー・メッセージ**”を参照してください。

表11-2　アセンブラのエラー・メッセージ（1/10）

A101	メッセージ	Source file size 0 'ファイル名'
	原　　因	サイズが0バイトのソース・ファイルを入力しました。
A102	メッセージ	Illegal processor type specified
	原　　因	対象デバイスの指定が間違っています。
A103	メッセージ	Syntax error in module header
	原　　因	ソース・モジュール・ヘッダに記述可能な制御命令の記述形式が間違っています。
A104	メッセージ	Can' t use this control outside module header
	原　　因	ソース・モジュール・ヘッダに記述する制御命令が，通常のソースに記述されています。
A105	メッセージ	Duplicate PROCESSOR control
	原　　因	ソース・モジュール・ヘッダの中でPROCESSOR制御命令が重複して記述されています。
A106	メッセージ	Illegal source file name for module name
	原　　因	ソース・ファイル名のプライマリ・ネームがシンボルの構成文字に反しているためモジュール名が作成できません。 または，ソース・ファイル中ですでに定義されているためモジュール名が作成できません。
A107	メッセージ	Default segment ?CSEG is already used
	原　　因	セグメント定義省略時にディフォールト・セグメントを定義しようとしました。
A108	メッセージ	Symbol table overflow 'シンボル名'
	原　　因	定義可能なシンボル数（2900個）の制限を越えています。 出力されるシンボル名は，制限を越えたシンボルの名称です。
A109	メッセージ	Too many DS
	原　　因	DS疑似命令が多くあるために，セグメント内のオブジェクト・コードの間隔が空きすぎて，オブジェクト・ファイルに情報を出力できません。
	ユーザの処置	ソース・ファイルを分割するかDS疑似命令の数を減らしてください。
A110	メッセージ	String table overflow
	原　　因	ストリング・テーブルの制限を越えました。
	ユーザの処置	9文字以上のシンボルの数を減らしてください。

保守／廃止

表 11 - 2　アセンブラのエラー・メッセージ（2/10）

A111	メッセージ	Object code more than 128 bytes
	原　　因	オブジェクト・コードがソース・ステートメント 1 行につき 128 バイトを越えました。
A112	メッセージ	No processor specified
	原　　因	対象デバイスがコマンド行上にも，ソース・モジュール・ファイル中にも指定されていません。

F201	メッセージ	Syntax error
	原　　因	文の記述形式が間違っています。
F202	メッセージ	Illegal operand
	原　　因	オペランドの記述が不正です。
F203	メッセージ	Illegal register
	原　　因	記述できないレジスタが指定されました。
F204	メッセージ	Illegal character
	原　　因	ソース・モジュール中に不正な文字の記述があります。
F205	メッセージ	Unexpected LF in string
	原　　因	文字列が閉じる前に改行コードが現れました。
F206	メッセージ	Unexpected EOF in string
	原　　因	文字列が閉じる前にファイルの終わりになりました。
F207	メッセージ	Unexpected null code in string
	原　　因	文字列中にヌル・コード（00H）が記述されました。

F301	メッセージ	Too complex expression
	原　　因	式が複雑すぎます。
F302	メッセージ	Absolute expression expected
	原　　因	リロケータブルな式が記述されています。
F303	メッセージ	Illegal expression
	原　　因	式の記述形式が間違っています。
F304	メッセージ	Illegal symbol in expression 'シンボル名'
	原　　因	式の中に使用できないシンボルが記述されています。
F305	メッセージ	Too long string as constant
	原　　因	文字定数の長さの制限（2 文字）を越えています。
F306	メッセージ	Illegal number
	原　　因	記述された数値が間違っています。

保守／廃止

表 11-2　アセンブラのエラー・メッセージ (3/10)

F307	メッセージ	Division by zero
	原　　因	0 で除算をしています。
F308	メッセージ	Too large integer
	原　　因	定数の値が 16 ビットを越えています。
F309	メッセージ	Illegal bit value
	原　　因	ビット値の記述に誤りがあります。
F310	メッセージ	Bit value out of range
	原　　因	ビット値が 0～7 の範囲を越えました。
F311	メッセージ	Operand out of range (n)
	原　　因	指定された値が，n (0～7) の範囲を越えました。
F312	メッセージ	Operand out of range (byte)
	原　　因	オペランドの値が範囲 (00H～FFH) を越えました。
F313	メッセージ	Operand out of range (addr5)
	原　　因	addr5 として記述可能な範囲 (40H～7EH) を越えました。
F314	メッセージ	Operand out of range (addr11)
	原　　因	addr11 として記述可能な範囲 (800H～FFFH) を越えました。
F315	メッセージ	Operand out of range (saddr)
	原　　因	saddr として記述可能な範囲 (0FE20H～0FF1FH) を越えました。
F316	メッセージ	Operand out of range (addr13)
	原　　因	addr13 として記述可能な範囲 (対象デバイスによって異なる) を越えました。
F317	メッセージ	Even expression expected
	原　　因	ワード・アクセスに奇数アドレスを記述しています。
F318	メッセージ	Operand out of range (sfr)
	原　　因	SFR/SFRP 疑似命令のオペランドの記述可能な範囲(FF00H～FFFFH)を越えたか，あるいは SFRP 疑似命令のオペランドとして奇数を記述しています。

F401	メッセージ	Illegal symbol for PUBLIC 'シンボル名'
	原　　因	このシンボルは PUBLIC 宣言できません。
F402	メッセージ	Illegal symbol for EXTRN/EXTBIT 'シンボル名'
	原　　因	このシンボルは EXTRN/EXTBIT 宣言できません。

保守／廃止

表 11 - 2　アセンブラのエラー・メッセージ（4/10）

F403	メッセージ	Can't define PUBLIC symbol 'シンボル名'
	原　　因	すでに PUBLIC 宣言されたシンボルに，PUBLIC 宣言できないシンボルを定義しました。
	ユーザの処置	saddr. bit 以外のビット項を定義したシンボル，SET シンボル，既に外部参照宣言されたシンボル，セグメント名，モジュール名，マクロ名，BSFR/WSFR シンボル（ユーザ定義シンボル），EQUD シンボルは，PUBLIC 宣言できないので PUBLIC 宣言を取り消すか，EQU の定義を変更してください。
F404	メッセージ	Public symbol is undefined 'シンボル名'
	原　　因	PUBLIC 宣言されたシンボルが定義されていません。
F405	メッセージ	Illegal bit symbol
	原　　因	機械語命令のオペランドのビット・シンボルに，前方参照のシンボルあるいはビット・シンボルとして不適当なシンボルを使用しています。
	ユーザの処置	ビット・シンボルには，後方参照あるいは EXTBIT 宣言したシンボルを記述してください。
F406	メッセージ	Can't refer to forward bit symbol 'シンボル名'
	原　　因	ビット・シンボルを前方参照しているか，または式の中にビット・シンボルを記述しています。
F407	メッセージ	Undefined symbol reference 'シンボル名'
	原　　因	未定義シンボルを使用しています。
F408	メッセージ	Multiple symbol definition 'シンボル名'
	原　　因	シンボル名が重複して定義されています。
F409	メッセージ	Too many symbols in operand
	原　　因	1 行以内に記述可能なオペランドのシンボル個数が制限を越えました。
F410	メッセージ	Phase error
	原　　因	アセンブル中にシンボルの値が変化しました（たとえば，BR 疑似命令の最適化処理によって変化したレーベルをオペランドの中に用いて定義した EQU シンボルなど）。
	ユーザの処置	そのシンボルを参照しているところを検索し，EQU シンボルではなく直接，式を記述するように修正してください。

F501	メッセージ	Too many default ORG segment
	原　　因	セグメント名指定のない ORG 疑似命令が制限個数（1 モジュール内に 20 個）以上記述されました。

保守／廃止

表 11 - 2　アセンブラのエラー・メッセージ（5/10）

F502	メッセージ	Illegal segment name
	原　　因	セグメント名として不正なシンボルが記述されています。
F503	メッセージ	Different segment type 'セグメント名'
	原　　因	同名セグメント定義において，セグメントのタイプが異なっています。
F504	メッセージ	Too many segments
	原　　因	定義できるセグメント数の制限（100 個）を越えています。
F505	メッセージ	Current segment is not exist
	原　　因	ENDS 疑似命令がセグメントが作られる前，あるいは一度セグメントが終了したあと，次のセグメントが作られる前に記述されています。
F506	メッセージ	Can't describe DB, DW, DS, ORG, label in BSEG
	原　　因	DB, DW, DS, ORG 疑似命令をビット・セグメント内で記述しています。
F507	メッセージ	Can't describe opcodes outside CSEG
	原　　因	機械語命令，BR 疑似命令をコード・セグメント以外で記述しています。
F508	メッセージ	Can't describe DBIT outside BSEG
	原　　因	DBIT 疑似命令をビット・セグメント以外で記述しています。
F509	メッセージ	Illegal address specified
	原　　因	アブソリュート・セグメントとして配置指定したアドレスが，そのセグメントに対応する範囲を越えています。
F510	メッセージ	Location counter overflow
	原　　因	ロケーション・カウンタがセグメントに対応した範囲を越えました。
F511	メッセージ	Segment name expected
	原　　因	再配置属性が AT のセグメント定義疑似命令でセグメント名が指定されていません。
F512	メッセージ	Segment size is odd numbers 'セグメント名'
	原　　因	再配置属性 callt0 のセグメントが奇数サイズで記述されています。

F601	メッセージ	Nesting over of include
	原　　因	インクルード・ファイルのネスティングできる制限（2 レベル）を越えています。
F602	メッセージ	Must be specified switches
	原　　因	スイッチ名が指定されていません。
F603	メッセージ	Too many switches described
	原　　因	スイッチ名の記述が制限個数（1 モジュール内で 5 個以内）を越えています。

