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。

お客様各位

資料中の「日立製作所」、「日立XX」等名称の株式会社ルネサス テクノロジへの変更について

2003年4月1日を以って三菱電機株式会社及び株式会社日立製作所のマイコン、ロジック、アナログ、ディスクリート半導体、及びDRAMを除くメモリ（フラッシュメモリ・SRAM等）を含む半導体事業は株式会社ルネサス テクノロジに承継されました。従いまして、本資料中には「日立製作所」、「株式会社日立製作所」、「日立半導体」、「日立XX」といった表記が残っておりますが、これらの表記は全て「株式会社ルネサス テクノロジ」に変更されておりますのでご理解の程お願い致します。尚、会社商標・ロゴ・コーポレートステートメント以外の内容については一切変更しておりませんので資料としての内容更新ではありません。

ルネサステクノロジ ホームページ（http://www.renesas.com）

2003年4月1日
株式会社ルネサス テクノロジ
カスタマサポート部

RENESAS

ご注意

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、慎重に制作したも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は負いません。
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、その他お気付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。

User's Manual

UNIX HI8-3H

構築マニュアル

ルネサスマイクロコンピュータサポートソフトウェア
Industrial Realtime Operating System H8/300H

ルネサス エレクトロニクス
www.renesas.com

Rev.2.00　1994.06

本製品は、東京大学理学部情報科学科　坂村　健博士のご指導のもとに開発されたもので、TRON 仕様にもとづいています。

はじめに

本マニュアルでは、μITRON[*1]仕様に準拠した産業機器組み込み用リアルタイム・マルチタスクOS(Operating System)であるＨＩ８－３Ｈ(Hitachi Industrial Realtime Operating System H8/300H)のシステム構築方法について説明します。
本マニュアルは、UNIX[*2]を使用して、ＨＩ８－３Ｈのシステムの構築を行なう手順を説明します。

ＨＩ８－３Ｈの機能については、ＨＩ８－３Ｈユーザーズマニュアルを参照してください。

＜マニュアルの構成＞
本マニュアルの構成を以下に示します。
第１章では，ＨＩ８－３Ｈのシステムの構築手順の概略を記述しています。
第２章では，セットアップテーブルを用いたＨＩ８－３Ｈのシステム構築方法を記述しています。
第３章では，Ｃ言語を用いたユーザプログラムとの結合方法について記述しています。
付録では，メモリ容量の算出表，標準提供システムの作成例を記述しています。

＜関連マニュアル＞
関連マニュアルは以下のとおりです。
・ＨＩ８－３Ｈ　ユーザーズマニュアル
・Ｈ８／３００シリーズ　クロスアセンブラ　ユーザーズマニュアル
・Ｈ８／３００シリーズ　Ｃコンパイラ　ユーザーズマニュアル
・Ｈシリーズ　リンケージエディタ　ユーザーズマニュアル
・Ｈシリーズ　ライブラリアン　ユーザーズマニュアル
・Ｈ８／３００Ｈシリーズ　ハードウェアマニュアル
・E7000　H8/3003.H8/3002.H8/3042シリーズ　エミュレータ　ユーザーズマニュアル

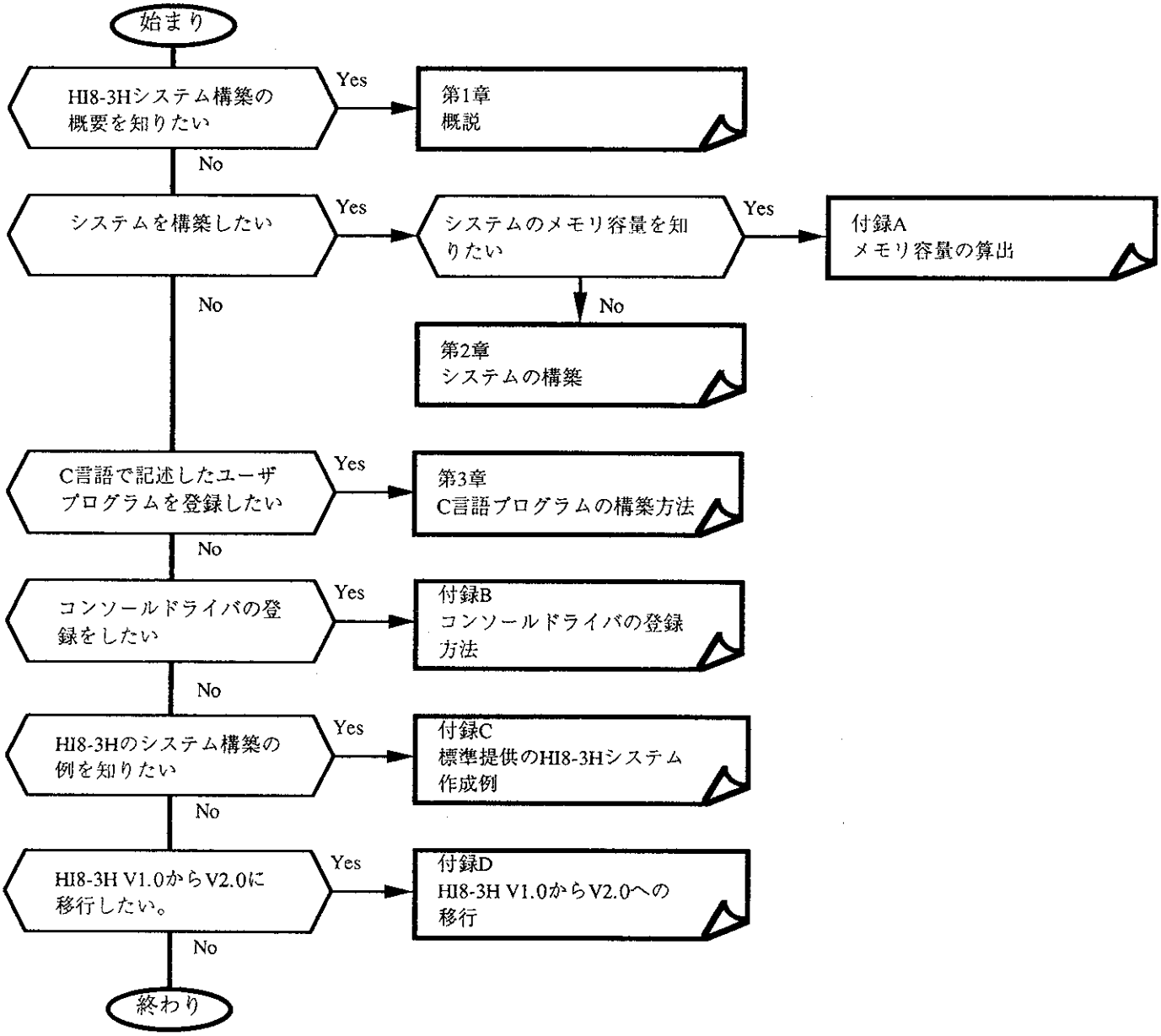
＜本マニュアルで使用する記号などの意味＞

(RET)	RETURNキーを示し、コマンド入力の終了を意味します。
＿＿＿	アンダーラインの部分は、ユーザがキー入力するコマンドラインです。
{ }	いずれかひとつを選択することを意味します。
[ ]	省略可能であることを意味します。
%	コマンドライン上のプロンプトです。
△	１つ以上のスペースまたはタブを意味します。
H'	整数定数の16進数には、H'を付けています。それ以外は10進数です。
< >	この記号で囲まれた内容を指定することを示します。

＜注意＞
HI8-3H Ver.2.0は、HI8-3H Ver.1.0に対し、次の機能向上を行なった製品です。
(1)H8/300Hノーマルモードをサポートしました。
(2)システムの作業領域を抑止する共有スタック機能をサポートしました。
(3)オンラインデバッガ用の機能としてトレース機能をサポートしました。

*1　μITRONは，"Micro Industrial TRONの略称です。
TRONは，"The Real time Operating System Nucleus"の略称です。
*2　UNIXは，米国UNIX System Laboratories,Inc.により管理されている、オペレーティングシステムの名称です。

マニュアルを読まれる前に、知りたい事項を下記フローからピックアップされることをおすすめします。



目　次

図　目　次

表　目　次

〈第２章　システムの構築〉

〈第３章　Ｃ言語プログラムの構築方法〉

〈付録Ａ　メモリ容量の算出〉

〈付録Ｂ　コンソールドライバの登録方法〉

〈付録Ｃ　標準提供のHI8-3Hシステム作成例〉

1．概　説

1．1　概　要

ＨＩ８－３Ｈは、H8/300Hシリーズ ＣＰＵで動作する、μITRON仕様に準拠した産業機器組込み用リアルタイム・マルチタスクOSです。

ＨＩ８－３Ｈの機器組み込みには、以下に示すシステム構築作業が必要です。

・ユーザプログラムの作成
・セットアップテーブルの作成（カーネルの環境を設定します）
・実行形式プログラムの作成（実行可能なロードモジュールを作成します）

1．2　システムの構築手順

システムの構築手順を以下に示します。

(1) ユーザプログラムの作成

アセンブリ言語またはＣ言語でユーザプログラムを記述し、アセンブル，コンパイルを行なってオブジェクトモジュールを作成します。

(2) セットアップテーブルの作成

HI8-3Hを動作させるために必要なシステムの環境を設定するセットアップテーブルを、アセンブリ言語で記述し、アセンブルを行なってオブジェクトモジュールを作成します。

