

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のですが、誤りが無いことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。  
標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット  
高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）  
特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関し、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社とその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

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

# SuperH RISC engine C/C++ コンパイラパッケージ

アプリケーションノート：<コンパイラ活用ガイド>

オブジェクト活用ガイド

本ドキュメントでは、オブジェクトの活用方法を紹介します。

## 目次

1.	ライブラリ内の特定シンボルのセクション変更 .....	2
1.1	概要 .....	2
1.2	ライブラリアンインタフェース .....	3
1.3	最適化リンケージエディタ .....	4
2.	空き領域をダミーデータで埋める方法 .....	6
2.1	概要 .....	6
2.2	手順 .....	7
3.	ROMに固定化されているシンボルの呼び出し .....	9
3.1	概要 .....	9
3.2	手順 .....	10
4.	ライブラリファイルとリロケータブルファイル .....	11
4.1	ライブラリファイルとリロケータブルファイルの違い .....	11
4.2	使用する関数のみリンクする方法 .....	12
4.3	既存ライブラリファイルのオブジェクトモジュールを全てリンク対象にする方法 .....	12
5.	物理アドレスのロードモジュールを作成する方法 .....	15
5.1	概要 .....	15
5.2	作成方法 .....	16
	サポート窓口<website and support,ws> .....	17

### 1. ライブラリ内の特定シンボルのセクション変更

#### 1.1 概要

本章では、ライブラリ内の特定シンボルのセクションを変更する方法を説明します。ここでは、標準ライブラリ内の `_INITSCT()` 関数のセクションを変更する場合を例に説明します。

プログラムを ROM から RAM にコピーして実行したい場合、標準ライブラリの `_INITSCT()` 関数を用いて、ROM から RAM へのコピーを行います。標準ライブラリ関数もユーザ関数と同様に P セクションに配置されるため、単純に P セクションを RAM へコピーしようとしても、`_INITSCT()` 関数自体がコピー対象に含まれているため、正しく処理が行えません。この場合、`_INITSCT()` 関数のセクションを P セクションから他のセクションに変更する必要があります。

ライブラリ内の特定シンボルのセクションを変更するには、ライブラリアンインタフェースを使用する方法と最適化リンケージエディタを使用する方法の 2 通りの方法があります。

[`_INITSCT()` 関数のセクションを変更しない場合]    [`_INITSCT()` 関数のセクションを変更する場合]

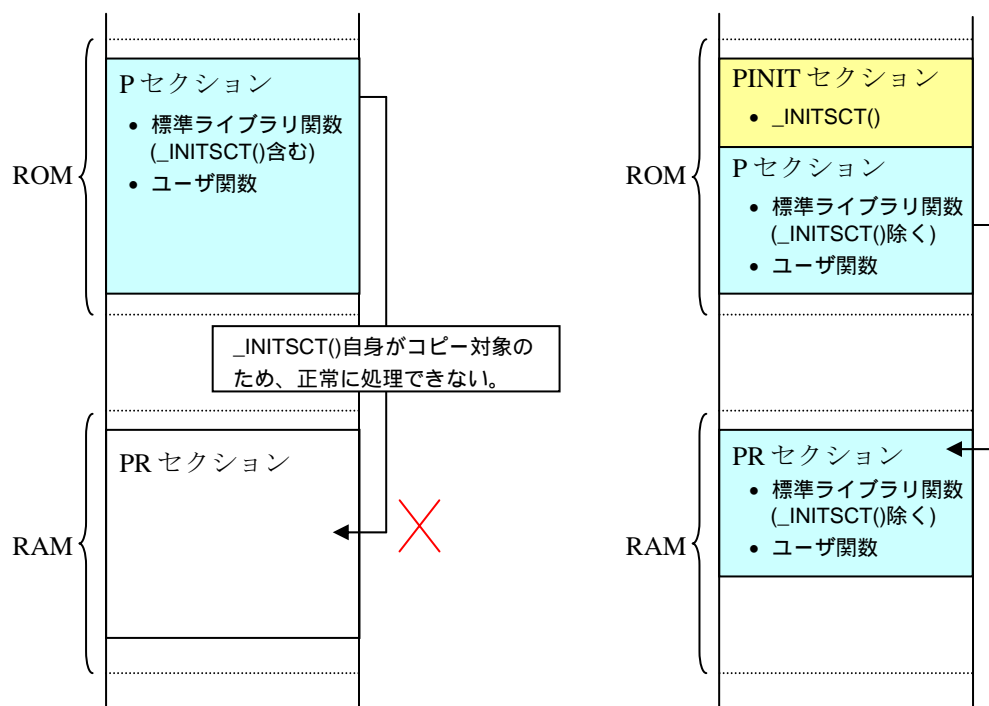


図 1-1

## 1.2 ライブラリアンインタフェース

ライブラリアンインタフェースは、ライブラリ内特定モジュールのセクションを変更することができます。ライブラリアンインタフェースを使用して、ライブラリの特定シンボルのセクションを変更する方法について説明します。High-performance Embedded Workshop (以下、ルネサス統合開発環境と呼ぶ)の[ツール]メニューより[Renesas H Series Librarian Interface]を選択して、ライブラリアンインタフェースを起動してください。そして、このライブラリアンインタフェースの[File]メニューの[Open]より、対象のライブラリを選択してください。ライブラリに含まれるオブジェクトモジュール一覧が表示されます。オブジェクトモジュール一覧から変更したいシンボルが含まれるセクションを選択して、[Action]メニューの[Rename Section]を選択してください。

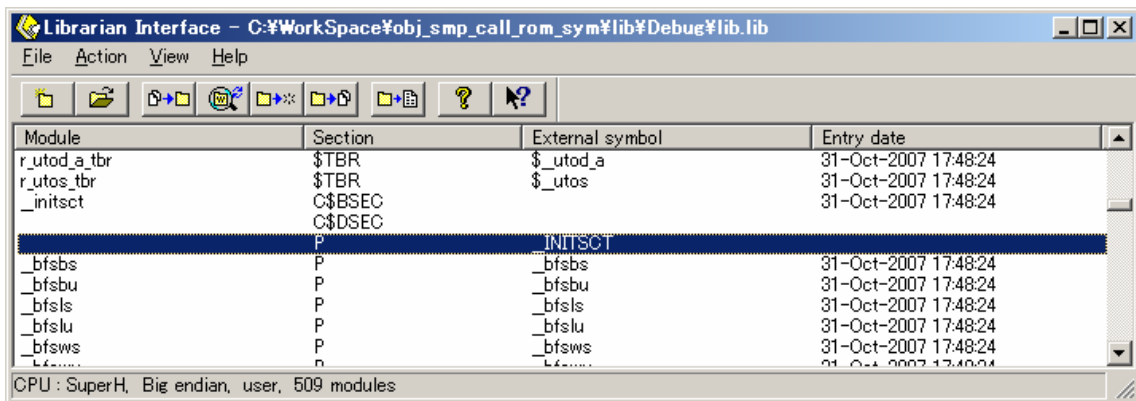


図 1-2

[Rename Section]ダイアログボックスが表示されます。このダイアログボックスで、[After]ボタンをクリックし、[After]ダイアログボックスを表示してください。この[After]ダイアログボックスの[Section Name After Renaming]に変更後のセクションを入力し、[OK]ボタンをクリックしてください。