保守／廃止

表 11 - 2　アセンブラのエラー・メッセージ（6/10）

F604	メッセージ	Nesting over of IF-clauses
	原　　因	IF/_IF 節のネスティングの制限（8 レベル）を越えています。
F605	メッセージ	Needless ELSE statement exists
	原　　因	必要のないところに ELSE 文が存在しています。
F606	メッセージ	Needless ENDIF statement exists
	原　　因	必要のないところに ENDIF 文が存在しています。
F607	メッセージ	Missing ELSE or ENDIF
	原　　因	IF 文または_IF 文に対となる ELSE 文，ENDIF 文の対応がとれていません。
F608	メッセージ	Missing ENDIF
	原　　因	IF 文または_IF 文と ENDIF 文との対応がとれていません。
F609	メッセージ	Illegal ELSEIF statement
	原　　因	ELSE 文のあとに ELSEIF または_ELSEIF 文が記述されています。
F610	メッセージ	Multiple symbol definition（MACRO）'シンボル名'
	原　　因	マクロ名として定義しようとしたシンボルが，すでに定義されています。
F611	メッセージ	Illegal syntax of parameter
	原　　因	マクロの仮パラメータの記述に誤りがあります。
F612	メッセージ	Too many parameter
	原　　因	1 マクロ定義の仮パラメータの個数が制限（16 個）を越えています。
F613	メッセージ	Same name parameter described 'シンボル名'
	原　　因	1 マクロ定義の仮パラメータとして同名のシンボルが指定されました。
F614	メッセージ	Can't nest macro definition
	原　　因	マクロ定義の中でマクロ定義を行っています。
F615	メッセージ	Illegal syntax of local symbol
	原　　因	LOCAL 疑似命令のオペランド記述に誤りがあります。
F616	メッセージ	Too many local symbols
	原　　因	1 つのマクロ・ボディ内で記述できるローカル・シンボル数の制限（64 個）を越えています。
F617	メッセージ	Missing ENDM
	原　　因	マクロ定義疑似命令に対応する ENDM 文がありません。
F618	メッセージ	Illegal syntax of ENDM
	原　　因	ENDM 文の記述が間違っています。
F619	メッセージ	Illegally defined macro
	原　　因	参照したマクロは定義時に誤りがあります。
F620	メッセージ	Illegal syntax of actual parameter
	原　　因	マクロの実パラメータの記述に誤りがあります。

保守／廃止

表 11 - 2　アセンブラのエラー・メッセージ（7/10）

F621	メッセージ	Nesting over of macro reference
	原　　因	マクロ参照において，ネスティングできる制限（8 レベル）を越えています。
F622	メッセージ	Illegal syntax of EXITM
	原　　因	EXITM 文の記述に誤りがあります。
F623	メッセージ	Illegal operand of REPT
	原　　因	REPT 疑似命令のオペランドに許されていない式が記述されています。
F624	メッセージ	More than ??RAFFFF
	原　　因	マクロ展開の際にローカル・シンボルの置き換えが，65535 個を越えました。
F625	メッセージ	Unexpected ENDM
	原　　因	余分な ENDM が現れました。
F626	メッセージ	Can't describe LOCAL outside macro definition
	原　　因	マクロ・ボディ以外の通常のソース・ステートメント中に LOCAL 疑似命令が記述されました。
F628	メッセージ	Illegal REPT/IRP block
	原　　因	①REPT/IRP がセグメント定義なしに記述され，また REPT/IRP ブロックに CSEG 疑似命令，レーベル，オブジェクトを生成する命令，DS 疑似命令のいずれも記述されなかった。 ②REPT/IRP がセグメント定義なしに記述され，また REPT/IRP ブロックに DSEG /BSEG 疑似命令が記述されました。

W701	メッセージ	Too long source line
	原　　因	ソース・ステートメント 1 行が 218 文字を越えています。
	プログラムの処理	219 文字以降を無視します。
W702	メッセージ	Duplicate PROCESSOR option and control
	原　　因	コマンド・ライン上の対象デバイス指定オプション（－C）とソース・ヘッダ中の PROCESSOR 制御命令が両方とも指定されています。
	プログラムの処理	コマンド・ライン上の対象デバイス指定オプションを有効とし，ソース・ヘッダ中の PROCESSOR 疑似命令を無視します。

保守／廃止

表11－2　アセンブラのエラー・メッセージ（8/10）

W703	メッセージ	Multiple defined module name
	原　　因	NAME疑似命令が二度以上定義されています。
	プログラムの処理	そのNAME疑似命令を無効として，すでに定義されたモジュール名を有効とします。
W704	メッセージ	Already declared EXTRN symbol 'シンボル名'
	原　　因	このシンボルは，すでにEXTRN宣言されています。
	ユーザの処理	1つのシンボルのEXTRN宣言は，1モジュールにつき1回にしてください。
W705	メッセージ	Already declared EXTBIT symbol 'シンボル名'
	原　　因	このシンボルは，すでにEXTBIT宣言されています。
	ユーザの処理	1つのシンボルのEXTBIT宣言は，1モジュールにつき1回にしてください。
W706	メッセージ	Missing END statement
	原　　因	ソース・ファイルの最後にEND文が記述されていません。
	プログラムの処理	ソース・ファイルの最後にEND文があったものと見なします。
W707	メッセージ	Illegal statement after END directive
	原　　因	END文のあとにコメント，空白，タブ，改行コード以外のものが記述されました。
	プログラムの処理	END文のあとの文を無視します。
W708	メッセージ	Already declared LOCAL symbol 'シンボル名'
	原　　因	このシンボルは，すでにLOCAL宣言されています。
	ユーザの処理	1つのシンボルのLOCAL宣言は，1マクロにつき1回にしてください。
W709	メッセージ	Few count of actual parameter
	原　　因	実パラメータの個数が仮パラメータの個数よりも少なく設定されています。
	プログラムの処理	足りない個数分，仮パラメータにはヌル・ストリングが与えられます。
W710	メッセージ	Over count of actual parameter
	原　　因	実パラメータの個数が仮パラメータの個数よりも多く設定されています。
	プログラムの処理	超過分の実パラメータは，無視されます。
W711	メッセージ	Too many errors to report
	原　　因	この行に対するエラーが多すぎます（エラーが6個以上）。
	プログラムの処理	6個目以降のエラー・メッセージは出力せずに処理を続けます。
W712	メッセージ	Insufficient cross-reference work area
	原　　因	クロスレファレンス・リストの出力処理を行うためのメモリが不足しています。
	プログラムの処理	クロスレファレンス・リストを出力せずに，処理を続けます。

A901	メッセージ	Can't open source file 'ファイル名'
	原　　因	ソース・ファイルがオープンできません。

保守／廃止

表 11-2　アセンブラのエラー・メッセージ（9/10）

A902	メッセージ	Can't open parameter file 'ファイル名'
	原　　因	パラメータ・ファイルがオープンできません。
A903	メッセージ	Can't open include file 'ファイル名'
	原　　因	インクルード・ファイルがオープンできません。
A904	メッセージ	Illegal include file 'ファイル名'
	原　　因	インクルード・ファイル名として，ドライブ名のみ，パス名のみ，デバイス型ファイル名のいずれかが指定されました。
A905	メッセージ	Can't open overlay file 'ファイル名'
	原　　因	オーバレイ・ファイルがオープンできません。
	ユーザの処置	オーバレイ・ファイルが正しいディレクトリ(MS-DOS ver2.11 を使用しているときはカレント・ディレクトリ，ver3.10 以上を使用しているときは実行形式プログラムがあるディレクトリ）にあることを確認してください。
A906	メッセージ	Illegal overlay file 'ファイル名'
	原　　因	オーバレイ・ファイルの内容が不正です。
A907	メッセージ	Can't open object file 'ファイル名'
	原　　因	オブジェクト・ファイルがオープンできません。
	ユーザの処置	ディレクトリに空きのあるディスクを使用してください。
A908	メッセージ	Can't open print file 'ファイル名'
	原　　因	アセンブル・リスト・ファイルがオープンできません。
	ユーザの処置	ディレクトリに空きのあるディスクを使用してください。
A909	メッセージ	Can't open error list file 'ファイル名'
	原　　因	エラー・リスト・ファイルがオープンできません。
	ユーザの処置	ディレクトリに空きのあるディスクを使用してください。
A910	メッセージ	Can't open temporary file 'ファイル名'
	原　　因	テンポラリ・ファイルがオープンできません。
	ユーザの処置	ディレクトリに空きのあるディスクを使用してください。
A911	メッセージ	System error
	原　　因	システム・エラーが発生しました。
A912	メッセージ	Can't set Control-C
	原　　因	アセンブラ実行中止のための，CTRL キー＋C の設定ができません。
A913	メッセージ	Can't read source file 'ファイル名'
	原　　因	ソース・ファイルにファイル I/O エラーが発生しました。
A914	メッセージ	Can't read parameter file 'ファイル名'
	原　　因	パラメータ・ファイルにファイル I/O エラーが発生しました。
A915	メッセージ	Can't read include file 'ファイル名'
	原　　因	インクルード・ファイルにファイル I/O エラーが発生しました。

表11-2　アセンブラのエラー・メッセージ（10/10）

A916	メッセージ	Can't read overlay file 'ファイル名'
	原　　因	オーバレイ・ファイルにファイルI/Oエラーが発生しました。
A917	メッセージ	Can't write object file 'ファイル名'
	原　　因	オブジェクト・ファイルにファイルI/Oエラーが発生しました。
	ユーザの処置	オブジェクト・ファイルを他のディスクに出力するか，指定したディスクに空き領域を作ってください。
A918	メッセージ	Can't write print file 'ファイル名'
	原　　因	アセンブル・リスト・ファイルにファイルI/Oエラーが発生しました。
	ユーザの処置	アセンブル・リスト・ファイルを他のディスクに出力するか，指定したディスクに空き領域を作ってください。
A919	メッセージ	Can't write error list file 'ファイル名'
	原　　因	エラー・リスト・ファイルにファイルI/Oエラーが発生しました。
	ユーザの処置	エラー・リスト・ファイルを他のディスクに出力するか指定したディスクに空き領域を作ってください。
A920	メッセージ	Can't read/write temporary file 'ファイル名'
	原　　因	テンポラリ・ファイルにファイルI/Oエラーが発生しました。
	ユーザの処置	テンポラリ・ファイルを他のディスクに出力するか指定したディスクに空き領域を作ってください。
A921	メッセージ	Assembler internal error
	原　　因	アセンブラ自身に内部エラーが発生しました。
	ユーザの処置	もう一度アセンブルを実行してください。
A922	メッセージ	Insufficient memory in hostmachine
	原　　因	システムにアセンブラを実行するための十分なメモリがありません。
A923	メッセージ	Insufficient memory for macro in hostmachine
	原　　因	マクロ処理の途中で内部メモリが不足しました。
	ユーザの処置	マクロ定義を少なくしてください。