(3) 実行形式プログラムの作成

ユーザプログラム、セットアップテーブルの各オブジェクトモジュール、HI8-3Hのライブラリファイルをリンケージエディタで結合し、実行形式プログラムを作成します。

HI8-3Hはリンケージエディタで結合するだけで、自動的にOSの機能モジュールの選択が行なわれます。

1．3　プログラムのロード

作成した実行形式プログラムをユーザ実機（ターゲットシステム）にロードするためには、以下の方法があります。

(1) H8/300Hシリーズ用E7000を使用する方法

H8/300Hシリーズ用E7000を使用して、ユーザ実機（ターゲットシステム）に実行形式プログラムをロードします。

(2) ROM を使用する方法

作成した実行形式プログラム（アブソリュートオブジェクト）をHシリーズオブジェクトコンバータでモトローラSタイプ形式のロードモジュールに変換し、H8/300Hシリーズ の内蔵ROM または外部ROM にプログラムを書き込み、ユーザ実機に搭載します。

図１－１にシステムの構築手順を示します。

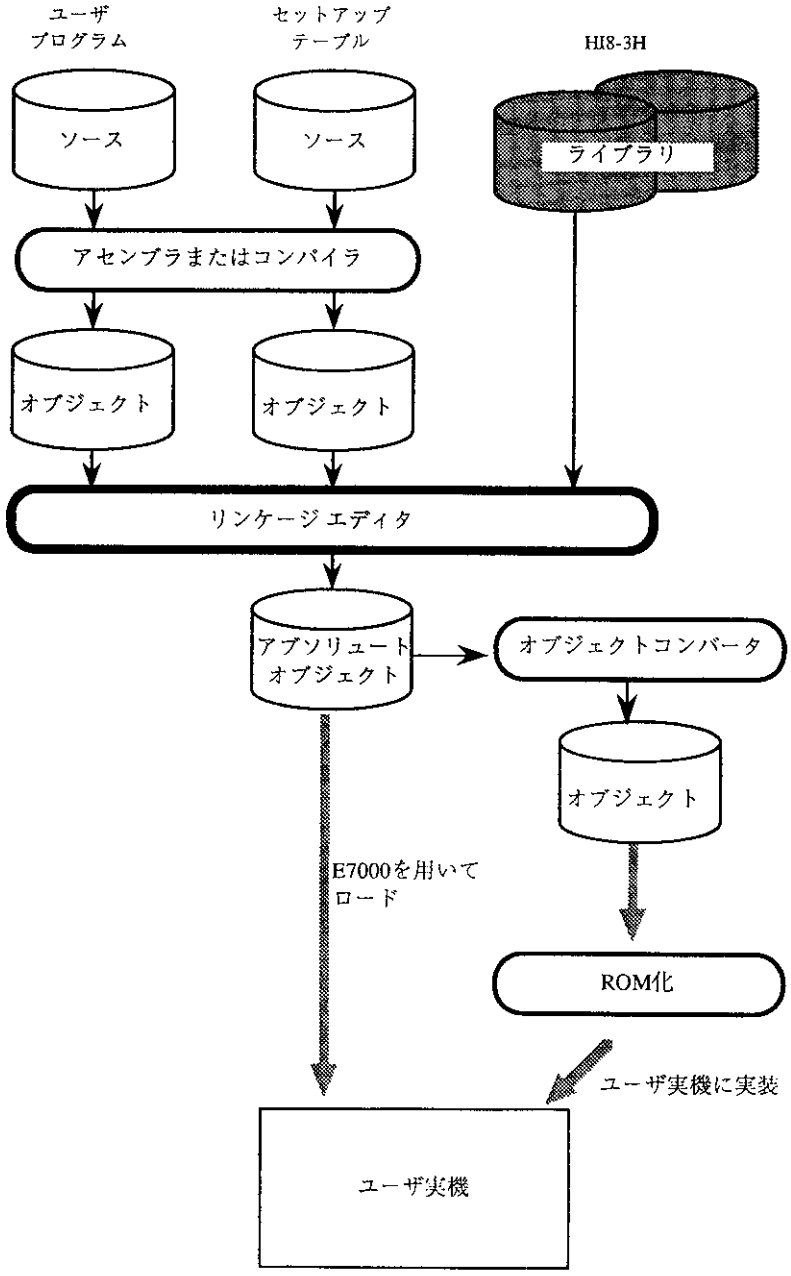


図１－１　システムの構築手順

2．システムの構築

2．1　概　要

HI8-3Hを使用したシステムの構築には、次の作業が必要になります。

(1) ユーザプログラムの作成

タスク、割込みハンドラ、システム初期化ハンドラなどユーザシステムに必要なプログラムを作成します。

(2) タイマドライバ、システム異常終了ルーチン、CPU初期化ルーチンのプログラム作成

HI8-3Hに必要なタイマドライバ、システム異常終了ルーチン、CPU初期化ルーチンのプログラムを作成します。

(3) セットアップテーブルの作成

HI8-3Hが必要とする作業領域やタスクなどの資源の情報を定義したセットアップテーブルを作成します。

(4) 割込みベクタテーブルの作成

割込みハンドラの先頭アドレスを登録した割込みベクタテーブルを作成します。

(5) 実行形式プログラムの作成

(1)～(4)で作成したファイルをアセンブルまたはコンパイルしてオブジェクトモジュールを作成し、結合して実行形式プログラムを作成します。

図２－１にシステムの構築手順の概要を示します。

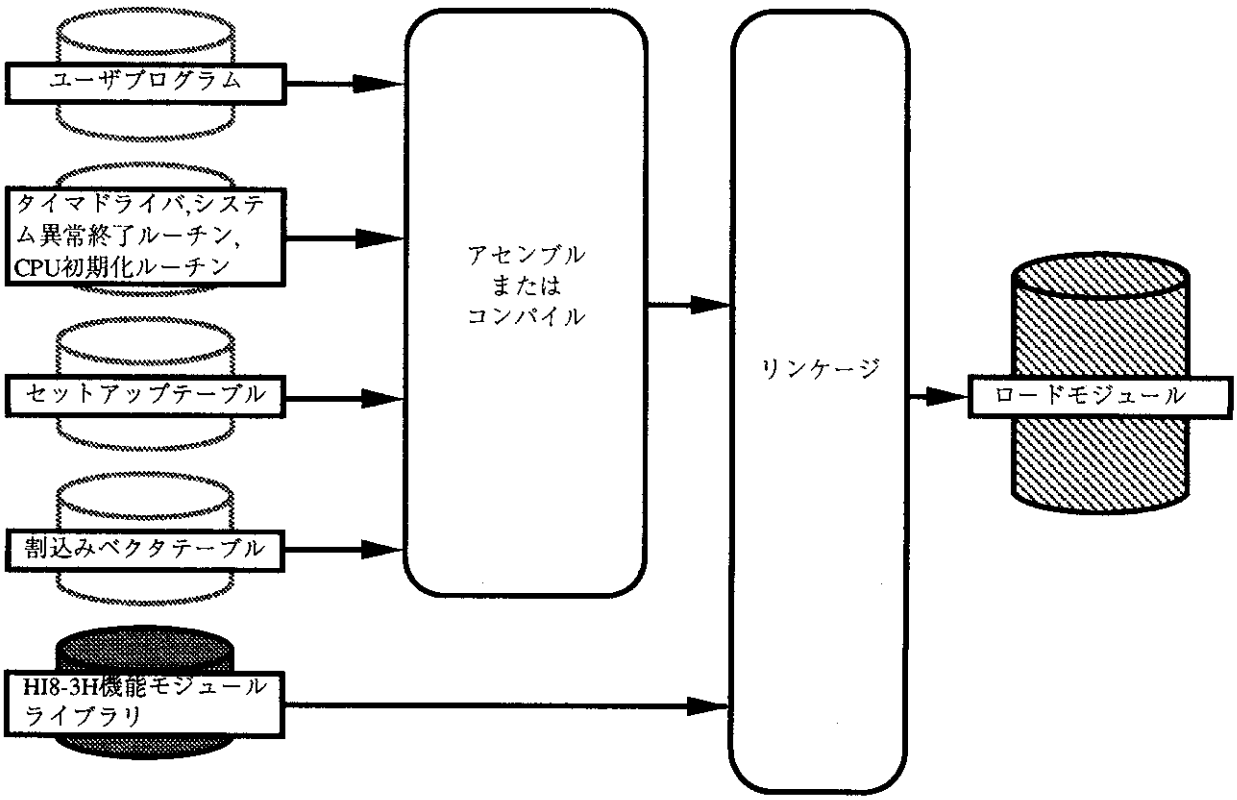


図２－１　システムの構築手順の概要

２．２　ユーザプログラムの作成

ユーザのシステムに必要となるプログラムを、アセンブリ言語またはＣ言語により作成します。作成するプログラムを以下に示します。

・タスク
・割込みハンドラ
・システム初期化ハンドラ

２．２．１　タスクの作成

ユーザシステムが行なう作業の１単位をタスクとして分割します。タスクの優先度はタスク毎に設定できます。

構築するシステムに応じて、タスクを作成します。

作成タスクは、セットアップテーブルに登録します。タスクの登録方法については、「２．４．２　タスクの登録」を参照してください。

なお、タスクについての詳細は、H18-3Hユーザーズマニュアルの「２．２　タスク」を参照してください。

２．２．２　割込みハンドラの作成

割込みハンドラとは、割込みが発生した時、実行されるプログラムです。割込みが発生するとカーネルの介入なしに直ちに起動されます。

構築するシステムに応じて、各種割込みに対応した割込みハンドラを作成します。

作成した割込みハンドラは、割込みベクタテーブルに登録します。

なお、割込みハンドラについての詳細は、H18-3Hユーザーズマニュアルの「２．６　割込み」を参照してください。

２．２．３　システム初期化ハンドラの作成

システム初期化ハンドラとは、H18-3Hのシステム起動処理を行なった後、タスクを起動する前に実行されるプログラムです。システム初期化ハンドラを使用すると、タスク起動前に任意の初期処理を行なうことができます。