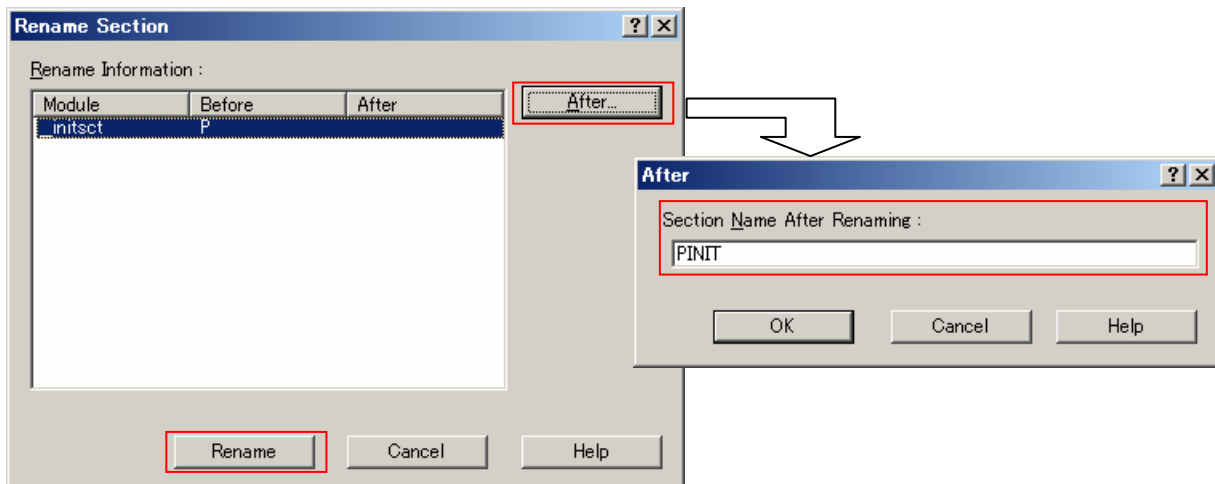


図 1-3

[Rename Section]ダイアログボックスで[Rename]ボタンをクリックすると、選択したセクションが更新されます。

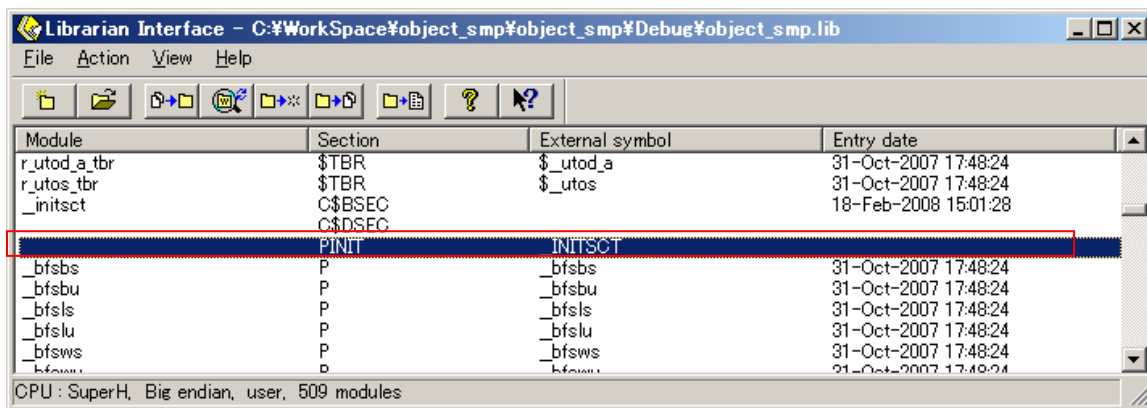


図 1-4

### 1.3 最適化リンケージエディタ

最適化リンケージエディタを使用して、ライブラリの特定シンボルのセクションを変更する方法について説明します。まず初めに、セクションとシンボル情報を含む、ライブラリのオブジェクトモジュール情報(ライブラリリスト)を出力してください。セクションとシンボル情報を含むライブラリリストは、list オプションと show=symbol,section オプションを指定すると出力できます。例えば、標準ライブラリ stdlib.lib のライブラリリストを出力するには、以下のように最適化リンケージエディタに指定してください。

```
optlnk -list -show=symbol,section -library=stdlib.lib -form=library -output=tmp_stdlib.lib
```

ライブラリリストには、以下の情報が出力されます。

```
...
*** Library List ***

MODULE      LAST UPDATE
SECTION
SYMBOL

MODULE:      オブジェクトモジュール名
LAST UPDATE: オブジェクトモジュールを登録した日付を表示します。オブジェクトモジュールが更新された場合は、最新の更新日付を表示します。
SECTION      オブジェクトモジュール内セクション名を表示します。
SYMBOL       セクション内のシンボル名を表示します。
```

出力されるライブラリリスト tmp\_stdlib.lbp より、\_INITSCT()関数はオブジェクトモジュール \_\_initsct の P セクションにあることがわかります。

```
...
*** Library List ***

MODULE      LAST UPDATE
SECTION
SYMBOL

div
          30-Oct-2006 16:20:00
P
  _div
  _ldiv
...
__initsct
          30-Oct-2006 16:20:00
C$BSEC
C$DSEC
P
  __INITSCT
...
```

次に、rename オプションを指定しライブラリ内特定モジュールのセクションを変更します。rename オプションの書式は以下の通りです。

rename=<サブオプション>[,...]

<サブオプション>: [<オブジェクトモジュール>](<変更前セクション>=<変更後セクション>)

\_INITSCT()関数のあるオブジェクトモジュール \_\_initsct のセクションを P から PINIT に変更する場合、以下のように指定してください。セクションが置き換わったライブラリ new\_stdlib.lib が生成されます。

optlnk -rename=\_\_initsct(P=PINIT) -library=stdlib.lib -form=library -output=new\_stdlib.lib

生成された new\_stdlib.lib のライブラリリストは以下のようになります。

```

*** Library List ***
...
MODULE      LAST UPDATE
SECTION
SYMBOL

div
          30-Oct-2006 16:20:00
  P
  _div
  _ldiv
          ...
__initsct
20-Feb-2008 15:35:14
C$BSEC
C$DSEC
PINIT
__INITSCT
          ...

```

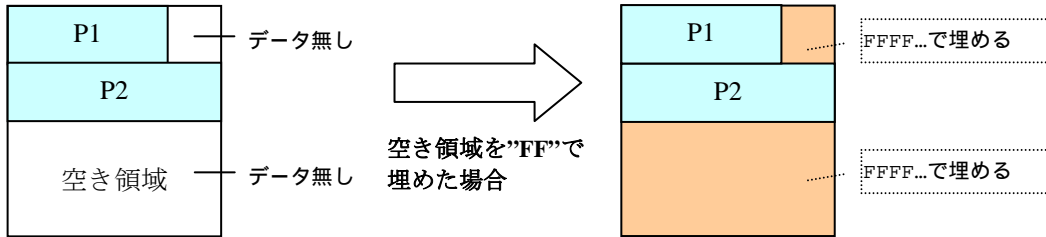
## 2. 空き領域をダミーデータで埋める方法

### 2.1 概要

デフォルトの設定で ROM イメージファイル(モトローラ S 形式ファイル、インテル HEX 形式ファイル、バイナリファイル)を生成した場合、ROM イメージファイルにはプログラム及び ROM データのみが存在します。そのため、メモリに ROM イメージファイルを書き込んだとき、空き領域ができることがあります。空き領域の無い ROM イメージファイルを作成したい場合は、最適化リンカージェディタの space オプションを使用してください。

[空き領域のある ROM イメージファイル]

[空き領域を埋めた ROM イメージファイル]