保守／廃止

11.4　リンカのエラー・メッセージ

A0××のエラー・メッセージについては，“**11.2　RA78K/I共通のエラー・メッセージ**”を参照してください。

表11-3　リンカのエラー・メッセージ（1/6）

A101	メッセージ	'ファイル名' invalid input file (or made by different hostmachine)
	原　　因	オブジェクト・モジュール・ファイル以外のファイルを入力しようとしたか，互換性のないホスト・マシンで作成されたオブジェクト・モジュール・ファイルをリンクしようとしました。
F102	メッセージ	Directive syntax error
	原　　因	ディレクティブの記述が間違っています。
A103	メッセージ	'ファイル名' Illegal processor type
	原　　因	アセンブルまたはコンパイルの対象デバイスが，このリンカの対象デバイスではありません。
	ユーザの処置	オブジェクト・モジュール・ファイルが正しいことを確認してください。リンカの扱えるアセンブルまたはコンパイルの対象デバイスを確認してください。また，オーバレイ・ファイルが正しいバージョンであることを確認してください（リンカは，アセンブラのオーバレイ・ファイルの一部を参照して対象デバイス固有の情報を得ています）。
A104	メッセージ	'ファイル名' Different processor type from first input file '最初に入力したファイル名'
	原　　因	最初に入力したオブジェクト・モジュール・ファイルと対象デバイスの異なるオブジェクト・モジュール・ファイルを入力しました。
W105	メッセージ	Library file 'ファイル名' has no public symbol
	原　　因	ライブラリ・ファイルにパブリック・シンボルが存在しません。そのため，ライブラリ・ファイル中に含まれるオブジェクト・モジュールはリンクされません。
A106	メッセージ	Can't create temporary file 'ファイル名'
	原　　因	テンポラリ・ファイルが作成できません。
F107	メッセージ	Name '名前' in directive has already defined
	原　　因	ディレクティブのメモリ領域名として，予約語または，すでに定義している名前を定義しようとしました。 この名前（予約語，メモリ空間名，メモリ領域名）は，すでに登録されています。
F108	メッセージ	Overlapped memory area 'メモリ領域1' and 'メモリ領域2'
	原　　因	メモリ・ディレクティブでメモリ領域のアドレスが重複しています。
F109	メッセージ	Memory area 'メモリ領域名' too long name (up to 31 characters)
	原　　因	ディレクティブ中でのメモリ領域名の指定が長すぎます。 ディレクティブ中でのメモリ領域名の長さの制限は31文字以下です。

表 11－3　リンカのエラー・メッセージ（2/6）

F110	メッセージ	Memory area 'メモリ領域名' already defined
	原　　因	メモリ・ディレクティブで指定されたメモリ領域名は，すでに登録されています。
F111	メッセージ	Memory area 'メモリ領域名' redefinition out of range
	原　　因	メモリ・ディレクティブで指定されているメモリ領域の範囲は再定義可能な範囲を越えています。
F112	メッセージ	Segment 'セグメント名' wrong allocation type
	原　　因	マージ・ディレクティブでセグメントの配置型の指定が間違っています。
F113	メッセージ	Linker internal error
	原　　因	内部エラー
	ユーザの処置	特約店または日本電気までご連絡ください。
F114	メッセージ	Illegal number
	原　　因	ディレクティブ中の数値の記述に誤りがあります。
F115	メッセージ	Too large value (up to 65535/0FFFFH)
	原　　因	ディレクティブ中で65535（0FFFFH）を越える値が記述されました。
F116	メッセージ	Memory area 'メモリ領域名' definition out of range
	原　　因	メモリ・ディレクティブにおいて，メモリ領域の先頭アドレスとサイズの和が，65535（0FFFFH）を越えました。
F117	メッセージ	Too many line number data in 'ファイル名'
	原　　因	ディバク情報であるライン・ナンバ・データを入力し，処理を続行します。なお，－Jオプション指定時のみオブジェクト・ファイルを出力します。

F201	メッセージ	Multiple segment definition 'セグメント名' in merge directive
	原　　因	マージ・ディレクティブで指定されたセグメントは，すでに登録されています(同じセグメントを複数のマージ・ディレクティブで割り付け指定しようとしています)。
F202	メッセージ	Segment type mismatch 'セグメント1' in file 'セグメント2' - ignored
	原　　因	このセグメントと同じ名前で，異なるセグメント・タイプの再配置属性を持つセグメントが存在しています。
A203	メッセージ	Segment 'セグメント名' unknown segment type
	原　　因	入力したオブジェクト・モジュール・ファイルのセグメント情報に誤りがあります（出力セグメントの結合型の指定が間違っています)。
F204	メッセージ	Memory area/space '名前' not defined
	原　　因	マージ・ディレクティブで指定されたメモリ領域名/メモリ空間名は定義されていません。

保守／廃止

表11-3　リンカのエラー・メッセージ（3/6）

F205	メッセージ	Name '名前' in directive has bad attribute
	原　　因	ディレクティブのセグメント名，メモリ領域名，メモリ空間名のいずれかに指定できないものを記述しています（メモリ領域名を指定すべきところにメモリ空間名を指定したなど）。
F206	メッセージ	Segment 'セグメント名' can't allocate to memory - ignored
	原　　因	セグメントをメモリ領域に割り付けることができません（セグメントを割り付けるのに十分なメモリ領域がありません）。
F207	メッセージ	Segment 'セグメント名' has illegal segment type
	原　　因	このセグメントの型情報が不正です。
	ユーザの処置	このセグメントが含まれるソース・プログラムを，もう一度アセンブルまたはコンパイルしなおしてください。 マージ・ディレクティブの記述を確認してください。
F208	メッセージ	Segment 'セグメント名' may not change attribute
	原　　因	アセンブル時に再配置属性を 'AT xxxxH' としたセグメント，またはORG疑似命令により作成したセグメントに対し，ディレクティブで結合型を変更しようとしました。
	ユーザの処置	リンク時に結合型を指定するセグメントに対しては，アセンブル時に配置アドレスを指定しないでください。
F209	メッセージ	Segment 'セグメント名' may not change arrangement
	原　　因	アセンブル時に再配置属性を 'AT xxxxH' としたセグメント，またはORG疑似命令により作成したセグメントに対し，ディレクティブで配置アドレスを変更しようとしました。
	ユーザの処置	リンク時に結合型を指定するセグメントに対しては，アセンブル時に配置アドレスを指定しないでください。
F210	メッセージ	Segment 'セグメント名' does not exist - ignored
	原　　因	ディレクティブで指定されたセグメントが存在しません。

F301	メッセージ	Relocatable object code address out of range (file 'ファイル名', segment 'セグメント名', address xxxxH, type 'アドレシング・タイプ')
	原　　因	入力したオブジェクト・モジュール・ファイル中に含まれるリロケータブル・オブジェクト・コードの修正情報が，オブジェクト・コードの存在しないアドレスに対して出力されています(リロケーション・エントリのアドレスが，オリジン・データの範囲外にあります)。
	ユーザの処置	シンボルの参照が正しいことを確認してください。

表 11－3　リンカのエラー・メッセージ（4/6）

F303	メッセージ	Can't find symbol index in relocatable object code (file 'ファイル名', segment 'セグメント名', address xxxxH, type 'アドレシング・タイプ')
	原　　因	入力したオブジェクト・モジュール・ファイル中に含まれるリロケータブル・コードの修正情報に誤りがあり，シンボル情報を正しく参照していません。 リロケーション・エントリとシンボル・インデクスとの対応がとれていません。
	ユーザの処置	シンボル・変数などの参照方法が正しいことを確認してください。
F304	メッセージ	Operand out of range (segment 'セグメント名', address xxxxH, type 'アドレシング・タイプ')
	原　　因	リロケータブル・オブジェクト・コードの解決に用いているオペランド値が，命令に対応したオペランド値の範囲を越えています。
	ユーザの処置	オペランド値をアドレシング・タイプごとに定められているオペランド値の範囲に納まるようにソース・プログラムを記述してください。
F305	メッセージ	Even value expected (segment 'セグメント名', address xxxxH, type 'アドレシング・タイプ')
	原　　因	callt または saddrp アドレシングのリロケータブル・オブジェクト・コードの解決に用いているオペランド値が奇数になりました（callt または saddrp アドレシングのオペランドは偶数でなければなりません）。

注意　F301～F305 のメッセージの中で，'address xxxxH' として表示されるアドレスは，セグメント配置後の絶対アドレスです。

A401	メッセージ	'ファイル名' Bad symbol table
	原　　因	入力したオブジェクト・モジュール・ファイルのシンボル情報が不正です。 入力ファイルのシンボル・エントリが '.file' シンボルで始まっていません。
A402	メッセージ	File 'ファイル名' has no string table for symbol
	原　　因	入力したオブジェクト・モジュール・ファイルのシンボル情報が不正です。
	ユーザの処置	もう一度アセンブルまたはコンパイルしなおしてください。 アセンブラのシンボル認識文字数を 8 文字，コンパイラの認識文字数を 7 文字にすることで回避可能な場合があります。
A403	メッセージ	Symbol 'シンボル名' unmatched type in file 'ファイル名 1' First defined in file 'ファイル名 2'
	原　　因	同名外部定義/参照シンボルの型がファイル 1 とファイル 2 で異なっています。
F404	メッセージ	Multiple Symbol definition 'シンボル名' in file 'ファイル名 1' First defined in file 'ファイル名 2'
	原　　因	オブジェクト・モジュール・ファイル 1 中で定義されている PUBLIC シンボルは，オブジェクト・モジュール・ファイル 2 ですでに PUBLIC 宣言されています。