構築するシステムに応じてシステム初期化ハンドラを作成します。

標準提供のシステム初期化ハンドラ（ラベル名『_HIPRG_SYSINI』）は、次のファイルに含まれています。

- 『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
- 『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

システム初期化ハンドラの登録方法については、「２．６　システム初期化ハンドラの登録」を参照してください。

なお、システム初期化ハンドラについての詳細は、H18-3Hユーザーズマニュアルの「２．９　H18-3Hのシステム起動処理」を参照してください。

２．３　タイマドライバ、システム異常終了ルーチン、CPU初期化ルーチンのプログラム作成

HI8-3Hに必要なプログラムを、アセンブリ言語またはＣ言語により作成します。

作成するプログラムを以下に示します。

・タイマドライバ(タイマ初期設定ルーチン、タイマ割込みハンドラ)
・システム異常終了ルーチン
・CPU初期化ルーチン

２．３．１　タイマドライバの作成

HI8-3Hは、ハードウェアからの一定周期の割込みを利用して、時間管理を行ないます。

標準提供の時間管理は、ハードウェアタイマとしてH8/300H 内蔵の16ビットインテグレーテッドタイマパルスユニット*1（以下、ITU と略す)を使用しています。

標準提供のタイマドライバは、次のファイルに含まれています。

・『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
・『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

タイマドライバの登録／未登録方法については、「２．７　タイマドライバの登録」を参照してください。

なお、タイマドライバについての詳細は、HI8-3Hユーザーズマニュアルの「２．８．４　タイマドライバ（タイマ初期設定ルーチン、タイマ割込みハンドラ）」を参照してください。

(1)タイマ初期設定ルーチン(ラベル名『_HIPRG_TIMINI』)

システムクロックとして使用するハードウェアタイマの初期化を行なうプログラムです。タイマ割込みの割込みレベルは、セットアップテーブルに設定したカーネル割込みマスクレベルより高いレベルにしないでください。

(2)タイマ割込みハンドラ(ラベル名『H_3H_TIM』)

ハードウェアタイマの割込み発生後、ハードウェアタイマの割込みをクリア（タイマ割込みリセット処理）して、ＯＳのタイマ割込み処理にジャンプするプログラムです。

ハードウェアタイマが、割込みをクリアする必要があるときに作成します。

【注】*1 ITU の詳しい説明は、各CPUのハードウェアマニュアルを参照してください。

(3)タイマ周期の変更

標準提供のタイマドライバは、ITU のジェネラル・レジスタＡ０(GRA0)をアウトプットコンペアレジスタとして使用し、タイマ周期を10msecに設定しています。

このタイマ周期を容易に変更できるように、次のファイルの中でequate定義しています。

- 『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
- 『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

以下にequate定義箇所を示します。

```
248行  ;###### setting data ###################;:
249行  ;
250行  GRA_DATA:       .equ    h'9c3f;d'40000-1;:(10000us/(p/4))-1:10ms=10000us,p=16MHz
251行  TCR_DATA:       .equ    CLR0|TPS1       ;:TCNT clear to GRA compare match
252行  ;                                       ;:pre scale to p/4
```

(a)ジェネラル・レジスタＡ０(GRA0)設定データ（ラベル名『GRA_DATA』）

アウトプット・コンペア・レジスタＡ０(GRA0)の設定データを指定します。

(b)タイマ・コントロール・レジスタ０(TCSR0)の設定データ（ラベル名『TCR_DATA』）

①カウンタ・クリア

GRA0のコンペア・マッチでタイマカウンタ(TCNT)をクリアするように、指定します。

②タイマ・プリスケーラ

タイマのカウンタ・クロックを指定します。

4種類の内部クロック(φ, φ/2, φ/4, φ/8)を指定します。

タイマ周期は、ＧＲＡ０設定データとタイマ・プリスケーラの値で決定します。

タイマ周期は、以下の式で算出します。

- タイマ周期　：x (sec)
- タイマ・プリスケーラ　：n
- ＧＲＡ０設定データ＝ x × n − 1

（例：標準提供の値）

CPU クロック(φ)　＝16M (Hz)
タイマ・プリスケーラ＝φ/4
タイマ周期　＝10m (sec)
ＧＲＡ０設定データ　＝0.01(16.000.000/4)-1
＝40.000-1
＝H'9c3f

CPUクロック(φ)が16MHz の場合，タイマ周期は以下の範囲まで設定できます。

タイマ・プリスケーラ：φ　＝ 0.125μ(sec) ～ 4.095 m(sec)
タイマ・プリスケーラ：φ/2＝ 0.25 μ(sec) ～ 8.191 m(sec)
タイマ・プリスケーラ：φ/4＝ 0.5 μ(sec) ～ 16.383 m(sec)
タイマ・プリスケーラ：φ/8＝ 1.0 μ(sec) ～ 32.767 m(sec)

(4)他のH8/300Hシリーズを使用する場合の注意

H8/3003またはH8/3042のITU 以外のタイマを利用する場合、新たにタイマドライバを作成してください。ただし、H8/300H内蔵のITUをハードウェアタイマとして使用する場合は、標準提供のタイマドライバのITUの各レジスタアドレスの定義を変更するだけで使用可能です。

ITUの各レジスタアドレスは、以下の標準提供のタイマドライバファイルの37行以降に定義しています。

- 『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
- 『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

２．３．２　システム異常終了ルーチンの作成

システム異常終了ルーチンはHI8-3Hのシステム実行中に致命的なエラーが発生した場合、起動されるプログラムです。

システム異常終了ルーチンが起動される場合、スタックにはエラー情報が設定されています。異常終了に応じたプログラムを作成する場合、スタックのエラー情報を参照してください。

システム異常終了ルーチンのプログラムの先頭には、ラベル名『_HIPRG_ABNOML』を付けてください。

標準提供のシステム異常終了ルーチンは、次のファイルに含まれています。

・『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
・『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

標準提供のシステム異常終了ルーチンは、割込みマスク(カーネル割込みマスクレベル)の状態で無限ループに入ります。

システム異常終了ルーチンの登録方法については、「２．８　システム異常終了ルーチンの登録」を参照してください。

なお、システム異常終了ルーチンの詳細については、HI8-3Hユーザーズマニュアルの「２．１１　HI8-3Hのシステムの異常終了処理」を参照してください。

システム異常終了ルーチンは、必ず作成してください。 システム異常終了ルーチンが無い場合、動作は保証されません。

２．３．３　CPU初期化ルーチンの作成

HI8-3Hのシステム起動前にCPUの初期化を行ない、HI8-3Hのシステム起動処理にジャンプするプログラムです。構築するユーザシステムに応じて作成してください。

CPU初期化ルーチンのプログラムの先頭には、ラベル名『H_3H_CPUINI』を付けてください。

標準提供のCPU初期化ルーチンは、次のファイル名です。

・『cpuini.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
・『cpuinin.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

標準提供のCPU初期化ルーチンは、システム・コントロール・レジスタを設定しています。

CPU初期化ルーチンの登録方法については、「２．５．１　割込みハンドラの登録」を参照してください。

なお、CPU初期化ルーチンの詳細については、HI8-3Hユーザーズマニュアルの「２．１０　ＣＰＵの初期化」を参照してください。

CPU初期化ルーチンは、必ず作成してください。 CPU初期化ルーチンが無い場合、動作は保証されません。

２．４　セットアップテーブルの作成

HI8-3Hを構築する場合に必要な情報を、セットアップテーブルに設定します。

セットアップテーブルは、H8/300Hノーマルモードと H8/300Hアドバンストモードで共通に使用できます。

・ファイル名『h3hsetup.mar』,『setup.cmn』:H8/300Hアドバンストモード(sampleaﾃﾞｨﾚｸﾄﾘ)
／H8/300Hノーマルモード(samplenﾃﾞｨﾚｸﾄﾘ)