モトローラ S 形式ファイル

モトローラ S 形式ファイル

```
S00E000073706163653120206D6F7424
S1092000000900090009BB
S10B20100009000900090009A0
S9030800F4
```

```
S00E000073706163653120206D6F7424
S1092000000900090009BB
S1052006FFFFFFD6
S10B20080009000900090009A8
S1132010FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFCC
S9030800F4
```

下線部がデータ

下線部がデータ

図 2-1

### 2.2 手順

最適化リンカージエータの space オプションを使用し、空き領域のない ROM イメージファイルを作成する方法を説明します。空き領域のない ROM イメージファイルを作成するには、まず、ROM イメージの出力範囲を指定する必要があります。出力範囲を指定するには、ルネサス統合開発環境の[SuperH RISC engine Standard Toolchain]ダイアログボックスの[最適化リンカ]タブで以下の設定をしてください。

[カテゴリ]: “出力”

[出力形式]: “HEX(ELF/DWARF アブソリュート付き)”、“S タイプ(ELF/DWARF アブソリュート付き)”  
もしくは、“バイナリ(ELF/DWARF アブソリュート付き)”

[オプション項目]: “出力ファイルの分割”

[出力ファイルの分割]: チェックボックスをオン

[追加]: 表示される[Add output file]ダイアログボックスで、出力範囲と出力ファイル名を設定

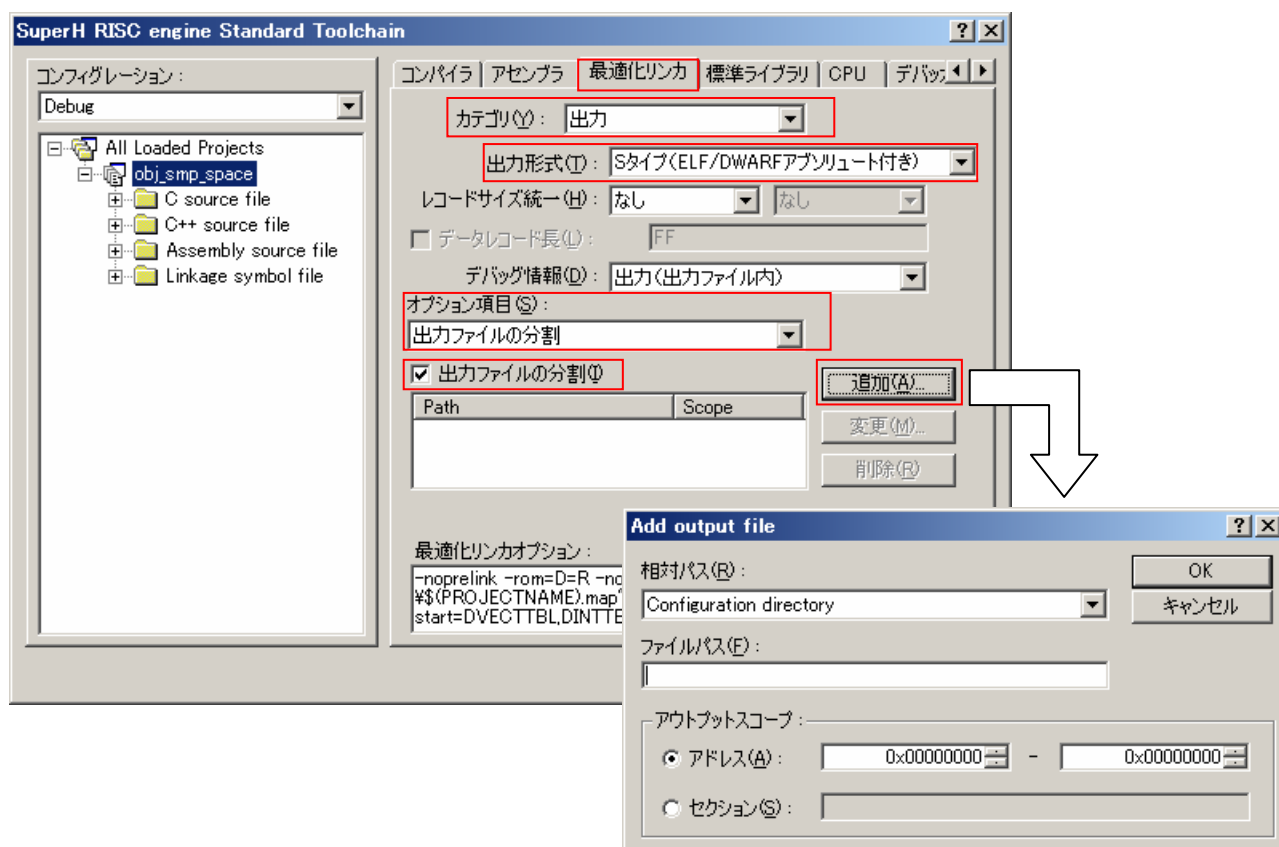


図 2-2

次に、space オプションを使用して空き領域をダミーデータで埋めるために、以下の設定をしてください。

[オプション項目]: “空きエリア出力指定”

[空きエリア出力]: “指定値”もしくは“乱数”