保守／廃止

表 11-3　リンカのエラー・メッセージ（5/6）

F405	メッセージ	Undefined symbol 'シンボル名' in file 'ファイル名'
	原　　因	ファイルで EXTERN 宣言されているシンボルは，他のファイルで PUBLIC 宣言されていません。
W406	メッセージ	Stack area less than 10 bytes
	原　　因	確保したスタック領域の大きさが 10 バイト以下です（−S オプションで指定されたメモリ領域に確保できたスタック領域の大きさが 10 バイト以下です）。
W407	メッセージ	Can't allocate stack area
	原　　因	スタック領域を確保するメモリ領域に，空き領域がありません（−S オプションで指定されたメモリ領域にスタック領域を確保できません）。
F410	メッセージ	Multiple module name definition 'モジュール名' in file 'ファイル名 1'. First defined in file 'ファイル名 2'
	原　　因	入力したファイル（ファイル名 1 およびファイル名2）のモジュール名が重複しています。

A501	メッセージ	Insufficient memory in hostmachine
	原　　因	システムにプログラムが動作するための十分なメモリがありません。

A901	メッセージ	Can't open overlay file 'ファイル名'
	原　　因	オーバレイ・ファイルがオープンできません。
	ユーザの処置	オーバレイ・ファイルが正しいディレクトリ（MS-DOS ver 2.11 を使用しているときはカレント・ディレクトリ，ver 3.10 以上を使用しているときは実行形式プログラムがあるディレクトリ）にあることを確認してください。
A902	メッセージ	file 'ファイル名' file not found
	原　　因	指定されたライブラリ・ファイルをオープンできません。
A903	メッセージ	Can't read input file 'ファイル名'
	原　　因	入力ファイルとして指定されたオブジェクト・モジュール・ファイルを読むことができません。
A904	メッセージ	Can't open output file 'ファイル名'
	原　　因	出力ファイルをオープンできません。
	ユーザの処置	出力ファイルを作成しようとしたディスクの状態（空き容量，メディアの状態など）を確認してください。

保守／廃止

表 11－3　リンカのエラー・メッセージ（6/6）

A905	メッセージ	Can't create temporary file 'ファイル名'
	原　　因	シンボル・エントリ用のテンポラリ・ファイルを作成することができません。
	ユーザの処置	テンポラリ・ファイルを作成しようとしたディスクの状態（空き容量，メディアの状態など）を確認してください。
A906	メッセージ	Can't write map file 'ファイル名'
	原　　因	リンク・リスト・ファイルにデータを書き込めません。
	ユーザの処置	リンク・リスト・ファイルを作成しようとしたディスクの状態(空き容量,メディアの状態など）を確認してください。
A907	メッセージ	Can't write output file 'ファイル名'
	原　　因	ロード・モジュール・ファイルに書き込みができません。
	ユーザの処置	出力ファイルを作成しようとしたディスクの状態（空き容量，メディアの状態など）を確認してください。
A908	メッセージ	Can't access temporary file 'ファイル名'
	原　　因	テンポラリ・ファイルに書き込みができません。
	ユーザの処置	テンポラリ・ファイルを作成しようとしたディスクの状態（空き容量，メディアの状態など）を確認してください。

保守／廃止

11.5　オブジェクト・コンバータのエラー・メッセージ

A0××のエラー・メッセージについては，“**11.2　RA78K/I共通のエラー・メッセージ**”を参照してください。

表11-4　オブジェクト・コンバータのエラー・メッセージ（1/2）

A100	メッセージ	'ファイル名' Illegal processor type
	原　　因	アセンブルまたはコンパイルの対象デバイスが，このプログラムの対象デバイスと異なります。
	ユーザの処置	ロード・モジュール・ファイルが正しいかどうか，そしてアセンブルまたはコンパイルの対象デバイスを確認してください。また，オーバレイ・ファイルのバージョンが正しいかどうかを確認してください。
A101	メッセージ	'ファイル名' invalid input file (or made by different hostmachine)
	原　　因	ロード・モジュール・ファイル以外のファイルを入力しようとしたか，または互換性のないホスト・マシンで作成されたロード・モジュール・ファイルをコンバートしようとしました。
A103	メッセージ	Symbol 'シンボル名' Illegal attribute
	原　　因	入力ファイルのシンボル属性の値に誤りがあります。
A104	メッセージ	'ファイル名' Illegal input file - not linked
	原　　因	オブジェクト・モジュール・ファイルを入力しようとしています。
A105	メッセージ	Insufficient memory in hostmachine
	原　　因	プログラムが動作するために十分なメモリがありません。
A106	メッセージ	Illegal symbol table
	原　　因	入力したロード・モジュール・ファイルのシンボル・テーブルに誤りがあります。

F200	メッセージ	Undefined symbol 'シンボル名'
	原　　因	アドレスが解決していないシンボルがあります。
	ユーザの処置	シンボルの値を定義してください。 このシンボルを外部参照シンボルとして参照しますが，外部定義していないときはシンボル値を定義しているモジュールで外部定義してください。
F201	メッセージ	Out of address range
	原　　因	ロード・モジュール・ファイルのオブジェクトのアドレスが範囲を越えています。

保守／廃止

表 11-4　オブジェクト・コンバータのエラー・メッセージ（2/2）

W300	メッセージ	xxxxH - yyyyH overlapped
	原　　因	xxxxH から yyyyH までのアドレスに対するオブジェクトが重複して出力されています。

A900	メッセージ	Can't open file 'ファイル名'
	原　　因	ファイルがオープンできません。
A901	メッセージ	Can't close file 'ファイル名'
	原　　因	ファイルがクローズできません。
A902	メッセージ	Can't read file 'ファイル名'
	原　　因	ファイルが正しく読めません。
A903	メッセージ	Can't access file 'ファイル名'
	原　　因	ファイルを正しく読み込みまたは書き込みができません。
A904	メッセージ	Can't write file 'ファイル名'
	原　　因	出力ファイルに正しくデータが書き込めません。
A905	メッセージ	Can't open overlay file 'ファイル名'
	原　　因	オーバレイ・ファイルがオープンできません。
	ユーザの処置	オーバレイ・ファイルが正しいディレクトリ(MS-DOS ver2.11 を使用しているときはカレント・ディレクトリ, ver3.10 以上を使用しているときは実行形式プログラムがあるディレクトリ）にあることを確認してください。

保守／廃止

11.6　ライブラリアンのエラー・メッセージ

A0××のエラー・メッセージについては，“**11.2　RA78K/I共通のエラー・メッセージ**”を参照してください。

表11-5　ライブラリアンのエラー・メッセージ（1/2）

A100	メッセージ	Internal error
	原　　因	内部エラーが発生しました。
F101	メッセージ	Invalid sub command
	原　　因	サブコマンド名が誤っています。
F102	メッセージ	Invalid syntax
	原　　因	サブコマンドのパラメータ指定に誤りがあります。
F103	メッセージ	Illegal input file - different target chip (file：ファイル名)
	原　　因	入力したオブジェクト・モジュール・ファイルの対象デバイス指定に誤りがあります。
F104	メッセージ	Illegal library file - different target chip (file：ファイル名)
	原　　因	指定ライブラリ・ファイルの対象デバイスに誤りがあります。
F105	メッセージ	Module not found (module：モジュール名)
	原　　因	指定モジュールがライブラリ・ファイル中に存在しません。
F106	メッセージ	Module already exists (module：モジュール名)
	原　　因	同名のモジュールが，すでに更新ライブラリ・ファイルまたは他の入力ファイル内に存在しています。
F107	メッセージ	Master library file is not specify
	原　　因	以前のオペレーションで，まだ更新ライブラリ・ファイルの指定がされていないのに，‘. ’での置き換えが指定されました。
F108	メッセージ	Multiple transaction file (file：ファイル名)
	原　　因	入力オブジェクト・モジュール・ファイル名が重複しています。
F109	メッセージ	Public symbol already exists (symbol：シンボル名)
	原　　因	同名の外部定義シンボル名が，すでに更新ライブラリ・ファイルまたは他の入力ファイル内に存在しています。
F110	メッセージ	Flie specification conflicted (file：ファイル名)
	原　　因	指定した入力ファイル名と出力ファイル名が，一致しています。

表 11－5　ライブラリアンのエラー・メッセージ（2/2）

F111	メッセージ	Illegal file format（file：ファイル名）
	原　　因	更新ライブラリ・ファイルまたは，他の入力ファイルのフォーマットが異常です。
F112	メッセージ	Library file not found（file：ファイル名）
	原　　因	指定したライブラリ・ファイルが見つかりません。
F113	メッセージ	Object module file not found（file：ファイル名）
	原　　因	指定したオブジェクト・モジュール・ファイルが見つかりません。
F114	メッセージ	No free space for temporary file
	原　　因	ディスク上にテンポラリ・ファイルを作成するための十分な空き容量がありません。
F115	メッセージ	Not enough memory
	原　　因	プログラムが動作するための，十分なメモリが確保できません。
F116	メッセージ	Sub command Buffer full
	原　　因	サブコマンドの継続行の長さが制限（128×15 文字）を越えています。 サブコマンド・ファイル中のサブコマンドの 1 行の長さが制限(128 文字)を越えています。
F117	メッセージ	Can't use device file
	原　　因	ファイル名として，デバイス型ファイルを指定しています。 （デバイス型ファイル名：＊.con，＊.prn，＊.nul）
F118	メッセージ	Illegal path（file：ファイル名）
	原　　因	指定したファイル（名称：ファイル名）の名称中に，存在しないパス名またはドライブ名があります。