標準提供のセットアップテーブル(ファイル名『h3hsetup.mar』,『setup.cmn』)の構成を図２－２に示します。

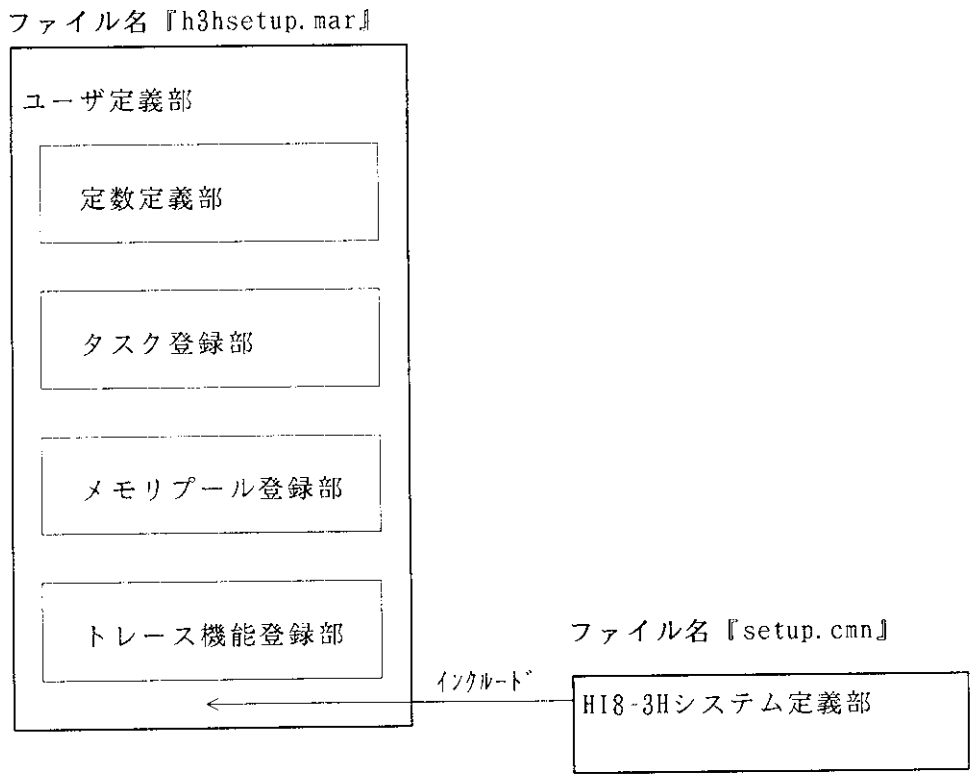


図２－２　セットアップテーブルの構成

セットアップテーブルは、ユーザ定義部とHI8-3Hシステム定義部から構成されています。

ユーザ定義部は、タスク定義数などユーザ環境の設定を行なうテーブルです。ユーザシステムの環境に合せて修正してください。

HI8-3Hシステム定義部は、HI8-3Hが使用する外部定義シンボルおよび作業領域などを定義しています。

HI8-3Hシステム定義部は、修正しないでください。 HI8-3Hシステム定義部を修正した場合、動作は保証されません。

以下にユーザ定義部の各定義項目を示します。

(1) 定数定義部

定数定義部には、HI8-3H、同期通信機能および時間管理機能を使用するために必要な情報を登録します。

- IMASK(カーネル割込みマスクレベル) *1
- MAXPRI(優先度定義数) *1
- FLGCNT(イベントフラグ定義数) *1
- SEMCNT(セマフォ定義数) *1
- MBXCNT(メイルボックス定義数) *1
- OSSTKSIZ(ＯＳスタックサイズ) *1
- TIMSTKSIZ(タイマスタックサイズ) *1
- TRCSTKSIZ(トレーススタックサイズ) *1

(2)タスク登録部

タスク登録部には、タスクを使用するために必要な情報を登録します。

- TSKSTKSIZ(タスク最小スタックサイズ) *1
- TSKx_SP(タスクスタック領域) *2
- _HI_TDT(タスク定義テーブル) *1
- TSKCNT(タスク定義数) *1

(3)メモリプール登録部

メモリプール登録部には、メモリプールを使用するために必要な情報を登録します。

- MBx_CNT(メモリブロック数) *2
- MBx_LEN(メモリブロック長) *2
- MPLx_TOP(メモリプール領域) *2
- _HI_MPLDT(メモリプール定義テーブル) *1
- MPLCNT(メモリプール定義数) *1

(4)トレース機能登録部

トレース機能登録部には、トレース機能を使用するために必要な情報を登録します。

- TRC_CNT(最大トレース情報取得数)
- TRC_BUF(トレースバッファ領域)
- INITRC(トレースバッファ情報テーブル)

登録／未登録に関わらず、全ての項目を設定してください。 設定されていない場合、結合時に未定義エラーとなります。

【注】 *1 これらのラベル名は、HI8-3Hが参照しているため、変更しないでください。
*2 これらのラベル名は、標準提供のセットアップテーブルの例です。
標準提供のセットアップテーブルでは、 ラベル名の xにID番号(1～4)を設定しています。

2．4．1　定数定義部の登録

表２－１に定数定義部の登録情報を示します。

表２－１　定数定義部の登録情報一覧

項番	ラベル名	内容	ファイル名
1	IMASK *1	カーネル割込みマスクレベル	h3hsetup.mar
2	MAXPRI *1	優先度定義数	
3	FLGCNT *1	イベントフラグ定義数	
4	SEMCNT *1	セマフォ定義数	
5	MBXCNT *1	メイルボックス定義数	
6	OSSTKSIZ *1	ＯＳスタックサイズ	
7	TIMSTKSIZ *1	タイマスタックサイズ	
8	TRCSTKSIZ *1	トレーススタックサイズ	

【注】*1　これらのラベル名は、HI8-3Hが参照しているため、変更しないでください。

```
        .radix  d                       ;;xxxxx -> d'xxxxx
;
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%    VALUE define section                                            %%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;------ Usage ---------------------------------------------------------------
;LABEL                  VALUE   ;;[ RANGE ]     ;; COMMENT
;----------------------------------------------------------------------------
IMASK:          .equ    03      ;;[0.....3]     ;; Max interrupt level
MAXPRI:         .equ    31      ;;[0....31]     ;; Max low priority
FLGCNT:         .equ    4       ;;[0...255]     ;; Eventflag definition count
SEMCNT:         .equ    4       ;;[0...255]     ;; Semaphore definition count
MBXCNT:         .equ    4       ;;[0...255]     ;; Mailbox definition count
;
OSSTKSIZ:       .equ    8+4+4+4+6;;[8.....]     ;; OS stack size
TIMSTKSIZ:      .equ    30+6+4+6;;[0.30...]     ;; Timer stack size
TRCSTKSIZ:      .equ    26+4+4+4+6;;[0.26...]   ;; Trace stack size
;
```

図２－３　セットアップテーブルの定数定義部（H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用）

(1)IMASK(カーネル割込みマスクレベル)

カーネルは、カーネルの持つ情報の整合性を保つために割込みをマスクして処理することがあります。『IMASK』には、その割込みマスクレベルを設定してください。

設定できる値は、 0～ 3のいずれかです。

『IMASK』に設定した値より高いレベルの割込みは、カーネル実行中に発生しても遅延されずに受け付けられます。表２－２にカーネル割込みマスクレベルと割込みの関係を示します。

表２－２　カーネル割込みマスクレベルと割込みの関係

項番	ｶｰﾈﾙ割込みﾏｽｸﾚﾍﾞﾙ(IMASK)	CCR設定値(I,UIﾋﾞｯﾄ)	プライオリティレベル１の割込み（優先）	プライオリティレベル０の割込み（非優先）
1	3	I=1,UI=1	遅延する	遅延する
2	2	I=1,UI=0	遅延しない	遅延する
3	1	I=0,UI=1	遅延しない	遅延しない
4	0	I=0,UI=0	遅延しない	遅延しない

【注】カーネル割込みマスクレベル『IMASK』に０または１を設定した場合、時間管理機能を使用できません。また、『IMASK』に設定したレベルより高い割込みは NMIと同等に扱われる為、その割込みに対応する割込みハンドラではシステムコールを発行できません。

(2)MAXPRI(優先度定義数)

『MAXPRI』には、構築するシステムでタスクの最も低い優先度を設定します。

設定できる値は、 1～31までです。値の小さい方が高い優先度を表します。

(3)FLGCNT(イベントフラグ定義数)

『FLGCNT』には、構築するシステムに必要なイベントフラグの総数(最大イベントフラグID)を設定します。

『FLGCNT』が 0の場合、イベントフラグは未登録となります。

HI8-3Hは、最大255 までのイベントフラグを登録することができます。

(4)SEMCNT(セマフォ定義数)

『SEMCNT』には、構築するシステムに必要なセマフォの総数(最大セマフォID)を設定します。

『SEMCNT』が 0の場合、セマフォは未登録となります。

HI8-3Hは、最大255 までのセマフォを登録することができます。

(5)MBXCNT(メイルボックス定義数)

『MBXCNT』には、構築するシステムに必要なメイルボックスの総数(最大メイルボックスID)を設定します。

『MBXCNT』が 0の場合、メイルボックスは未登録となります。

HI8-3Hは、最大255 までのメイルボックスを登録することができます。

(6)OSSTKSIZ(OSスタックサイズ)

ＯＳが使用するスタック領域のサイズを設定します。

ＯＳ用スタック領域のサイズを算出する方法は、「付録Ａ　メモリ容量の算出」を参照してください。

(7)TIMSTKSIZ(タイマスタックサイズ)

タイマドライバが使用するスタック領域のサイズを設定します。

タイマ割込み用スタック領域のサイズを算出する方法は、「付録Ａ　メモリ容量の算出」を参照してください。

タイマドライバを使用しない場合、タイマスタックサイズに０を設定してください。

タイマドライバの登録方法については、「２．７　タイマドライバの登録」を参照してください。

(8)TRCSTKSIZ(トレーススタックサイズ)