“指定値”を指定した場合は値を設定してください

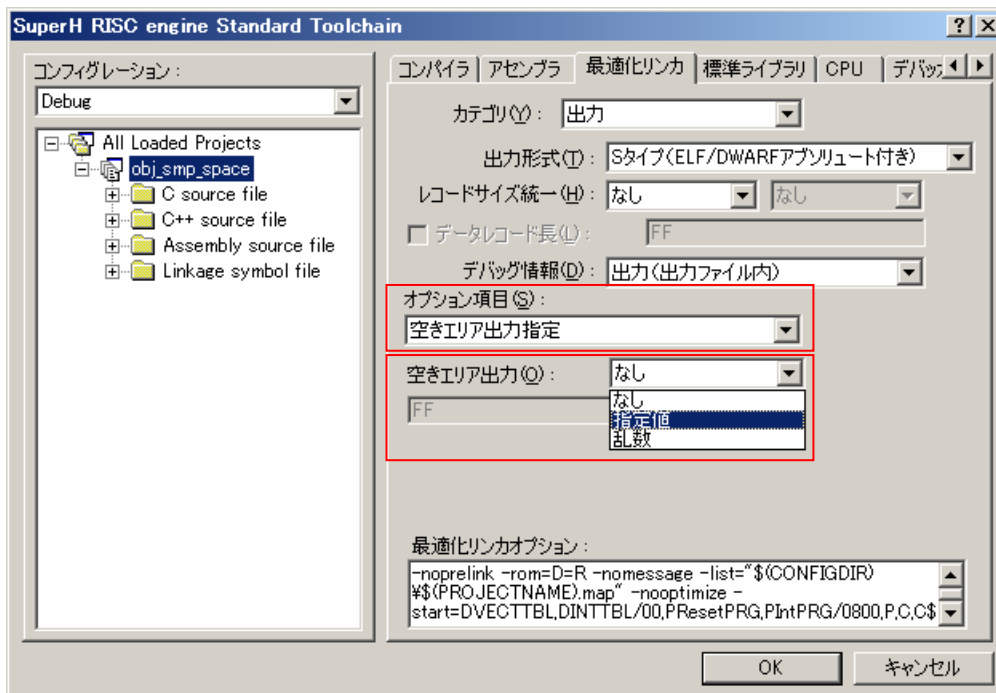


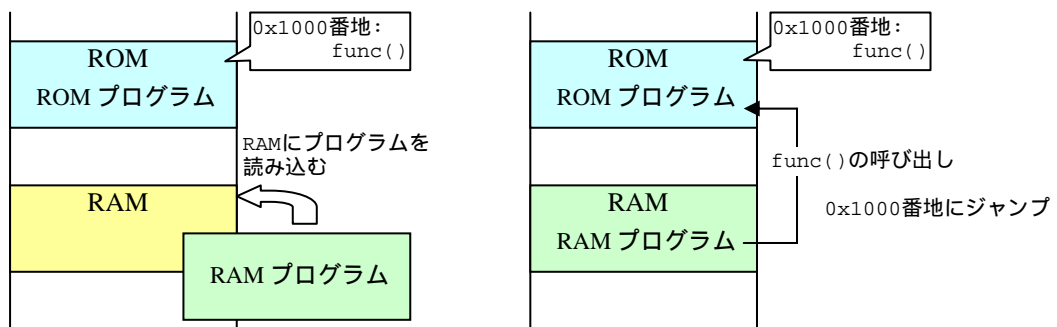
図 2-3

### 3. ROMに固定化されているシンボルの呼び出し

#### 3.1 概要

本章では、新規に作成するロードモジュールから、すでにROMに固定化されているシンボルを呼び出す方法を説明します。ここでは、新規に作成するロードモジュールをRAMに読み込み、そのRAMに読み込んだロードモジュールからROMに固定化されているロードモジュールの関数を呼び出す場合を例に説明します。

ROMに固定化されている関数を呼び出すには、ROMプログラムの関数アドレスが分かっている必要があります。ROMプログラムのロードモジュール生成時に、シンボルのアドレス情報が記載されたシンボルアドレスファイルをあらかじめ出力しておきます。このシンボルアドレスファイルはアセンブラ制御命令で記載されているため、アセンブリソースファイルとして使用できます。RAMプログラムのロードモジュールを生成するときに、このシンボルアドレスファイルを実アセンブル、リンクすることで、RAMプログラムからROMプログラムの関数呼び出しが可能になります。



RAMプログラムのロードモジュール生成時に、func()が0x1000番地にあることを結び付ける必要があります。

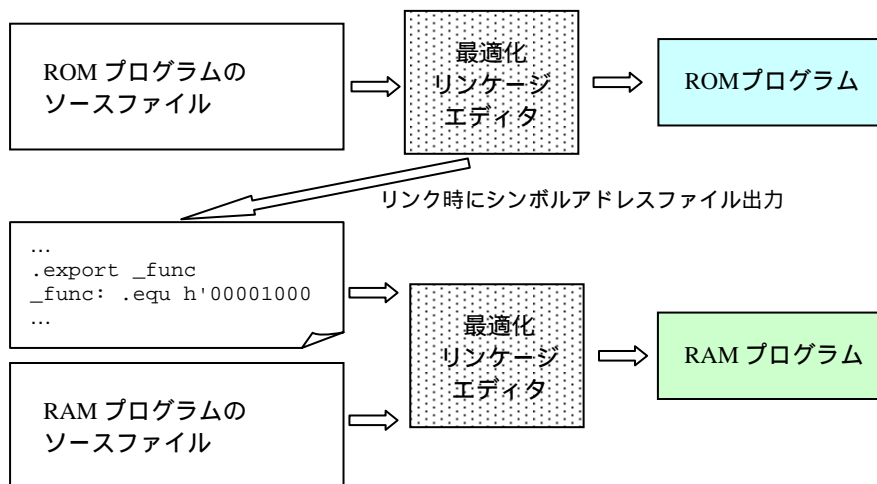


図 3-1

### 3.2 手順

先述した例の RAM プログラムのロードモジュールを生成するまでの手順を説明します。まず、ROM プログラムのロードモジュールを生成するときに、シンボルアドレスファイルを出力します。シンボルアドレスファイルを生成するには、ルネサス統合開発環境の[SuperH RISC engine Standard Toolchain]ダイアログボックスの[最適化リンク]タブで以下の設定をしてください。ロードモジュール生成時、指定したセクションのシンボルアドレスファイル(拡張子“fsy”)が出力されます。

[カテゴリ]: “セクション”

[設定項目]: “シンボルアドレスファイル”