A901	メッセージ	File open error（file：ファイル名）
	原　　因	ファイルに誤りがあるか，またはシステムが正常に動作していません。
F902	メッセージ	File read error（file：ファイル名）
	原　　因	ファイルに誤りがあるか，またはシステムが正常に動作していません。
A903	メッセージ	File write error（file：ファイル名）
	原　　因	ファイルに誤りがあるか，またはシステムが正常に動作していません。
A904	メッセージ	File seek error（file：ファイル名）
	原　　因	ファイルに誤りがあるか，またはシステムが正常に動作していません。
A905	メッセージ	File close error（file：ファイル名）
	原　　因	ファイルに誤りがあるか，またはシステムが正常に動作していません。

保守／廃止

11.7　リスト・コンバータのエラー・メッセージ

A0××のエラー・メッセージについては，“**11.2　RA78K/I共通のエラー・メッセージ**”を参照してください。

表11-6　リスト・コンバータのエラー・メッセージ（1/2）

A101	メッセージ	File is not 78K/I 'ファイル名'
	原　　因	入力ファイル名が78K/Iのものではありません。
W101	メッセージ	Load module file is older than object module file 'ロード・モジュール・ファイル名，オブジェクト・モジュール・ファイル名'
	原　　因	オブジェクト・モジュール・ファイルよりも古いロード・モジュール・ファイルが指定されました。
A102	メッセージ	Load module file is not executable 'ファイル名'
	原　　因	ロード・モジュール・ファイル以外のファイルを入力しようとしたか，互換性のないホスト・マシンで作成されたロード・モジュール・ファイルをコンバートしようとしました。
W102	メッセージ	Load module file is older than assemble list file 'ロード・モジュール・ファイル名，アセンブル・リスト・ファイル名'
	原　　因	アセンブル・リスト・ファイルよりも古いロード・モジュール・ファイルが指定されました。
A103	メッセージ	Load module file has relocation data 'ファイル名'
	原　　因	ロード・モジュール・ファイルのアドレスが解決されていません。
W103	メッセージ	Assemble list has error statement 'ファイル名'
	原　　因	アセンブル・リスト内にエラー行があります。
A104	メッセージ	Object module file is executable 'ファイル名'
	原　　因	オブジェクト・モジュール・ファイルが実行形式です。
W104	メッセージ	Segment name is not found in assemble list file 'セグメント名'
	原　　因	アセンブル・リスト内にオブジェクト・モジュール・ファイルのセグメント名が見つかりません。
A105	メッセージ	Segment name is not found in load module file 'セグメント名'
	原　　因	ロード・モジュール・ファイル内にオブジェクト・モジュール・ファイルのセグメント名が見つかりません。

保守／廃止

表 11-6　リスト・コンバータのエラー・メッセージ（2/2）

W105	メッセージ	Segment data length is different 'セグメント名'
	原　　因	アセンブル・リスト・ファイル上のセグメントのデータの長さとオブジェクト・モジュール・ファイル上のデータの長さが異なります。
	プログラムの処理	余分なセグメントのデータは無視して処理を実行します。
A106	メッセージ	Segment name is not found in object module file 'セグメント名'
	原　　因	オブジェクト・モジュール・ファイル内にアセンブル・リスト・ファイルのセグメント名が見つかりません。
A107	メッセージ	Not enough memory
	原　　因	作業用メモリが足りません。
A108	メッセージ	Load module file has no symbol data 'ロード・モジュール名'
	原　　因	リンカで－NGオプションを指定したためロード・モジュール中にシンボル情報が出力されていません。

A901	メッセージ	File open error has occurred 'ファイル名'
	原　　因	ファイルがオープンできません。
A902	メッセージ	File read error has occurred 'ファイル名'
	原　　因	ファイルが正しく読めません。
A903	メッセージ	File write error has occurred 'ファイル名'
	原　　因	ファイルに正しくデータが書き込めません。
A904	メッセージ	File seek error has occurred
	原　　因	ファイル・シーク・エラーが発生しました。
A905	メッセージ	Overlay file cannot open 'ファイル名'
	原　　因	オーバレイ・ファイルがオープンできません。
	ユーザの処置	オーバレイ・ファイルが正しいディレクトリ（MS-DOS ver2.11を使用しているときはカレント・ディレクトリ，ver3.10以上を使用しているときは実行形式プログラムがあるディレクトリ）にあることを確認してください。
A999	メッセージ	Internal error
	原　　因	プログラム内部エラー

保守／廃止

付録 A　サンプル・プログラム

以下に示すRA78K/Iで使用するプログラムなどのサンプル・リストを紹介します。

○ソース・リスト

○実行例

○出力リスト

アセンブル・リスト

シンボル・リスト

クロスレファレンス・リスト

マップ・リスト

パブリック・シンボル・リスト

ローカル・シンボル・リスト

ライブラリ情報出力リスト

アブソリュート・アセンブル・リスト

A.1　ソース・リスト

(1) 78K1MAIN. ASM

```
A>type 78k1main.asm

$       PROCESSOR(138)

        NAME    SAMPM
;*************************************************
;*                                               *
;*      HEX -> ASCII Conversion Program          *
;*                                               *
;*              main-routine                     *
;*                                               *
;*************************************************

        PUBLIC  MAIN,START
        EXTRN   CONVAH

        DSEG
STASC:  DS      2

        CSEG
        ORG     0H
MAIN:   DW      START

        CSEG
START:  MOVW    SP,#0FEE0H
        MOV     MM,#00
        MOV     STBC,#00

        MOV     E,#LOW HDTSA            ;set hex 2-code data in HL registor

        CALL    !CONVAH                 ;convert ASCII <- HEX
                                        ;output BC-register <- ASCII code
        MOVW    DE,#STASC               ;set DE <- store ASCII code table
        MOV     A,B
        MOV     [DE+],A
        MOV     A,C
        MOV     [DE+],A

        BR      $$

DATA    DSEG    AT 0FE00H
HDTSA:  DB      1AH

        END
```

(2) 78K1SUB. ASM

```
A>type 78k1sub.asm

$       PROCESSOR(138)

        NAME    SAMPS
;*******************************************************
;*                                                     *
;*   HEX -> ASCII Conversion Program                   *
;*                                                     *
;*              sub-routine                            *
;*                                                     *
;* input condition  : (FE00H + E-reg) <- hex 2 code    *
;*                                                     *
;* output condition : BC-reg <-ASCII 2 code            *
;*                                                     *
;*******************************************************

        PUBLIC  CONVAH

        CSEG
CONVAH: MOV     A,#0
        ROL4    [E]             ;hex upper code load
        CALL    !SASC
        MOV     B,A             ;store result

        MOV     A,#0
        ROL4    [E]             ;hex lower code load
        CALL    !SASC
        MOV     C,A             ;store result

        RET

;*******************************************************
;* subroutine   convert ASCII code                     *
;*      input   Acc (lower 4bits) <- hex code          *
;*      output  Acc               <- ASCII code        *
;*******************************************************

        CSEG
SASC:   CMP     A,#0AH          ;check hex code > 9
        BC      $SASC1
        ADD     A,#07H          ;bias(+7)
SASC1:  ADD     A,#30H          ;bias(+30)
        RET

        END
```

A

A.2　実 行 例

```
A>ra78k1 -c138 78k1main.asm -g -kx -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.


A>ra78k1 -c138 78k1sub.asm -g -kx -lw80

78K/I Series Assembler Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1 Start
Pass2 Start

Assembly complete,      0 error(s) and      0 warning(s) found.


A>lk78k1 78k1main.rel 78k1sub.rel -g -o78k1.map -kp -kl

78K/I Series Linker Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1989

Link complete,      0 error(s) and      0 warning(s) found.


A>oc78k1 78k1.lnk

78K/I Series Object Converter Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Object Conversion Complete,       0 error(s) and      0 warning(s) found.


A>lb78k1

78K/I Series Librarian Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990
*create 78k1.lib
*add 78k1.lib 78k1main.rel 78k1sub.rel
*exit


A>78k1 78k1main -l78k1.lnk

List Conversion Program for RA78K/I Vx.xx  [xx xxx xx]
   Copyright (C) NEC Corporation 1990

Pass1: start....
Pass2: start.....
Conversion complete.
```

保守／廃止

A.3　出力リスト

A.3.1　アセンブル・リスト

(1) 78K1MAIN. ASM のアセンブル・リスト

```
78K/I Series Assembler Vx.xx                    Date:xx xxx xxxx Page:   1


Command:  -c138 78k1main.asm -g -kx -lw80
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

      Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                    $       PROCESSOR(138)
    2     2
    3     3                            NAME    SAMPM
    4     4                    ;*********************************************
                               *
    5     5                    ;*
                               *
    6     6                    ;*      HEX -> ASCII Conversion Program
                               *
    7     7                    ;*
                               *
    8     8                    ;*              main-routine
                               *
    9     9                    ;*
                               *
   10    10                    ;*********************************************
                               *
   11    11
   12    12                            PUBLIC  MAIN,START
   13    13                            EXTRN   CONVAH
   14    14
   15    15 ----                       DSEG
   16    16 0000               STASC:  DS      2
   17    17
   18    18 ----                       CSEG
   19    19 ----                       ORG     0H
   20    20 0000 R0000         MAIN:   DW      START
   21    21
   22    22 ----                       CSEG
   23    23 0000  0BFCE0FE     START:  MOVW    SP,#0FEE0H
   24    24 0004  2BC400               MOV     MM,#00
   25    25 0007  09C0FF00             MOV     STBC,#00
   26    26
   27    27 000B  BC00                 MOV     E,#LOW HDTSA            ;set hex
                                2-code data in HL registor
   28    28
   29    29 000D R280000               CALL    !CONVAH                 ;convert
                                ASCII <- HEX

                                   ⋮

Target chip:uPD78138
Assembly complete,     0 error(s) and     0 warning(s) found. (    0)
```