トレース機能が使用するスタック領域のサイズを設定します。

トレース機能用スタック領域のサイズを算出する方法は、「付録Ａ　メモリ容量の算出」を参照してください。

トレース機能を使用しない場合、トレーススタックサイズに０を設定してください。

２．４．２　タスクの登録

表２－３にタスクを登録するために必要な情報を示します。

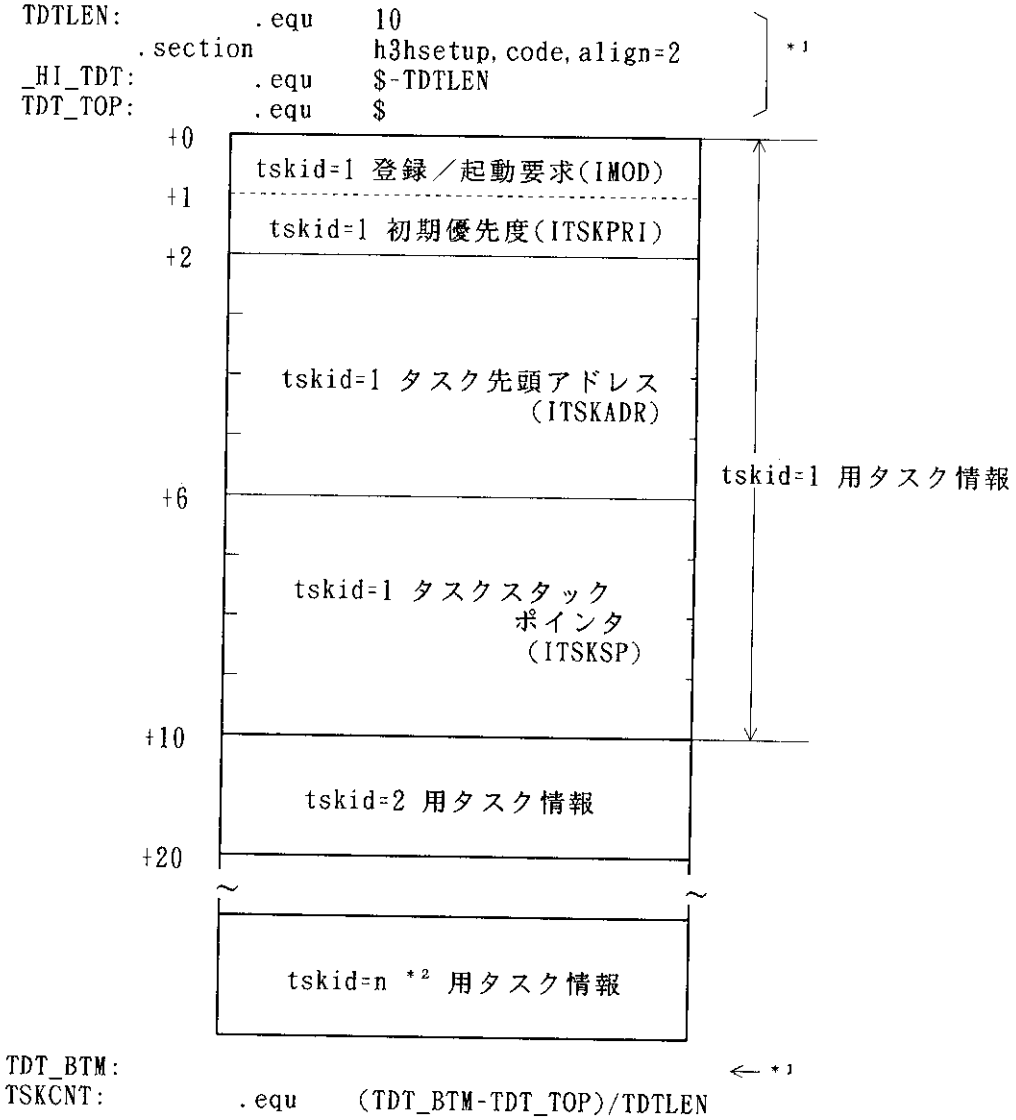
表２－３　タスク登録情報一覧

項番	ラベル名	内容	ファイル名
1	TSKSTKSIZ *1	タスク最小スタックサイズ	h3hsetup.mar
2	TSKx_SP *2	タスクスタック領域	
3	_HI_TDT *1	タスク定義テーブル IMOD　(登録／起動要求) ITSKPRI(初期優先度) ITSKADR(タスク先頭アドレス) ITSKSP (タスクスタックポインタ)	
4	TSKCNT *1	タスク定義数	

【注】*1　これらのラベル名は、HI8-3Hが参照しているため、変更しないでください。

*2　これらのラベル名は、標準提供のセットアップテーブルの例です。
標準提供のセットアップテーブルでは、ラベル名の xにID番号(1～ 5)を設定しています。

図２－４にタスク定義テーブルのフォーマットを示します。



【注】 *1 セットアップテーブル内で定義していますが、この定義は変更しないでください。
*2 nは最大タスクID(タスク定義数『TSKCNT』)です。

図２－４　タスク定義テーブルのフォーマット

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%    TASK define section                                              %%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;------ Usage -------------------------------------------------------------
;               TASK_TOP_LABEL                      ;: COMMENT
;--------------------------------------------------------------------------
        .import H_CNSDRV0                           ;: console (SCI0)
        .import H_CNSDRV1                           ;: console (SCI1)
;
;------ Usage -------------------------------------------------------------
;               .res.b  SIZE +TSKSTKSIZ ;[RANGE];: COMMENT
;TSK?_SP_LABEL: .equ    $                           ;: COMMENT
;--------------------------------------------------------------------------
TSKSTKSIZ:      .equ    32+4+4+4+4+6+16 ;[32...];: Task minimum stack size
        .section        h3hstack,stack,align=2
                .res.b  ( 4) +TSKSTKSIZ ;[32...];: tskid1 stack area
TSK1_SP:        .equ    $
                .res.b  8
                .res.b  ( 4) +TSKSTKSIZ ;[32...];: tskid2 stack area
TSK2_SP:        .equ    $
                .res.b  8
                .res.b  (32) +TSKSTKSIZ ;[32...];: tskid3 stack area
TSK3_SP:        .equ    $
                .res.b  8
                .res.b  (32) +TSKSTKSIZ ;[32...];: tskid4 stack area
TSK4_SP:        .equ    $
                .res.b  8
;
;------ Usage -------------------------------------------------------------
;LABEL          .data.b IMOD, IPRI                  ;: COMMENT
;               .data.l TSK_TOP, TSK_SP             ;: COMMENT
;--------------------------------------------------------------------------
NOEXS:          .equ    0                           ;: initial mode = NO EXIST
RDY:            .equ    1                           ;: initial mode = READY
DMT:            .equ    (-1)                        ;: initial mode = DORMANT
TDTLEN:         .equ    10;<- Not Change !          ;: TDT Length
        .section        h3hsetup,code,align=2
_HI_TDT:        .equ    $-TDTLEN                    ;: Task define table
TDT_TOP:        .equ    $                           ;:
tdt_id1:        .data.b RDY, 1                      ;: init. mode, init. priority
                .data.l H_CNSDRV0, TSK1_SP          ;: top address, stack pointer
tdt_id2:        .data.b RDY, 1                      ;: init. mode, init. priority
                .data.l H_CNSDRV1, TSK2_SP          ;: top address, stack pointer
tdt_id3:        .data.b NOEXS, 3                    ;: init. mode, init. priority
                .data.l 0, TSK3_SP                  ;: top address, stack pointer
tdt_id4:        .data.b NOEXS, 4                    ;: init. mode, init. priority
                .data.l 0, TSK4_SP                  ;: top address, stack pointer
tdt_id5:        .data.b NOEXS, 5                    ;: init. mode, init. priority
                .data.l 0, TSK4_SP                  ;: top address, stack pointer
TDT_BTM:
TSKCNT:         .equ    (TDT_BTM-TDT_TOP)/TDTLEN
                                ;:[0...255]         ;: Task definition count
;
```

図２－５　セットアップテーブルのタスク登録部(H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

(1)TSKSTKSIZ(タスク最小スタックサイズ)

『TSKSTKSIZ』には、タスクが使用するスタック領域の最小サイズを設定します。

タスク最小スタックサイズは、タスクが独自に使用するスタックを除いたサイズです。

タスクが使用するスタックサイズを算出する方法は、「付録Ａ　メモリ容量の算出」を参照してください。

(2)TSKx_SP(タスクスタック領域)

各タスクのタスクスタック領域を確保します。

確保したタスクスタック領域の最終アドレスには、タスク定義テーブルのタスクスタックポインタに設定するため、ラベル(標準提供の例では、ラベル名『TSKx_SP』)を付けてください。

共有スタック機能を使用するときは、すべてのタスクスタック領域の最終アドレス(ラベル名『TSKx_SP』)から上位アドレス側に８バイト分の共有スタック機能用の領域を確保してください。

(標準提供の例と同様に、すべてのタスクスタック領域『TSK1_SP』～『TSK4_SP』に共有スタック機能用の領域を確保してください)

各タスクのスタック領域のサイズは、以下のように設定します。

各タスクのスタック領域のサイズ＝　X　＋　TSKSTKSIZ　＋８

X	:個々のタスクが独自に使用するスタックサイズ
TSKSTKSIZ	:タスク最小スタックサイズ
８	:共有スタック機能用の制御領域

(3) _HI_TDT(タスク定義テーブル)

『_HI_TDT』は、各タスクのタスク情報を登録するテーブルの先頭アドレスです。

タスク定義テーブルには、以下のタスク情報をそれぞれタスクID＝１から順に設定します。

(a)IMOD(登録／起動要求)

該当タスクIDのタスクの登録およびシステム起動時のタスク初期状態を設定します。

NOEXS(=0)	:未登録
RDY(=1)	:起動時に実行可能状態
DMT(0,1以外)	:起動時に休止状態

(b)ITSKPRI(初期優先度)

タスクの起動時の優先度を設定します。

設定できる値は、１～『MAXPRI』(優先度定義数)の範囲内です。

(c)ITSKADR(タスク先頭アドレス)

タスクの先頭アドレスを設定します。

(d)ITSKSP(タスクスタックポインタ)

タスクが使用するスタック領域の最終アドレスを設定します。

タスクスタック領域の最終アドレスに付けたラベル(標準提供の例では、ラベル名『TSKx_SP』)を設定してください。

共有スタック領域を使用するタスクは同じラベルを設定してください。

(標準提供の例では、タスクID=4，5がタスクスタック領域『TSK4_SP』を共有スタック領域として使用します)

H18-3Hは、初期起動時のタスクスタックポインタが他の領域と重なっていないかチェックする機能を持っていません。個々のタスクスタック領域は、充分注意して確保してください。

(4)TSKCNT(タスク定義数)

『TSKCNT』には、構築するシステムに必要なタスクの総数(最大タスクID)を設定します。タスク定義テーブルで未登録にしたタスクIDもタスク定義数に含まれます。『TSKCNT』が0の場合、タスクは未登録となります。

HI8-3Hは、最大255 までのタスクを登録することができます。

『TSKCNT』は、タスク定義テーブルのサイズから算出しています。