[追加]: シンボルアドレスファイルを生成するセクションを指定

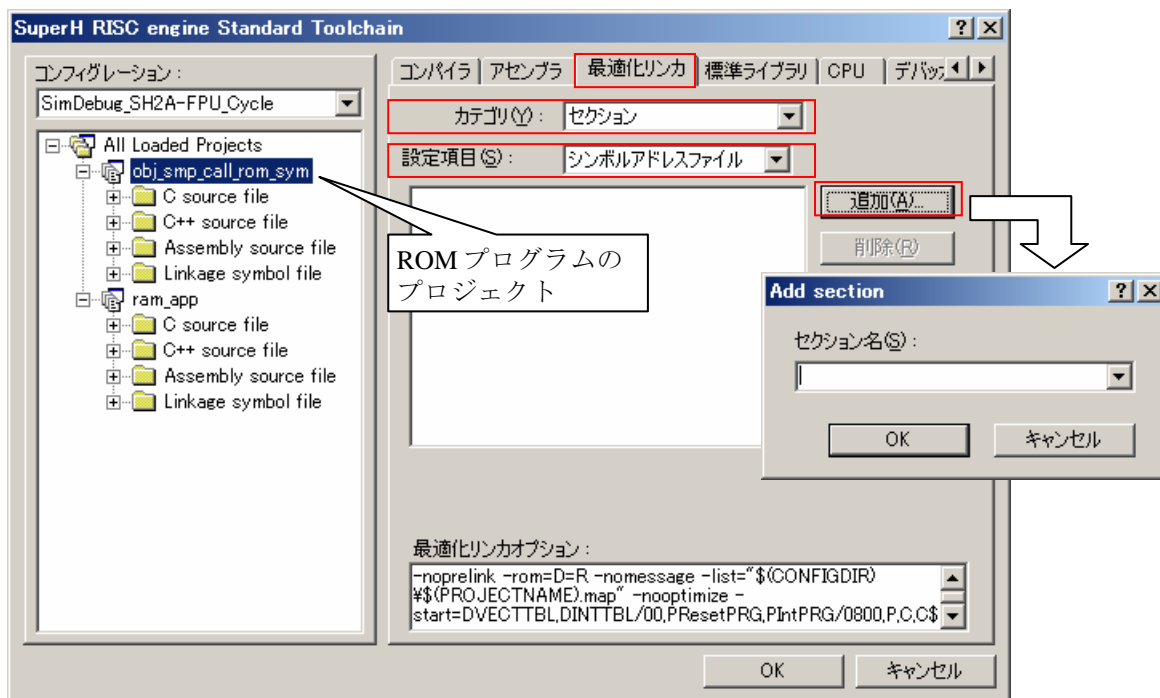


図 3-2

次に、生成されたシンボルアドレスファイルを、RAM プログラム作成時にプロジェクトに取り込み、ロードモジュールを生成してください。

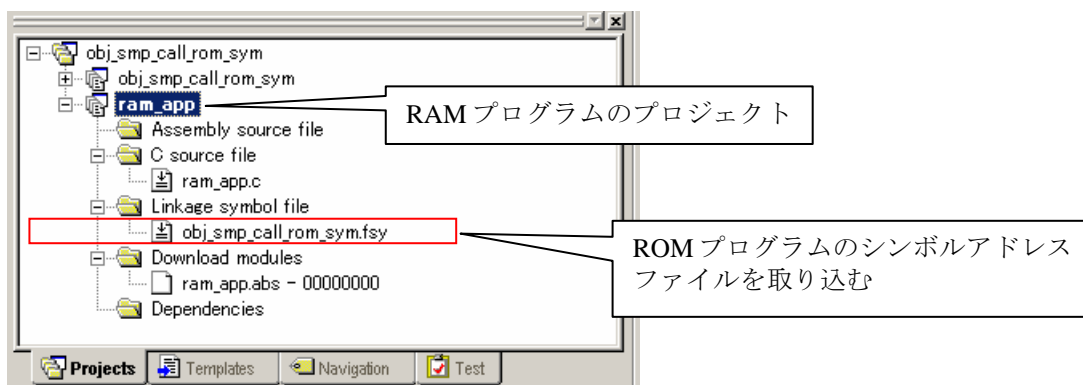


図 3-3

## 4. ライブラリファイルとリロケータブルファイル

### 4.1 ライブラリファイルとリロケータブルファイルの違い

ライブラリファイルとリロケータブルファイルの違いについて説明します。ライブラリファイルは参照のあるシンボルを含むオブジェクトモジュールのみがリンクされます。これに対し、リロケータブルファイルは全てのオブジェクトモジュールがリンクされます。

#### (1) ライブラリファイル

参照されるシンボルを含むオブジェクトモジュールのみリンクされます。直接参照されるシンボルだけではなく、間接参照されるシンボルを含むオブジェクトモジュールもリンクされます。図 4-1 に示した例では、オブジェクト 1 からライブラリのオブジェクトモジュール module1.obj の func1\_A() が参照されるため、module1.obj がリンクされます。さらに、module1.obj からライブラリのオブジェクトモジュール module3.obj の func3\_A() が参照されるため、module3.obj もリンクされます。

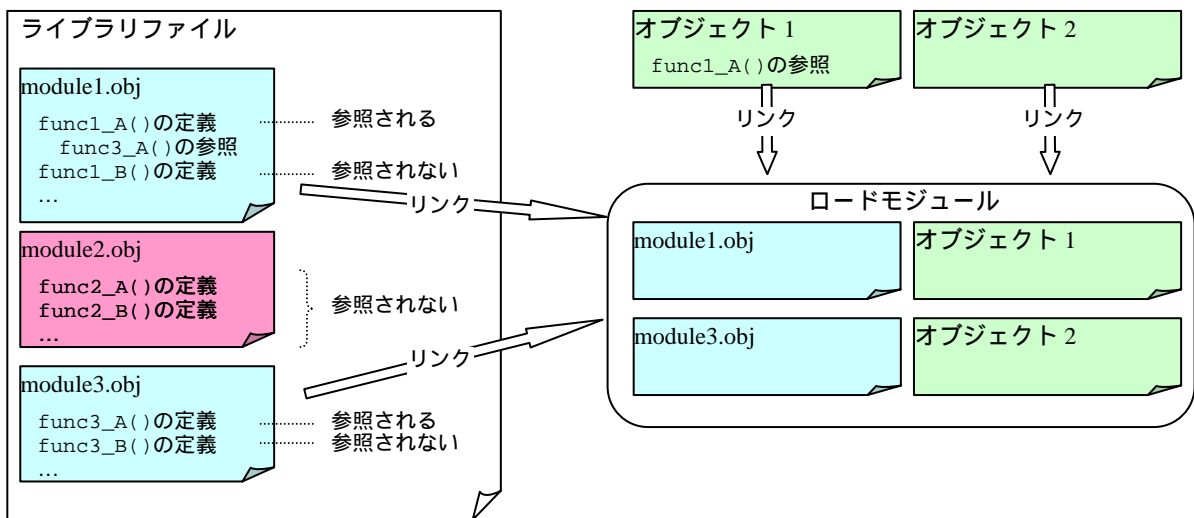


図 4-1

#### (2) リロケータブルファイル

参照の有無に関わらず全てのオブジェクトモジュールがリンクされます。図 4-2 に示した例では、リロケータブルファイルのオブジェクトモジュール module2.obj のシンボルはどこからも参照されませんが、module2.obj はリンクされます。

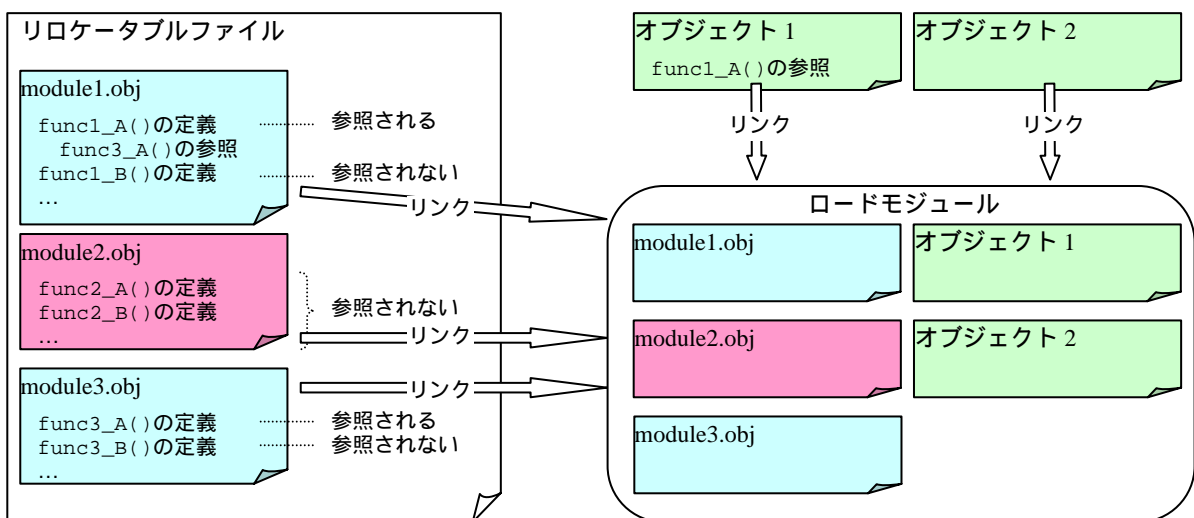


図 4-2

### 4.2 使用する関数のみリンクする方法

使用する関数のみリンクするには、1つのソースファイルに1つの関数定義を記述し、ライブラリを作成してください。ライブラリにすることで、使用する関数のみリンクされます。下記図に示す例では、オブジェクト1から、直接参照される func1\_A()を含むモジュール module1A.objと、間接参照される func3\_A()を含むモジュール module3A.objのみリンクされます。