(2) 78K1SUB. ASM のアセンブル・リスト

```
78K/I Series Assembler Vx.xx                    Date:xx xxx xxxx Page:   1

Command:  -c138 78k1sub.asm -g -kx
Para-file:
In-file:  78K1SUB.ASM
Obj-file: 78K1SUB.REL
Prn-file: 78K1SUB.PRN

     Assemble list

 ALNO  STNO ADRS  OBJECT   M I  SOURCE STATEMENT

    1     1                      $       PROCESSOR(138)
    2     2
    3     3                              NAME    SAMPS
    4     4                      ;*****************************************************
    5     5                      ;*                                                   *
    6     6                      ;*   HEX -> ASCII Conversion Program                 *
    7     7                      ;*                                                   *
    8     8                      ;*              sub-routine                          *
    9     9                      ;*                                                   *
   10    10                      ;* input condition  : (FE00H + E-reg) <- hex 2 code *
   11    11                      ;*                                                   *
   12    12                      ;* output condition : BC-reg <-ASCII 2 code          *
   13    13                      ;*                                                   *
   14    14                      ;*****************************************************
   15    15
   16    16                              PUBLIC  CONVAH
   17    17
   18    18 ----                         CSEG
   19    19 0000  B900           CONVAH: MOV     A,#0
   20    20 0002  0599                   ROL4    [E]
   21    21 0004 R281300                 CALL    !SASC
   22    22 0007  2431                   MOV     B,A
   23    23
   24    24 0009  B900                   MOV     A,#0
   25    25 000B  0599                   ROL4    [E]
   26    26 000D R281300                 CALL    !SASC
   27    27 0010  2421                   MOV     C,A
   28    28
   29    29 0012  56                     RET
   30    30
   31    31                      ;*****************************************************
   32    32                      ;* subroutine   convert ASCII code                  *
   33    33                      ;*      input   Acc (lower 4bits) <- hex code       *
   34    34                      ;*      output  Acc               <- ASCII code     *
   35    35                      ;*****************************************************
   36    36
   37    37 ----                         CSEG
   38    38 0013  AF0A           SASC:   CMP     A,#0AH          ;check hex code
          :

Segment informations:

ADRS  LEN      NAME
0000  001CH    ?CSEG
          :

Target chip:uPD78138
Assembly complete,     0 error(s) and     0 warning(s) found. (    0)
```

A.3.2　シンボル・リスト

(1) 78K1MAIN. ASMのシンボル・リスト

```
                Symbol Table List

VALUE  ATTR  RTYP NAME          VALUE  ATTR  RTYP NAME

       CSEG       ?ASEG1              CSEG       ?CSEG
       DSEG       ?DSEG        ----H         EXT  CONVAH
       DSEG       DATA         FE00H  ADDR       HDTSA
  0H   ADDR  PUB  MAIN                 MOD        SAMPM
  0H   ADDR  PUB  START           0H   ADDR       STASC
```

(2) 78K1SUB. ASMのシンボル・リスト

```
                Symbol Table List

VALUE  ATTR  RTYP NAME          VALUE  ATTR  RTYP NAME

       CSEG       ?CSEG          0H   ADDR  PUB  CONVAH
       MOD        SAMPS         13H   ADDR       SASC
 19H   ADDR       SASC1
```

A.3.3　クロスレファレンス・リスト

(1) 78K1MAIN. ASMのクロスレファレンス・リスト

```
                Cross-Reference List

NAME      VALUE  R ATTR  RTYP SEGNAME  XREFS

?ASEG1              CSEG       ?ASEG1      19#
?CSEG               CSEG       ?CSEG       18#    22#
?DSEG               DSEG       ?DSEG       15#
CONVAH    ----H  E       EXT              13@    29
DATA                DSEG       DATA        39#
HDTSA     FE00H     ADDR       DATA        27     40#
MAIN        0H      ADDR  PUB  ?ASEG1      12@    20#
SAMPM               MOD                     3#
START       0H   R ADDR  PUB  ?CSEG       12@    20     23#
STASC       0H   R ADDR       ?DSEG       16#    31
```

A

(2) 78K1SUB. ASM のクロスレファレンス・リスト

```
                Cross-Reference List

NAME        VALUE  R ATTR  RTYP SEGNAME  XREFS

?CSEG               CSEG       ?CSEG       18#    37#
CONVAH       0H     R ADDR  PUB  ?CSEG       16@    19#
SAMPS               MOD                     3#
SASC        13H     R ADDR       ?CSEG       21     26     38#
SASC1       19H     R ADDR       ?CSEG       39     41#
```

A.3.4　マップ・リスト

```
78K/I Series Linker Vx.xx                    Date:xx xxx xxxx Page:   1

Command:   78k1main.rel 78k1sub.rel -g -o78k1.lnk -p78k1.map -kp -k1
Para-file:
Out-file:  78K1.LNK
Map-file:  78K1.MAP
Direc-file:
Directive:


*** Link information ***

     4 output segment(s)
   38H byte(s) real data
    18 symbol(s) defined


*** Memory map ***


  SPACE=REGULAR

  MEMORY=ROM
  BASE ADDRESS=0000H    SIZE=8000H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?ASEG1                       0000H     0002H   CSEG AT
                  ?ASEG1   SAMPM      0000H     0002H
* gap *                               0002H     007EH
         ?CSEG                        0080H     0035H   CSEG
                  ?CSEG    SAMPM      0080H     0019H
                  ?CSEG    SAMPS      0099H     001CH
* gap *                               00B5H     7F4BH

  MEMORY=RAM
  BASE ADDRESS=FC80H    SIZE=0280H
         OUTPUT   INPUT    INPUT      BASE      SIZE
         SEGMENT  SEGMENT  MODULE     ADDRESS
         ?DSEG                        FC80H     0002H   DSEG
                  ?DSEG    SAMPM      FC80H     0002H
* gap *                               FC82H     017EH
         DATA                         FE00H     0001H   DSEG AT
                  DATA     SAMPM      FE00H     0001H
* gap *                               FE01H     00FFH
```

A.3.5　パブリック・シンボル・リスト

```
*** Public symbol list ***

MODULE    ATTR       VALUE    NAME

SAMPM     ADDR       0000H    MAIN
SAMPM     ADDR       0080H    START
SAMPS     ADDR       0099H    CONVAH
```

A.3.6　ローカル・シンボル・リスト

```
*** Local symbol list ***

MODULE    ATTR       VALUE    NAME

SAMPM     MOD                 SAMPM
SAMPM     DSEG                ?DSEG
SAMPM     ADDR       FC80H    STASC
SAMPM     CSEG                ?CSEG
SAMPM     CSEG                ?ASEG1
SAMPM     ADDR       FE00H    HDTSA
SAMPM     DSEG                DATA
SAMPS     MOD                 SAMPS
SAMPS     CSEG                ?CSEG
SAMPS     ADDR       00ACH    SASC
SAMPS     ADDR       00B2H    SASC1
```

A.3.7　ライブラリ情報出力リスト

```
78K/I Series librarian Vx.xx    DATE : xx xxx xx                    PAGE   1

LIB-FILE NAME : 78K1.LIB        (xx xxx xx)

0001  78K1MAIN.REL    (xx xxx xx)

  MAIN                          START

  NUMBER OF PUBLIC SYMBOLS :   2

0002  78K1SUB.REL     (xx xxx xx)

  CONVAH

  NUMBER OF PUBLIC SYMBOLS :   1
```

A.3.8　アブソリュート・アセンブル・リスト

```
78K/I Series Assembler Vx.xx                    Date:xx xxx xxxx Page:   1


Command:  -C138 78K1MAIN.ASM -ka
Para-file:
In-file:  78K1MAIN.ASM
Obj-file: 78K1MAIN.REL
Prn-file: 78K1MAIN.PRN

     Assemble list

 ALNO  STNO ADRS  OBJECT    M I  SOURCE STATEMENT

    1     1                     $       PROCESSOR(138)
    2     2
    3     3                             NAME    SAMPM
    4     4                     ;*********************************************
    5     5                     ;*                                           *
    6     6                     ;*      HEX -> ASCII Conversion Program      *
    7     7                     ;*                                           *
    8     8                     ;*              main-routine                 *
    9     9                     ;*                                           *
   10    10                     ;*********************************************
   11    11
   12    12                             PUBLIC  MAIN,START
   13    13                             EXTRN   CONVAH
   14    14
   15    15 ----                        DSEG
   16    16 0000                STASC:  DS      2
   17    17
   18    18 ----                        CSEG
   19    19 ----                        ORG     0H
   20    20 0000 R0000          MAIN:   DW      START
   21    21
   22    22 ----                        CSEG
   23    23 0000  0BFCE0FE      START:  MOVW    SP,#0FEE0H
   24    24 0004  2BC400                MOV     MM,#00
   25    25 0007  09C0FF00              MOV     STBC,#00
   26    26
   27    27 000B  BC00                  MOV     E,#LOW HDTSA  ;set hex 2-code data in HL registor
   28    28
   29    29 000D R280000                CALL    !CONVAH       ;convert ASCII <- HEX
   30    30                                                   ;output BC-register <- ASCII code
   31    31 0010 R640000                MOVW    DE,#STASC     ;set DE <- store ASCII code table
   32    32 0013  D3                    MOV     A,B
   33    33 0014  50                    MOV     [DE+],A
   34    34 0015  D2                    MOV     A,C
   35    35 0016  50                    MOV     [DE+],A

                                              ⋮

Target chip:uPD78138
Assembly complete,      0 error(s) and      0 warning(s) found. (    0)
```