```
TSKCNT = ( TDT_BTM - TDT_TOP ) / TDTLEN
       TDT_BTM  : タスク定義テーブルの終端ラベル
       TDT_TOP  : タスク定義テーブルの先頭ラベル
       TDTLEN   : １つのタスク情報のサイズ(１０バイト)
```

２．４．３　メモリプールの登録

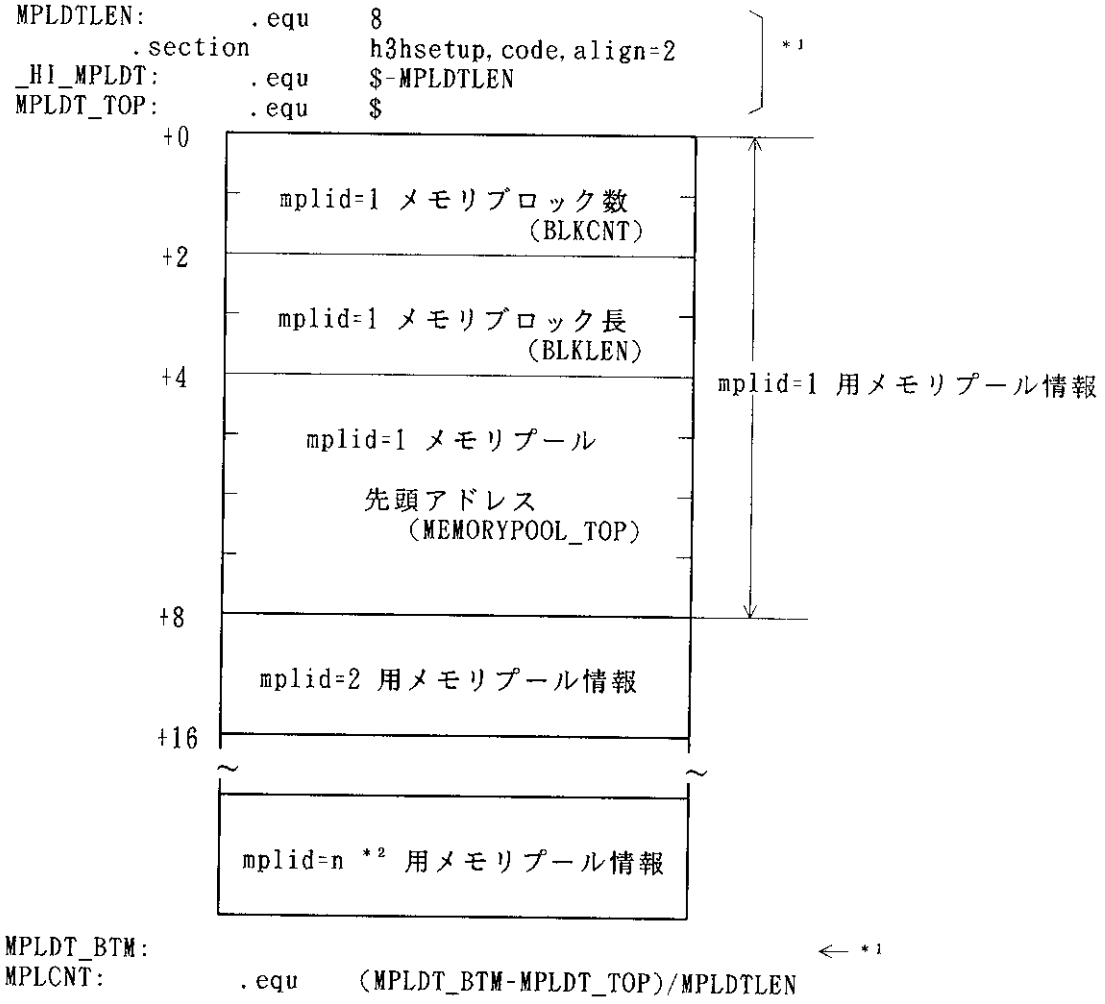
表２－４にメモリプールを登録するために必要な情報を示します。

表２－４　メモリプール登録情報一覧

項番	ラベル名	内容	ファイル名
1	MBx_CNT *2 , MBx_LEN *2	メモリブロック数, メモリブロック長	h3hsetup.mar
2	MPLx_TOP *2	メモリプール領域	
3	_HI_MPLDT *1	メモリプール定義テーブル BLKCNT(メモリブロック数) BLKLEN(メモリブロック長) MEMORYPOOL_TOP(メモリプール先頭アドレス)	
4	MPLCNT *1	メモリプール定義数	

【注】*1　これらのラベル名は、HI8-3Hが参照しているため、変更しないでください。
*2　これらのラベル名は、標準提供のセットアップテーブルの例です。
標準提供のセットアップテーブルでは、 ラベル名の xにID番号(1～ 4)を設定しています。

図２－６にメモリプール定義テーブルのフォーマットを示します。



【注】 *1 セットアップテーブル内で定義していますが、この定義は変更しないでください。
*2 nは最大メモリプールID(メモリプール定義数『MPLCNT』)です。

図２－６　メモリプール定義テーブルのフォーマット

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%    MEMORYPOOL define section                                    %%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;------ Usage ---------------------------------------------------------
;MB?_CNT_LABEL: .equ    VALUE    ;:[  RANGE  ]    ;: COMMENT
;MB?_LEN_LABEL: .equ    VALUE    ;:[  RANGE  ]    ;: COMMENT
;---------------------------------------------------------------------
MB1_CNT:        .equ    32       ;:[0...65535]    ;: memory block count
MB1_LEN:        .equ    12       ;:[2...65530]    ;: memory block length
;
MB2_CNT:        .equ    32       ;:[0...65535]    ;: memory block count
MB2_LEN:        .equ    12       ;:[2...65530]    ;: memory block length
;
MB3_CNT:        .equ    32       ;:[0...65535]    ;: memory block count
MB3_LEN:        .equ    12       ;:[2...65530]    ;: memory block length
;
MB4_CNT:        .equ    32       ;:[0...65535]    ;: memory block count
MB4_LEN:        .equ    12       ;:[2...65530]    ;: memory block length
;
;------ Usage ---------------------------------------------------------
;MPL?_TOP_LABEL:.res.b  MEMORYPOOL_SIZE           ;: COMMENT
;---------------------------------------------------------------------
        .section        h3hmpl,data,align=2
MPL1_TOP:       .res.b  MB1_CNT * (MB1_LEN + 4) ;: mplid1 memorypool area
MPL2_TOP:       .res.b  MB2_CNT * (MB2_LEN + 4) ;: mplid2 memorypool area
MPL3_TOP:       .res.b  MB3_CNT * (MB3_LEN + 4) ;: mplid3 memorypool area
MPL4_TOP:       .res.b  MB4_CNT * (MB4_LEN + 4) ;: mplid4 memorypool area
;
;------ Usage ---------------------------------------------------------
;LABEL          .data.w BLKCNT, BLKLEN            ;: COMMENT
;               .data.l MEMORYPOOL_TOP            ;: COMMENT
;---------------------------------------------------------------------
MPLDTLEN:       .equ    8;<- Not Change !         ;: MPLDT Length
        .section        h3hsetup,code,align=2
_HI_MPLDT:      .equ    $-MPLDTLEN                ;: MemoryPool define table
MPLDT_TOP:      .equ    $                         ;:
mpldt_id1:      .data.w MB1_CNT, MB1_LEN          ;: Memory block count, length
                .data.l MPL1_TOP                  ;: Memorypool top address
mpldt_id2:      .data.w MB2_CNT, MB2_LEN          ;: Memory block count, length
                .data.l MPL2_TOP                  ;: Memorypool top address
mpldt_id3:      .data.w MB3_CNT, MB3_LEN          ;: Memory block count, length
                .data.l MPL3_TOP                  ;: Memorypool top address
mpldt_id4:      .data.w MB4_CNT, MB4_LEN          ;: Memory block count, length
                .data.l MPL4_TOP                  ;: Memorypool top address
MPLDT_BTM:
MPLCNT:         .equ    (MPLDT_BTM-MPLDT_TOP)/MPLDTLEN
                                 ;:[0...255]      ;: Memorypool definition count
;
```

図２－７　セットアップテーブルのメモリプール登録部(H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

(1)MBx_CNT.MBx_LEN(メモリブロック数，メモリブロック長)

各メモリプールのメモリブロック数、メモリブロック長をequate定義します。

equate定義したメモリブロック数，メモリブロック長のラベル(標準提供の例では、ラベル名『MBx_CNT』、『MBx_LEN』)は、メモリプール領域の確保およびメモリプール定義テーブルの設定に使用します。

【注】　MBx_CNT（メモリブロック数）は、H8/300Hアドバンストモードが32、H8/300Hノーマルモードが16にequate定義されています。

(2)MPLx_TOP(メモリプール領域)