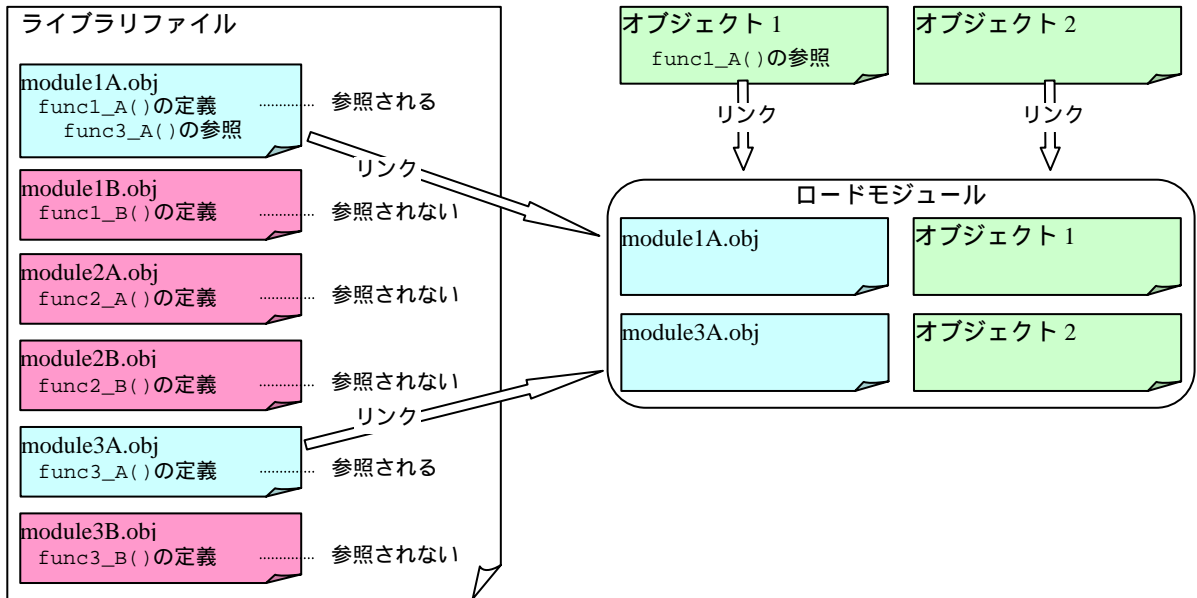


図 4-3

### 4.3 既存ライブラリファイルのオブジェクトモジュールを全てリンク対象にする方法

ライブラリファイル形式でアプリケーションが提供されている場合、先に説明したとおり、実際に呼び出している関数のみがリンク対象となります。しかし、将来の拡張性を考慮して、ライブラリファイル内の全ての関数をあらかじめリンク対象にしたい場合があります。このようなときには、ライブラリファイルをリロケータブルファイルに変換してリンクしてください。

ライブラリファイルをリロケータブルファイルに変換する方法は複数ありますが、本章では、ライブラリファイルからオブジェクトファイルを抽出し、抽出した複数のオブジェクトファイルから1つのリロケータブルファイルを作成する方法を紹介します。

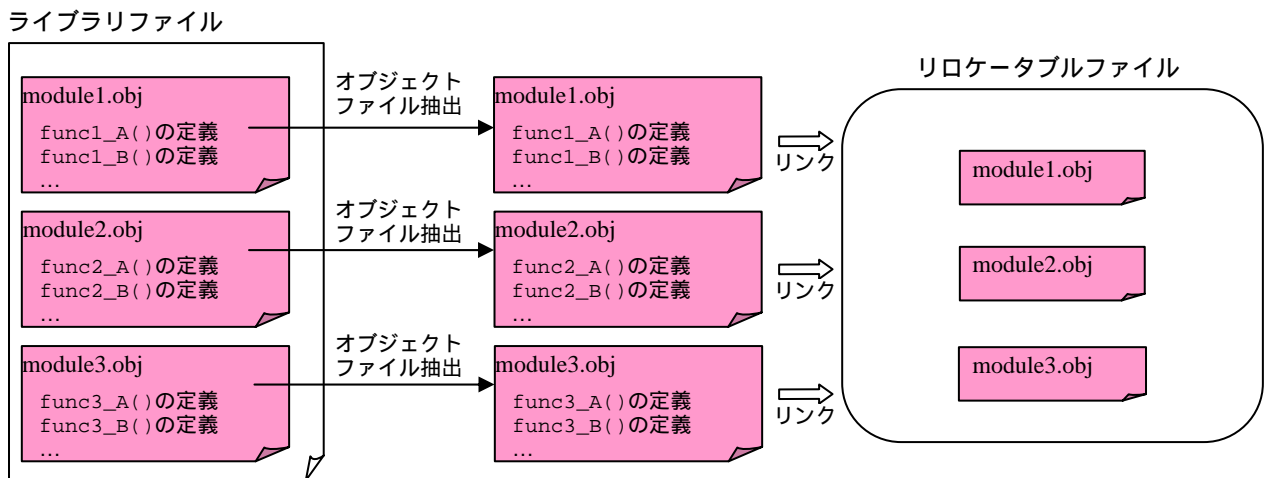


図 4-4

ライブラリファイルからオブジェクトファイルを抽出するには、ライブラリアンインタフェースを使用する方法と最適化リンケージエディタを使用する方法の2通りの方法があります。

### (1) ライブラリアンインタフェースを使用する方法

ライブラリアンインタフェースを使用してオブジェクトファイルを抽出する方法を説明します。ライブラリアンインタフェース(図 1-2)のオブジェクトモジュール一覧から抽出したいオブジェクトモジュールを選択し、ライブラリアンインタフェースの[Action]メニューから[Extract]を選択してください。[Extract]ダイアログボックスが表示されます。[Extract]ダイアログボックスで以下のように指定し、[OK]ボタンを押すと、選択したオブジェクトモジュールがオブジェクトファイルに抽出されます。

[Output file type]: "Object file"

[Output folder]: オブジェクトファイルを出力するフォルダを指定

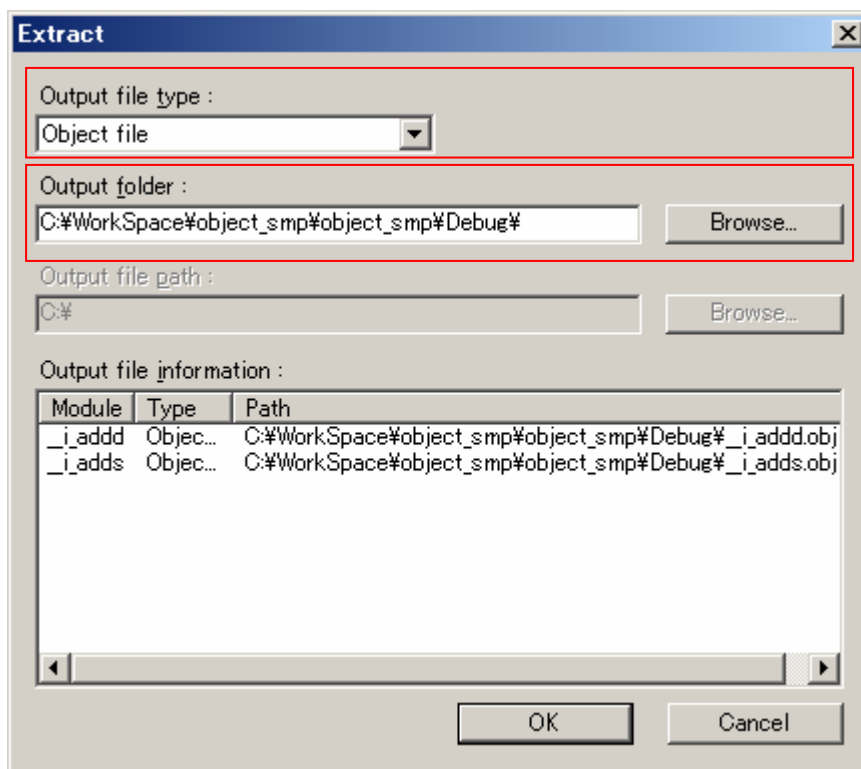


図 4-5

ルネサス統合開発環境で、抽出したオブジェクトファイルをプロジェクトに取り込んでください。

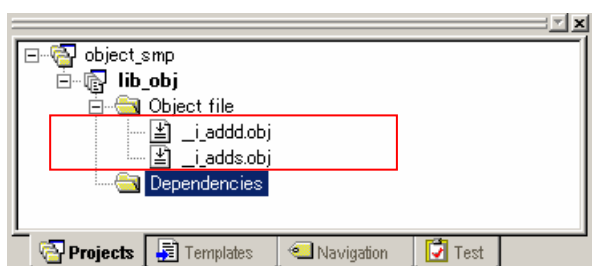


図 4-6

[SuperH RISC engine Standard Toolchain]ダイアログボックスの[最適化リンカ]タブで以下の設定をして、プロジェクトに取り込んだオブジェクトファイルからリロケータブルファイルを生成してください。