保守／廃止

保守／廃止

付録 B　使用上の注意一覧

RA78K/I を使用するときの注意事項を示します。

保守／廃止

(1) MS-DOS（PC DOS）に関する注意事項：

●RA78K/I の各プログラムを起動する際は，環境設定ファイル 'CONFIG. SYS' の中の 'FILES=' の設定を 13 以上（FILES＝13）にしてください。

●内部メモリは 384 K バイト以上必要です。

(2) カレント・ディレクトリに関する注意事項：

●MS-DOS ver 2.11 を使用する場合，すべてのプログラム・ファイルはカレント・ディレクトリになければなりません。

(3) シンボル数の制限：

●アセンブラで指定できるシンボル数の上限は，ローカル・シンボルとパブリック・シンボルを合わせて 2900 までです。

●リンカでは，シンボル数の制限はありません（なお入力ファイル数は218までです）。

(4) アセンブラに入力可能なセグメント数の制限：

●ORG 疑似命令で定義したセグメントは20個です。

●CSEG，DSEG および BSEG 疑似命令で定義したセグメントは80個です。

(5) オプションに関する注意事項：

●指定した全てが有効となるオプションを除き，同一のオプションを複数指定した場合は，最後に指定したオプションが有効となります。

●アセンブラではアセンブルする対象デバイスの指定は省略できません。

なお，オプション指定の詳細については，“**4.4.3 (1) デバイス種別指定**”をお読みください。

指定方法：① アセンブラ起動時にコマンド・ライン上で，－Ｃオプションで指定する。

② $processor（$pc）制御命令で指定する。

●ヘルプ指定オプション（--）を指定すると，ほかに指定したすべてのオプションは無効となります。

保守／廃止

付録 C　オプション一覧

プログラムのオプションを表形式でまとめました。

プログラム開発の際にお役立てください。

このオプション一覧は索引としても使用できます。

保守／廃止

C.1　アセンブラ・オプション一覧

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
1	デバイス種別指定	-C デバイス種別	対象デバイスの種別を指定します。	独立	省略不可	45
2	オブジェクト・モジュール・ファイル出力指定	-O ファイル名	オブジェクト・モジュール・ファイルを出力します。	－O と －NO を同時に指定した場合は，あとで指定した方が有効です。	-O 入力ファイル名. REL	45
		－NO	オブジェクト・モジュール・ファイルを出力しません。			
3	オブジェクト・モジュール・ファイル強制出力指定	－J	オブジェクト・モジュール・ファイルを強制的に出力します。	－J と －NJ を同時に指定した場合は，あとで指定した方が有効です。	－NJ	45
		－NJ	－J オプションを無効にします。			
4	ディバグ情報出力指定	－G	ディバグ情報をオブジェクト・モジュール・ファイルへ出力します。	－G と －NG を同時に指定した場合は，あとで指定した方が有効です。	－G	45
		－NG	－G オプションを無効にします。			
5	インクルード・ファイル読み込みパス指定	-I[, パス名]…（複数指定可能）	インクルード・ファイルを指定したパスから読み込みます。	独立	環境変数 'INC78K1' により指定されたパス	45
6	アセンブル・リスト・ファイル出力指定	-P [ファイル名]	アセンブル・リスト・ファイルを出力します。	－P と －NP を同時に指定した場合は，あとで指定した方が有効です。	-P 入力ファイル名. PRN	45
		－NP	アセンブル・リスト・ファイルを出力しません。			
7	アセンブル・リスト・ファイル情報指定	－KA	アセンブル・リスト・ファイル中に，アセンブル・リストを出力します。	●－KS と －KX が同時に指定された場合は，－KS を無視します。 ●－KA と －NKA，－KS と －NKS，－KX と －NKX を同時に指定した場合は，あとで指定した方が有効です。 ●－NKA，－NKS，－NKX を同時に指定した場合は，－P を無視します。	－KA	45
		－NKA	－KA オプションを無効にします。			
		－KS	アセンブル・リスト・ファイル中に，シンボル・リストを出力します。		－NKS	45
		－NKS	－KS オプションを無効にします。			
		－KX	アセンブル・リスト・ファイル中に，クロス・レファレンス・リストを出力します。		－NKX	45
		－NKX	－KX オプションを無効にします。			

保守／廃止

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
8	アセンブル・リスト・ファイル形式指定	-LW[文字数]	アセンブル・リスト・ファイルの1行に印字する文字数を変更します。	−NP を指定した場合は，−LW が無視されます。	-LW132（コンソール出力の場合は80文字）	45
		−LL[行数]	アセンブル・リスト・ファイルの1頁に印字する行数を変更します。	−NP を指定した場合は，−LL が無視されます。	-LL66(コンソール出力の場合は改頁しません)	45
		-LH[文字列]	アセンブル・リスト・ファイルのヘッダに，指定された文字列を出力します。	−NP を指定した場合は，−LH が無視されます。	なし	45
		-LT[文字数]	タブの展開文字数を変更します。	−NP を指定した場合は，−LT が無視されます。	−LT8	45
		−LF	アセンブル・リスト・ファイルの最後に改頁コードを付加します。	●−LF と −NLF を同時に指定した場合は，あとで指定した方が有効です。 ●−NP を指定した場合は，−LF が無視されます。	−NLF	45
		−NLF	−LF オプションを無効にします。			
9	エラー・リスト・ファイル出力指定	-E [ファイル名]	エラー・リスト・ファイルを出力します。	−E と −NE を同時に指定した場合は，あとで指定した方が有効です。	−NE	45
		−NE	−E を無効にします。			
10	パラメータ・ファイル指定	-F [ファイル名]	入力ファイル名，オプションを指定したファイルから入力します。	独立	起動行上からのみオプション，入力ファイル名の入力が可能。	45
11	テンポラリ・ファイル作成パス指定	−T [パス]	テンポラリ・ファイルを，指定したパスに作成します。	独立	環境変数，'TMP'により指定されたパス。	46
12	ヘルプ指定	−−	ディスプレイ(コンソール)にヘルプ・メッセージを出力します。	他のオプションをすべて無効にします。	表示しません。	46

C

保守／廃止

C.2　リンカ・オプション一覧

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
1	ロード・モジュール・ファイル出力指定	-O[ファイル名]	ロード・モジュール・ファイルを出力します。	－O と －NO を同時に指定した場合は，あとで指定した方が有効です。	-O 入力ファイル名. LNK	102
		－NO	ロード・モジュール・ファイルを出力しません。			
2	ロード・モジュール・ファイル強制出力指定	－J	ロード・モジュール・ファイルを強制的に出力します。	－J と －NJ を同時に指定した場合は，あとで指定した方が有効です。	-NJ	102
		－NJ	－J オプションを無効にします。			
3	ディバグ情報出力指定	－G	ディバグ情報をロード・モジュール・ファイルへ出力します。	●－G と －NG を同時に指定した場合は，あとで指定した方が有効です。 ●－NG が指定された場合は，－KP，－KL の指定にかかわらず，ローカル・シンボル・リスト，パブリック・シンボル・リストは出力されません。	－G	102
		－NG	－G オプションを無効にします。			
4	スタック解決用シンボル生成指定	－S[領域名]	スタック解決用のパブリック・シンボルを自動生成します。	－S と －NS を同時に指定した場合は，あとで指定した方が有効です。	－NS	102
		－NS	－S オプションを無効にします。			
5	ディレクティブ・ファイル指定	-D[ファイル名]	特定のファイルをディレクティブ・ファイルとして指定します。	独立	なし	102
6	リンク・リスト・ファイル出力指定	-P[ファイル名]	リンク・リスト・ファイルの出力を指定します。	－P と －NP を同時に指定した場合は，あとで指定した方が有効です。	-P 入力ファイル名. MAP	102
		－NP	－P オプションを無効にします。			

保守／廃止

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
7	リンク・リスト・ファイル情報指定	－KM	リンク・リスト・ファイル中に，マップ・リストを出力します。	●－KMと－NKMを同時に指定した場合は，あとで指定した方が有効です。 ●－NKM，－NKP，－NKLをすべて指定した場合は，－Pは無効となります。 ●－NKM を指定した場合，－KD は無効となります。 ●－KDと－NKD，－KPと－NKP，－KLと－NKLを同時に指定した場合は，あとで指定した方が有効です。 ●－NG を指定した場合は，－KP，－KLの指定にかかわらず，ローカル・シンボル・リスト，パブリック・シンボル・リストは出力されません。	－KM	102
		－NKM	－KMオプションを無効にします。			
		－KD	リンク・リスト・ファイル中に，リンク・ディレクティブ・ファイルを出力します。		－KD	102
		－NKD	－KDオプションを無効にします。			
		－KP	アセンブル・リスト・ファイル中に，クロスレファレンス・リストを出力します。		－NKP	102
		－NKP	－KPオプションを無効にします。			
		－KL	リンク・リスト・ファイル中にローカル・シンボル・リストを出力します。		－NKL	102
		－NKL	－KLオプションを無効にします。			
8	リンク・リスト・ファイル形式指定	－LL[行数]	リストの１頁に印字する行数を変更します。	－NP を指定した場合は，－LL は無視されます。	－LL66 (ディスプレイ(コンソール)出力の場合は改頁しません。)	102
		－LF	リスト・ファイルの最後に，改頁コードを付加します。	●－LF と －NLF を同時に指定した場合は，あとで指定した方が有効です。 ●－NP を指定した場合は，あとで指定した方が有効です。	－NLF	102
		－NLF	－LFオプションを無効にします。			
9	エラー・リスト・ファイル出力指定	-E [ファイル名]	エラー・リスト・ファイルを出力します。	－E と －NE を同時に指定した場合は，あとで指定した方が有効です。	－NE	102
		－NE	－Eオプションを無効にします。			
10	ライブラリ・ファイル指定	-B [ファイル名]	特定のファイルを，ライブラリ・ファイルとして入力します。	独立	なし	102
11	ライブラリ・ファイル読み込みパス指定	-I パス名 [,パス名]… (複数指定可能)	ライブラリ・ファイルを指定したパスから読み込みます。	－Bオプションで，パス名を含まないライブラリ・ファイルを指定した場合は無効となります。	環境変数 'LIB78K1' により指定されたパス。	102