各メモリプールのメモリプール領域を確保します。

確保したメモリプール領域の先頭アドレスには、メモリプール定義テーブルのメモリプール先頭アドレスに設定するため、ラベル(標準提供の例では、ラベル名『MPLx_TOP』)を付けてください。

メモリプール領域のサイズは、以下のように設定します。

メモリプール領域のサイズ = MBx_CNT × (MBx_LEN + 4)
MBx_CNT : 該当メモリプールIDのメモリブロック数
MBx_LEN : 該当メモリプールIDのメモリブロック長

(3)_HI_MPLDT(メモリプール定義テーブル)

『_HI_MPLDT』は、各メモリプールのメモリプール情報を登録するテーブルの先頭アドレスです。

メモリプール定義テーブルには、以下のメモリプール情報をそれぞれメモリプールID=1から順に設定します。

(a) BLKCNT(メモリブロック数)

メモリブロック数は、該当メモリプールIDのメモリプール内で使用できるメモリブロックの数を設定します。0を設定すると該当メモリプールIDは、未登録となります。

(b) BLKLEN(メモリブロック長)

メモリブロック長には、該当メモリプールIDのメモリプール内で使用できるメモリブロックのサイズ(バイト)を設定します。0と奇数を設定しないでください。

(c) MEMORYPOOL_TOP(メモリプール先頭アドレス)

メモリプールとして使用するメモリ領域の先頭アドレスを設定します。

メモリプール領域の先頭アドレスに付けたラベル(標準提供の例では、ラベル名『MPLx_TOP』)を設定してください。

(4) MPLCNT(メモリプール定義数)

『MPLCNT』には、構築するシステムに必要なメモリプールの総数(最大メモリプールID)を設定します。メモリプール定義テーブルで未登録にしたメモリプールIDもメモリプール定義数に含まれます。『MPLCNT』が0の場合、メモリプールは未登録となります。H18-3Hは、最大255までのメモリプールを登録することができます。

『MPLCNT』は、メモリプール定義テーブルのサイズから算出しています。

MPLCNT = (MPLDT_BTM - MPLDT_TOP) / MPLDTLEN
MPLDT_BTM : メモリプール定義テーブルの終端ラベル
MPLDT_TOP : メモリプール定義テーブルの先頭ラベル
MPLDTLEN : 1つのメモリプール情報のサイズ(8バイト)

２．４．４　トレース機能の登録

表２－５にトレース機能を登録するために必要な情報を示します。

表２－５　トレース機能登録情報一覧

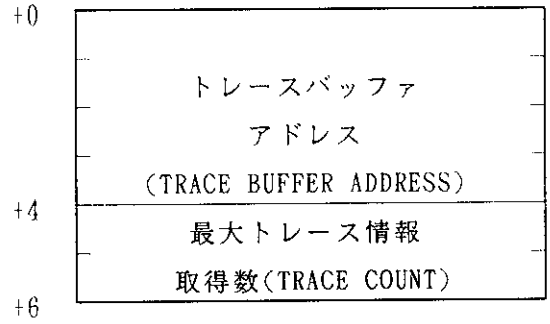
項番	ラベル名	内容	ファイル名
1	TRC_CNT *2	最大トレース情報取得数	h3hsetup.mar
2	TRC_BUF *2	トレースバッファ領域	
3	INITRC *1	トレースバッファ情報テーブル TRACE BUFFER ADDRESS （トレースバッファアドレス） TRACE COUNT(最大トレース情報取得数)	

【注】*1　このラベル名は、H8-3Hが参照しているため、変更しないでください。
*2　これらのラベル名は、標準提供のセットアップテーブルの例です。

図２－８にトレースバッファ情報テーブルのフォーマットを示します。

```
        .section        h3hsetup.code.align=2
INITRC:     .equ    $
```

*1



【注】*1　セットアップテーブル内で定義していますが、この定義は変更しないでください。

図２－８　トレースバッファ情報テーブルのフォーマット

```
;------ Usage ----------------------------------------------------------------
;TRC_CNT:.equ   TRACE COUNT                          ;: COMMENT
;TRC_BUF:.equ   TRACE BUFFER ADDRESS                 ;: COMMENT
;-----------------------------------------------------------------------------
        .section        h3htrc,data,align=2
TRC_CNT:        .equ    4                           ;: trace count
TRC_BUF:        .res.b  16+(TRC_CNT*24)             ;: trace buffer address
;
;------ Usage ----------------------------------------------------------------
;INITRC          .data.l TRACE BUFFER ADDRESS        ;: COMMENT
;                .data.w TRACE COUNT                 ;: COMMENT
;-----------------------------------------------------------------------------
        .section        h3hsetup,code,align=2
INITRC:         .equ    $                           ;:
                .data.l TRC_BUF                     ;: trace buffer address
                .data.w TRC_CNT                     ;: trace count
;
        .include        "setup.cmn"
;
;*********************************************;
        .end; of H3HSETUP.MAR
```

図２－９　セットアップテーブルのトレース機能登録部(H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

(1)TRC_CNT(最大トレース情報取得数)

トレース機能で取得するトレース情報の最大数をequate定義します。

equate定義した最大トレース情報取得数のラベル(標準提供の例では、ラベル名『TRC_CNT』)は、トレースバッファ情報テーブルおよびトレースバッファ領域の設定に使用します。

(2)TRC_BUF(トレースバッファ領域)

トレースバッファ領域を確保します。

確保したトレースバッファ領域の先頭アドレスには、トレースバッファ情報テーブル(INITRC)のトレースバッファアドレスに設定するため、ラベル(標準提供の例では、ラベル名『TRC_BUF』)を付けてください。

トレースバッファ領域のサイズは、以下のように設定します。

トレースバッファ領域のサイズ ＝ 16 ＋ TRC_CNT × 24
TRC_CNT：最大トレース情報取得数

(3)INITRC(トレースバッファ情報テーブル)

『INITRC』は、トレース機能を使用するための情報を登録するテーブルの先頭アドレスです。

トレースバッファ情報テーブルには、以下の設定を行ないます。

(a) TRACE BUFFER ADDRESS(トレースバッファアドレス)

トレース機能により取得する情報を格納するメモリ領域の先頭アドレスを設定します。

トレースバッファ領域の先頭アドレスに付けたラベル(標準提供の例では、ラベル名『TRC_BUF』)を設定してください。

トレース機能を使用しない場合は、０を設定してください。

(b) TRACE COUNT(最大トレース情報取得数)

最大トレース情報取得数には、トレース機能で取得する情報の最大数を設定します。

トレース機能を使用しない場合は、０を設定してください。

２．４．５　HI8-3Hシステム定義部

HI8-3Hシステム定義部は、HI8-3Hが使用する外部定義シンボル，定数の設定およびHI8-3Hのシステム作業領域を確保しています。

ユーザ定義部に設定した値から、外部定義シンボル，定数の設定およびHI8-3Hのシステム作業領域を確保します。

> HI8-3Hシステム定義部は、修正しないでください。
> HI8-3Hシステム定義部を修正した場合、動作は保証されません。

表２－６にHI8-3Hのシステム作業領域の詳細を示します。

表２－６　HI8-3Hのシステム作業領域の詳細

項番	領域	ラベル名	内　容	備考
1	SYSMT領域	_HI_SYSMT	システム管理テーブル	常に必要です
2	TCB 領域	_HI_TCB	タスク管理ブロック	
3	FLGCB領域	_HI_FLGCB	イベントフラグ管理ブロック	
4	SEMCB領域	_HI_SEMCB	セマフォ管理ブロック	
5	MBXCB領域	_HI_MBXCB	メイルボックス管理ブロック	
6	MPLCB領域	_HI_MPLCB	メモリプール管理ブロック	
7	TIMCB領域	_HI_TIMCB	タイマ管理ブロック	
8	OS用ｽﾀｯｸ領域	_HI_OS_SP	OSが使用するスタック領域	常に必要です
9	ﾀｲﾏ用ｽﾀｯｸ領域	_HI_TIM_SP	ﾀｲﾏﾊﾝﾄﾞﾗが使用するスタック領域	
１０	ﾄﾚｰｽ用ｽﾀｯｸ領域	_HI_TRC_SP	ﾄﾚｰｽ機能が使用するスタック領域	
１１	ﾄﾚｰｽ用領域	TBACB	ﾄﾚｰｽ機能が使用する領域	

HI8-3Hのシステム作業領域には、以下のサイズを確保します。（単位:バイト）

HI8-3Hのシステム作業領域＝OS作業領域＋OS用ｽﾀｯｸ領域＋ﾀｲﾏ用ｽﾀｯｸ領域＋ﾄﾚｰｽ用ｽﾀｯｸ領域
＝10＋（4×優先度定義数）　←― SYSMT領域
＋18×タスク定義数　←― TCB 領域
＋ 4×イベントフラグ定義数　←― FLGCB 領域
＋ 6×セマフォ定義数　←― SEMCB 領域
＋ 8×メイルボックス定義数　←― MBXCB 領域
＋ 6×メモリプール定義数　←― MPLCB 領域
＋10 *1　←― TIMCB 領域
＋OSスタックサイズ　←― OS用スタック領域
＋タイマスタックサイズ　←― タイマ用スタック領域
＋トレーススタックサイズ　←― トレース機能用スタック領域
＋8*2　←― トレース用領域

【注】*1　タイマスタックサイズ(ラベル名『TIMSTKSIZ』)を０にした場合、タイマ管理ブロック(TIMCB領域)は確保されません。

*2　トレーススタックサイズ(ラベル名『TRCSTKSIZ』)を０にした場合、トレース機能が使用する領域(ﾄﾚｰｽ用領域)は確保されません。

２．５　割込みベクタテーブルの作成

割込みベクタテーブルは、システム動作中に割込みが発生した場合、その割込みの種類に対応した処理に制御が移るよう、各割込みハンドラの先頭アドレスを登録しておくテーブルです。