[カテゴリ]: “出力”

[出力形式]: “リロケータブル”

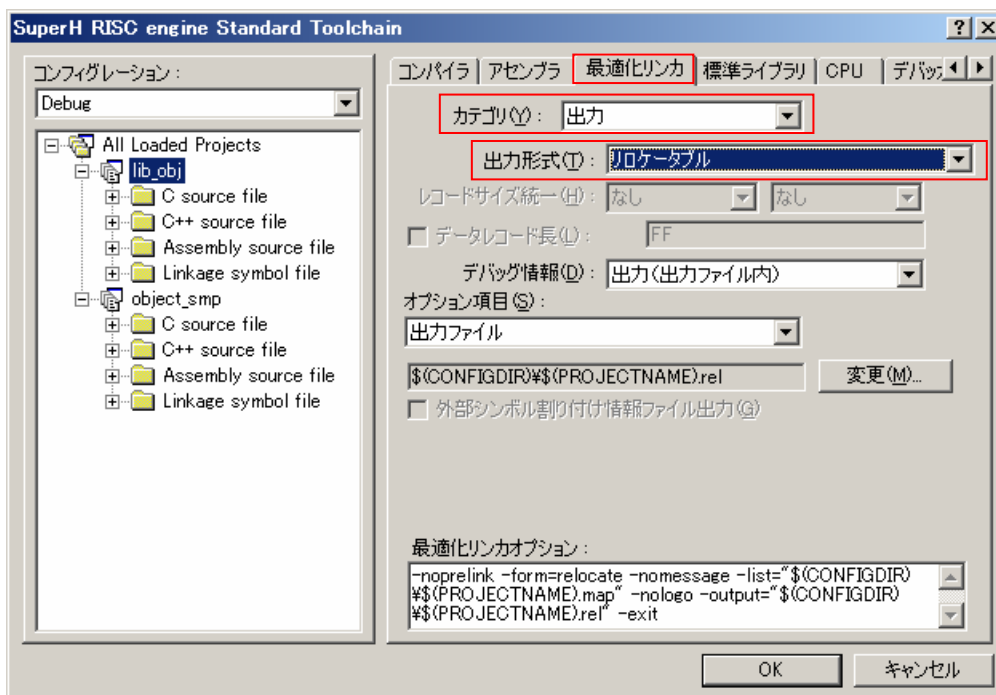


図 4-7

### (2) 最適化リンカージェディタを使用する方法

最適化リンカージェディタを使用してオブジェクトファイルを抽出する方法を説明します。まず、ライブラリリストを出力して、ライブラリのオブジェクトモジュールの情報を取得してください。ライブラリリストの出力方法は、「1.3最適化リンカージェディタ」を参照してください。次に、以下のようにextraオプションでオブジェクトモジュールを指定してください。指定したオブジェクトモジュールがオブジェクトファイルに抽出されます。

```
optlnk -extra=<オブジェクトモジュール> -form=object -library=<ライブラリファイル>
```

form=relocate オプションでリロケータブルファイルを生成してください。例えば、オブジェクトファイル\_\_i\_addd.obj、\_\_i\_adds.obj からリロケータブルファイルを生成したい場合は以下のように指定してください。

```
optlnk __i_addd.obj __i_adds.obj -form=relocate
```

5. 物理アドレスのロードモジュールを作成する方法

5.1 概要

論理アドレス空間を持つSHマイコン向けのオブジェクトを生成する場合、アドレス解決は論理アドレス空間のアドレスで行います。例えば、ルネサス統合開発環境でSH7750用のプロジェクトを生成すると、図5-1のセクション割り当てになっています。RSTHandlerは0xA0000000に配置されていますが、これは論理アドレス空間でのアドレス表現です。ルネサス統合開発環境でこのロードモジュールをダウンロードした場合は、自動的に上位3bitが無視され、物理アドレス空間のアドレスにデータがダウンロードされます。つまり、RSTHandlerは0xA0000000番地ではなく、0x00000000番地にダウンロードされます。しかし、サードパーティー製のフラッシュ書き込みツールなどでは、上位3ビットを無視する機能が無い場合があります。この場合は、論理アドレス空間で作成されたロードモジュールをダウンロードすると、論理アドレス空間のアドレスにそのままダウンロードされます。例の場合では、RSTHandlerが0xA0000000番地にダウンロードされます。この現象を回避するためには、アドレス解決が論理アドレス空間のアドレスで行われており、かつ、データ配置が物理アドレス空間のアドレスであるロードモジュールを生成する必要があります。