C

保守／廃止

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
12	パラメータ・ファイル指定	-F [ファイル名]	入力ファイル名，オプションを指定したファイルより入力します。	独立	起動行上からのみオプション，入力ファイル名の入力が可能。	102
13	テンポラリ・ファイル作成	-T [パス名]	テンポラリ・ファイルを，指定したパスに作成します。	独立	環境変数，'TMP'により指定されたパス。	102
14	ヘルプ指定	ー ー	ディスプレイ（コンソール）にヘルプ・メッセージを出力します。	他のオプションをすべて無効にします。	表示しません。	102

保守／廃止

C.3　オブジェクト・コンバータ・オプション一覧

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
1	HEX 形式オブジェクト・モジュール・ファイル出力指定	-O[ファイル名]	HEX 形式オブジェクト・モジュール・ファイルを出力します。	－O と －NO を同時に指定した場合は，あとで指定した方が有効です。	-O 入力ファイル名. HEX (拡張空間に対するファイル・タイプ。H1～H15)	147
		－NO	HEX 形式オブジェクト・モジュール・ファイルを出力しません。			
2	シンボル・テーブル・ファイル出力指定	-S [ファイル名]	シンボル・テーブル・ファイルを出力します。	－S と －NS を同時に指定した場合は，あとで指定した方が有効です。	-S 入力ファイル名. SYM (拡張空間に対するファイル・タイプ。H1～H15)	147
		－NS	シンボル・テーブル・ファイルを出力しません。			
3	オブジェクト・アドレス順ソート指定	－R	HEX 形式オブジェクトをアドレス順にソートします。	●－R と －NR を同時に指定した場合は，あとで指定した方が有効です。 ●－NO を指定した場合は，－R は無視されます。	－NR	147
		－NR	－R オプションを無効にします。			
4	オブジェクト充塡指定	－U[充塡値] [[,ｽﾀｰﾄ[,ｻｲｽﾞ]]	HEX 形式オブジェクトが出力されないアドレス領域に対して，指定した充塡値をオブジェクト・コードとして出力します。	－NO を指定した場合は，－U は無視されます。	なし	147
5	エラー・リスト・ファイル	-E [ファイル名]	エラー・リスト・ファイルを出力します。	－E と －NE を同時に指定した場合は，あとで指定した方が有効です。	－NE	147
		－NE	－E オプションを無効にします。			
6	パラメータ・ファイル指定	-F [ファイル名]	入力ファイル，オプションを，指定したファイルから入力します。	独立	起動行上からのみオプション，入力ファイル名を入力できます。	147
7	ヘルプ指定	－－	ディスプレイ（コンソール）にヘルプ・メッセージを出力します。	他のすべてのオプションを無効にします。	表示しません。	147

C

保守／廃止

C.4　ライブラリアン・オプション一覧

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
1	リスト・ファイル形式指定	-LW[文字数]	リスト・ファイルの1行に印字する文字数を変更します。	●LIST（サブコマンド）を指定しない場合は無効です。 ●－LFと－NLFを同時に指定した場合は，あとで指定した方が有効です。	-LW132（コンソール出力の場合は80文字）	170
		－LL[行数]	リスト・ファイルの1頁の行数を変更します。		-LL66（コンソール出力の場合は改頁しません）	170
		－LF	リスト・ファイルの最後に，改頁コードを付加します。		－NLF	170
		－NLF	－LFオプションを無効にします。			
2	テンポラリ・ファイル作成	－T[パス名]	テンポラリ・ファイルを指定したパスに作成します。	独立	環境変数'TMP'により指定されたパス。	170
3	ヘルプ指定	－－	ディスプレイ（コンソール）にヘルプ・メッセージを出力します。	他のすべてのオプションを無効とします。	表示しません。	170

C.5　リスト・コンバータ・オプション一覧

項番	分　類	記述形式	機　能	他のオプションとの関係	省略時解釈	参照頁
1	オブジェクト・モジュール・ファイル入力指定	-R［ファイル名］	オブジェクト・モジュール・ファイルを入力します。	独立	-R アセンブル・リスト・ファイル名．REL	201
2	ロード・モジュール・ファイル入力指定	-L［ファイル名］	ロード・モジュール・ファイルを入力します。	独立	-L アセンブル・リスト・ファイル名．LNK	201
3	アブソリュート・アセンブル・リスト・ファイル出力指定	-O［ファイル名］	アブソリュート・アセンブル・リスト・ファイルを出力します。	独立	-O アセンブル・リスト・ファイル名．P	201
4	エラー・リスト・ファイル出力指定	-E［ファイル名］	エラー・リスト・ファイルを出力します。	－Eと－NEを同時に指定した場合は，あとで指定した方が有効です。	－NE	201
		－NE	－Eオプションを無効にします。			
5	パラメータ・ファイル指定	-F［ファイル名］	入力ファイル，オプションを指定したファイルより入力します。	独立	起動行上からのみオプション，ファイル名の入力が可能。	201
6	ヘルプ指定	－－	ディスプレイ（コンソール）にヘルプ・メッセージを出力します。	他のすべてのオプションを無効にします。	表示しません。	201

保守／廃止

付録 D　サブコマンド一覧

サブコマンドを，一覧表にまとめて示します。

プログラム開発の際にお役立てください。

このサブコマンド一覧は，索引としても使用できます。

保守／廃止

項番	分　類	記　述　形　式	機　能	短縮形	参照頁
1	CREATE	CREATE △ライブラリ・ファイル名[△トランザクション]	ライブラリ・ファイルを新規に作成します。	C	177
2	ADD	ADD △ライブラリ・ファイル名△トランザクション	ライブラリ・ファイルにモジュールを追加します。	A	177
3	DELETE	DELETE △ライブラリ・ファイル名▲(▲モジュール名[▲,…]▲)	ライブラリ・ファイル内のモジュールを削除します。	D	177
4	REPLACE	REPLACE △ライブラリ・ファイル名△トランザクション	ライブラリ・ファイル内のモジュールを他のモジュールと置き換えます。	R	177
5	PICK	PICK △ライブラリ・ファイル名▲(▲モジュール名[▲,…]▲)	ライブラリ・ファイル内のモジュールを取り出します。	P	177
6	LIST	LIST [△オプション]△ライブラリ・ファイル名[▲(▲モジュール名[▲,…]▲)]	ライブラリ・ファイル内のモジュール情報を出力します。	L	177
7	HELP	HELP	ディスプレイ(コンソール)にヘルプ・メッセージを出力します。	H	177
8	EXIT	EXIT	ライブラリアンを終了します。	E	177

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］ RA78K/I　アセンブラ・パッケージ　ユーザーズ・マニュアル　操作編

（EEU-883（第1版），September 1992　P）

［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，NEC 半応技本部員，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。

NEC 半導体応用技術本部インフォメーションセンター
FAX：(044) 548-7900

キリトリ

保守／廃止

保守／廃止

NEC 日本電気株式会社

本　　社	〒108-01 東京都港区芝五丁目7番1号(日本電気本社ビル)	
コンシューマ, OA, インダストリ 半導体販売事業部	〒108-01 東京都港区芝五丁目7番1号(日本電気本社ビル)	東 京(03)3454−1111
中部支社 半導体販売部	〒460 名古屋市中区栄四丁目14番5号(松下中日ビル)	名古屋(052)242−2755
関西支社 半導体販売部	〒540 大阪市中央区城見一丁目4番24号(日本電気関西ビル)	大　阪(06)945−3178 大　阪(06)945−3200 大　阪(06)945−3208

北海道支社	札　幌	(011)231−0161
東北支社	仙　台	(022)261−5511
岩手支店	盛　岡	(0196)51−4344
山形支店	山　形	(0236)23−5511
郡山支店	郡　山	(0249)23−5511
いわき支店	いわき	(0246)21−5511
長岡支店	長　岡	(0258)36−2155
水戸支店	水　戸	(0292)26−1717
神奈川支店	横　浜	(045)324−5511
群馬支店	高　崎	(0273)26−1255
太田支店	太　田	(0276)46−4011
宇都宮支店	宇都宮	(0286)21−2281
小山支店	小　山	(0285)24−5011
長野支店	長　野	(0262)35−1444
松本支店	松　本	(0263)35−1666
諏訪支店	諏　訪	(0266)53−5350
甲府支店	甲　府	(0552)24−4141
埼玉支社	大　宮	(048)641−1411

立川支社	立　川	(0425)26−0911
千葉支社	千　葉	(043)227−5441
静岡支店	静　岡	(054)255−2211
沼津支店	沼　津	(0559)63−4455
浜松支店	浜　松	(053)452−2711
北陸支社	金　沢	(0762)23−1621
福井支店	福　井	(0776)22−1866
富山支店	富　山	(0764)31−8461
京都支社	京　都	(075)352−2261
神戸支店	神　戸	(078)332−3311
中国支社	広　島	(082)242−5504
鳥取支店	鳥　取	(0857)27−5311
岡山支店	岡　山	(0862)25−4455
徳島支社	徳　島	(0886)36−1200
新居浜支店	新居浜	(0897)32−5001
松山支店	松　山	(0899)45−4111
九州支社	福　岡	(092)271−7700
北九州支店	北九州	(093)541−2887

(技術お問い合せ先)

半導体応用技術本部 マイクロコンピュータ技術部	〒210	川崎市川崎区駅前本町15番5号(十五番館)	川　崎 (044)246−3922
半導体応用技術本部 中部応用システム技術部	〒460	名古屋市中区栄四丁目14番5号(松下中日ビル)	名古屋 (052)242−2762
半導体応用技術本部 西日本応用システム技術部	〒540	大阪市中央区城見一丁目4番24号(日本電気関西ビル)	大　阪 (06)945−3383

半導体応用技術本部
インフォメーションセンター
FAX(044)548−7900

A92.7