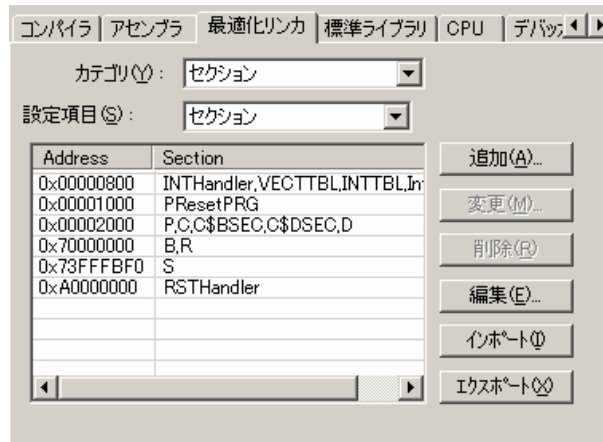


図 5-1

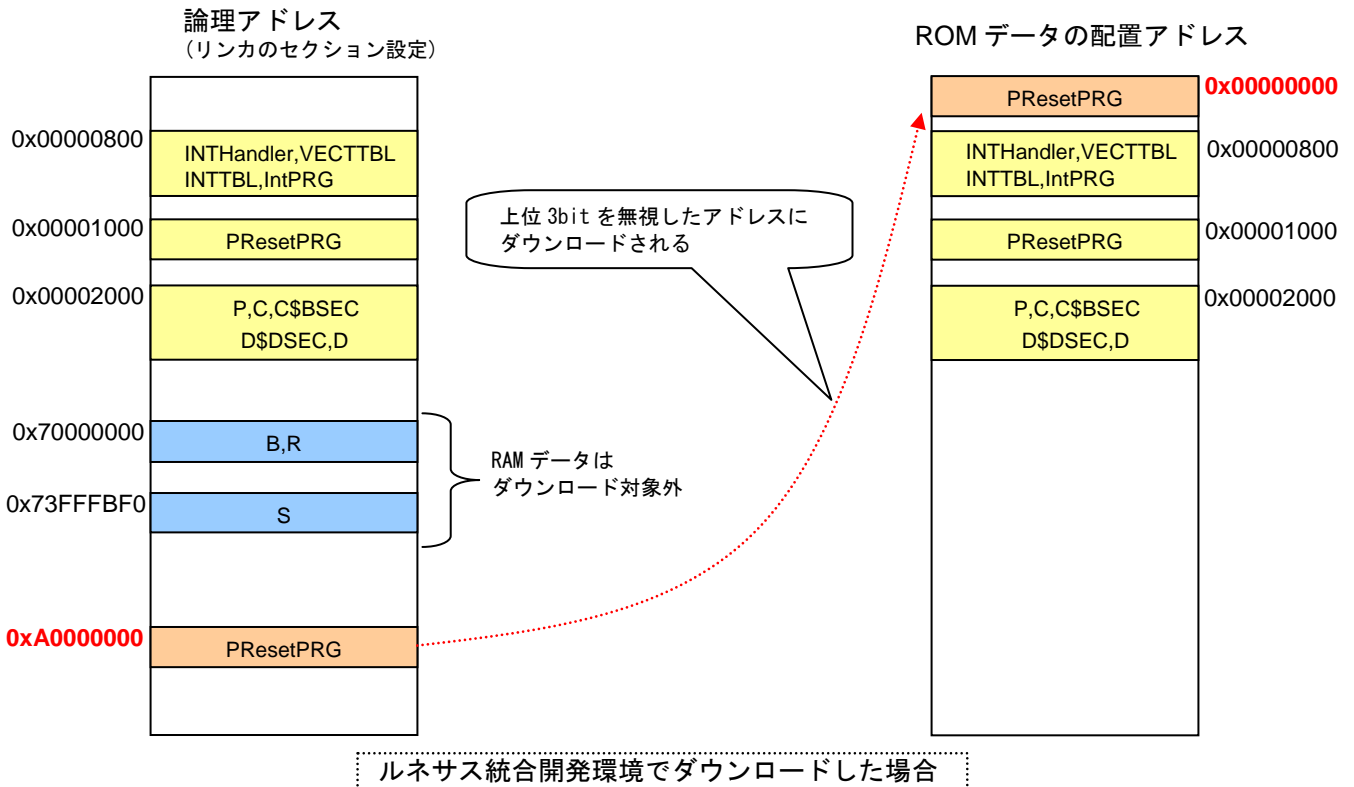


図 5-2

## 5.2 作成方法

物理アドレス空間に配置されたロードモジュールを作成するためには、最適化リンカージェネレータのROM化支援機能を使用します。ROM化支援機能は、初期値を持つ変数をROMからRAMに転送して使用するときなどに利用される機能です。この機能を使うと、アドレス解決をRAMのアドレスで行い、データはROMのアドレスに配置することができます。

物理アドレス空間に配置されたロードモジュールを生成するには、物理アドレス空間をROMデータに、論理アドレス空間をRAMに見立てて、ROM化支援機能を使用します。SH7750での設定例を記します。

### (1) セクション設定

ROM化支援機能の設定をする前にセクションの設定を行います。まず、RSTHandlerの配置アドレスの上位3bitを0に置き換えて、RSTHandlerの配置を論理アドレス空間のアドレスから物理アドレス空間のアドレスに変更します。次に、アドレス解決用の新規セクションを元々RSTHandlerが配置されていた論理アドレス空間のアドレスに確保します。例ではアドレス解決用の新規セクションV\_RSTHandlerを0xA0000000番地に確保します。

[SuperH RISC engine Standard Toolchain]ダイアログボックスの[最適化リンカ]タブで以下のように設定します。

[カテゴリ]: “セクション”

[設定項目]: “セクション”

[変更]: RSTHandlerの配置アドレスを0xA0000000番地から0x00000000番地へ変更

[追加]: 0xA0000000番地にV\_RSTHandlerを確保

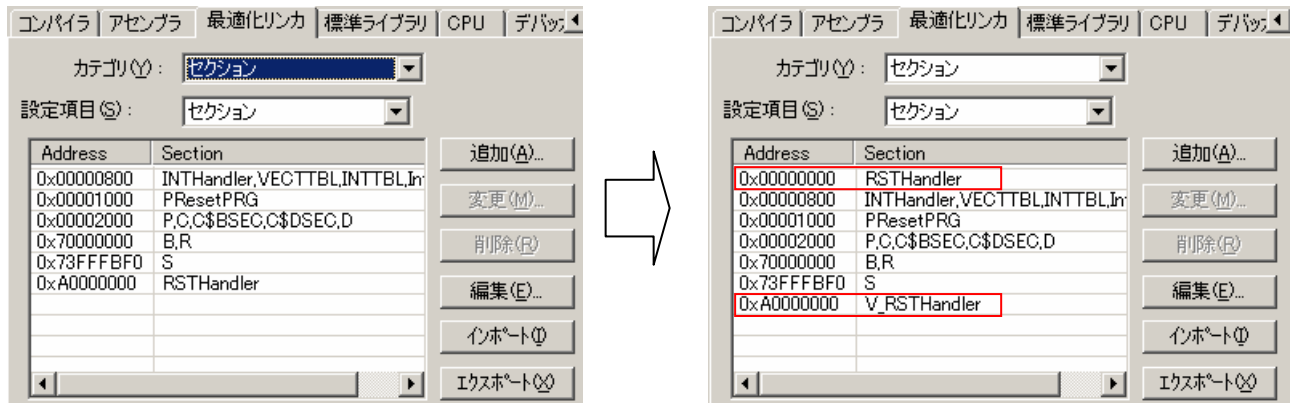


図 5-3

### (2) ROM化支援機能の設定

[SuperH RISC engine Standard Toolchain]ダイアログボックスの[最適化リンカ]タブでROM化支援機能の設定を以下のようにします。

[カテゴリ]: “出力”

[オプション項目]: “ROMからRAMへマップするセクション”

[追加]: 表示される[Modify Rom to Ram]ダイアログボックスで、次の内容を設定

[ROMセクション]: RSTHandler

[RAMセクション]: V\_RSTHandler

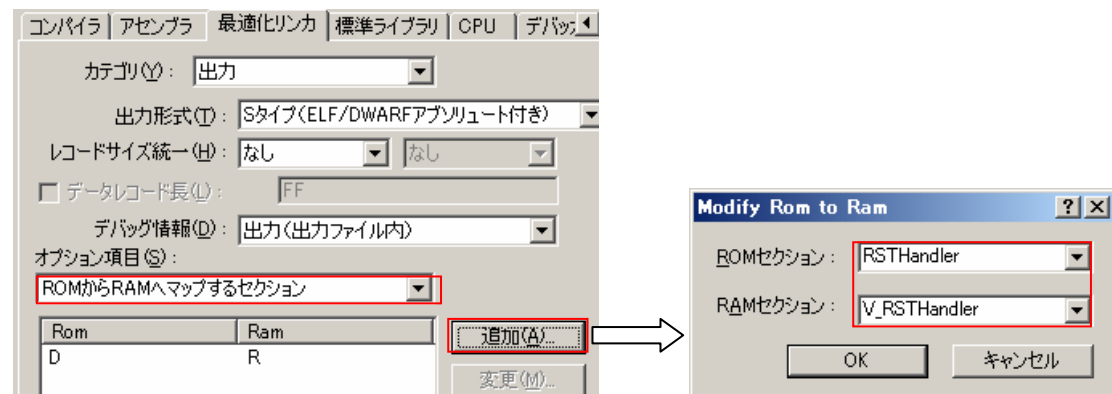


図 5-4

サポート窓口<website and support,ws>

ルネサステクノロジホームページ

<http://japan.renesas.com/>

お問合せ先

<http://japan.renesas.com/inquiry>

[csc@renesas.com](mailto:csc@renesas.com)

改訂記録<revision history,rh>

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2008.4.1	—	初版発行

## 安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

## 本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(<http://www.renesas.com>)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したのですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任を負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気づきの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。