割込みベクタテーブルは、以下のファイルが標準提供されていますので、これを参考にして作成してください。

- 『h3hvectr.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
- 『h3hvectn.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

２．５．１　割込みハンドラの登録

割込みハンドラを登録するには、割込みベクタテーブルの該当ベクタ番号に、割込みハンドラの先頭アドレスを設定してください。

(1)CPU初期化ルーチンの設定

HI8-3Hを動作させるには、CPU初期化ルーチン(ラベル名『H_3H_CPUINI』)をリセットベクタ(ベクタ番号 0)に登録しなければなりません。CPU初期化ルーチンは、リセット割込み後にCPUを初期化し、HI8-3Hのシステム起動処理に制御を移します。

(2)未定義割込みハンドラの設定

使用しないベクタ番号にはラベル名『H_3HINTxx 』(xxはベクタ番号)を登録してください。

ラベル名『H_3HINTxx 』は、HI8-3Hの未定義割込みハンドラの先頭アドレスです。

HI8-3Hの未定義割込みハンドラは、未定義割込み発生時の情報をスタックに設定し、システム異常終了ルーチンに移行します。

(3)システム予約

ベクタ番号 1，2は、システム予約ですので変更しないでください(ラベル名『H_SHARED』，『H_DBGHDL』)。

図２－１０に割込みハンドラの登録を示します。

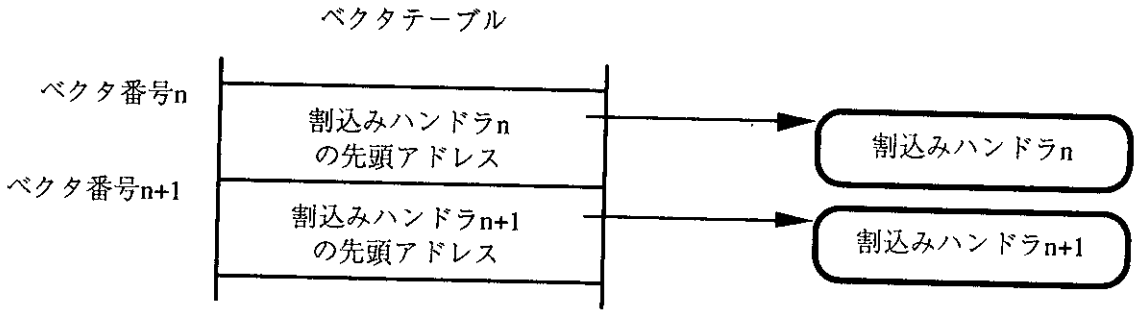


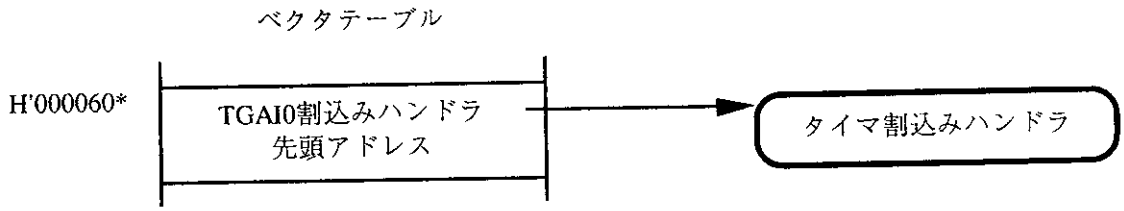
図２－１０　割込みハンドラの登録

(4)タイマ割込みハンドラの設定

wai_tsk, set_tim, get_timの各システムコールを使用する場合、タイマ割込みハンドラを登録しなければなりません。

標準提供のタイマ割込みハンドラを登録する場合は、先頭アドレス(ラベル名『H_3H_TIM』)をベクタ番号24に登録してください。

図２－１１にタイマ割込みハンドラの登録を示します。



【注】＊　H8/300Hノーマルモードの場合、H'0030番地になります。

図２－１１　タイマ割込みハンドラの登録

(5)他のＨ８／３００Ｈシリーズを使用する場合の設定

他のＨ８／３００Ｈシリーズを使用する場合、割込み要因の数が異なるため、以下の割込みベクタテーブルと未定義割込みハンドラの修正が必要です。

・割込みベクタテーブル(H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用(sampleaﾃﾞｨﾚｸﾄﾘ)
：ファイル名『h3hvectr.mar』)
・割込みベクタテーブル(H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用(samplenﾃﾞｨﾚｸﾄﾘ)：ファイル名『h3hvectn.mar』)
・未定義割込みハンドラ(H8/300Hﾉｰﾏﾙ(samplenﾃﾞｨﾚｸﾄﾘ)/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ(sampleaﾃﾞｨﾚｸﾄﾘ)共通
：ファイル名『h3hilint.mar』)

(a)割込み要因の数に応じて、割込みベクタテーブルにラベル名『H_3HINTxx』を追加してください。

(b)未定義割込みハンドラには、ベクタテーブルに設定したラベル名『H_3HINTxx』を付けた未定義割込みハンドラのプログラムを追加してください。未定義割込みハンドラのプログラムには、サブルーチン呼出しの命令（jsr　　@H_ilint）を設定してください。

割込み要因の詳細は、各CPUのハードウェアマニュアルを参照してください。

図２－１２に割込みベクタテーブル(H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用(sampleaﾃﾞｨﾚｸﾄﾘ)：ファイル名『h3hvectr.mar』)のコーディング例を示します。

図２－１２では、ベクタ番号 0にCPU初期化ルーチン，ベクタ番号24にタイマ割込みハンドラ，ベクタ番号52～54にコンソールドライバの割込みハンドラ(H8/3003のSCI0用:H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ)，ベクタ番号56～58にコンソールドライバの割込みハンドラ(H8/3003のSCI1用:H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ)を登録しています。

また、ベクタ番号 1，2はシステム予約、それ以外は、未定義割込みハンドラを登録しています。

```
;3333333333333333333333333333333333333333333333333333333333333333333333333333333333
;HHH                                                                             HHH
;HHH     H18-3H Version (uITRON V2.01/02)                                        HHH
;HHH     H18-3H vector table for H8/3003 Ver2.00                                 HHH
;HHH                                                                             HHH
;HHH     Copyright (c)  Hitachi, Ltd. 1993.                                      HHH
;HHH     Licensed Material of Hitachi, Ltd.                                      HHH
;HHH                                                                             HHH
;3333333333333333333333333333333333333333333333333333333333333333333333333333333333
        .program        h3hvectr
        .heading        "### h3hvectr.mar : for H8/3003 ###"
        .section        h3hvectr,code,locate=0
;
        .import         H_3HINT01,H_3HINT02,H_3HINT03,H_3HINT04,H_3HINT05
        .import         H_3HINT06,H_3HINT07,H_3HINT08,H_3HINT09,H_3HINT10
        .import         H_3HINT11,H_3HINT12,H_3HINT13,H_3HINT14,H_3HINT15
        .import         H_3HINT16,H_3HINT17,H_3HINT18,H_3HINT19,H_3HINT20
        .import         H_3HINT21,H_3HINT22,H_3HINT23,H_3HINT24,H_3HINT25
        .import         H_3HINT26,H_3HINT27,H_3HINT28,H_3HINT29,H_3HINT30
        .import         H_3HINT31,H_3HINT32,H_3HINT33,H_3HINT34,H_3HINT35
        .import         H_3HINT36,H_3HINT37,H_3HINT38,H_3HINT39,H_3HINT40
        .import         H_3HINT41,H_3HINT42,H_3HINT43,H_3HINT44,H_3HINT45
        .import         H_3HINT46,H_3HINT47,H_3HINT48,H_3HINT49,H_3HINT50
        .import         H_3HINT51,H_3HINT52,H_3HINT53,H_3HINT54,H_3HINT55
        .import         H_3HINT56,H_3HINT57,H_3HINT58,H_3HINT59,H_3HINT60
;
        .import         H_3H_CPUINI     ;: in 'cpuini'
        .import         H_3H_TIM        ;: in 'h3huser'
        .import         H_CNSHDL0ER     ;: in 'h3hcns_0'
        .import         H_CNSHDL0RX     ;: in 'h3hcns_0'
        .import         H_CNSHDL0TX     ;: in 'h3hcns_0'
        .import         H_CNSHDL1ER     ;: in 'h3hcns_1'
        .import         H_CNSHDL1RX     ;: in 'h3hcns_1'
        .import         H_CNSHDL1TX     ;: in 'h3hcns_1'
;
        .import         H_SHARED        ;: not change
        .import         H_DBGHDL        ;: not change
;
;*************************************************************************
;*specifications ;                                                       *
;*name       = h3hvectr : h8/3003 vector table for HI8-3H          ;     *
;*function   = 1. h8 interrupt handler address define for hi8      ;     *
;*           = 2. h8 exception handler address define for hi8      ;     *
;*           = 3. hi8 system standard support module               ;     *
;*           =    (1) reset    : "H_300H_INI"  for power on        ;     *
;*           = 4. hi8 system standard support module               ;     *
;*           =    (1) ITU0 tgai: "H_3H_TIM"    for system timer    ;     *
;*           = 5. hi8 system standard support i/o interrupt handler;     *
;*           =    (1) for console i/o driver [sci use]             ;     *
;*           =              (a) : "H_CNSHDL0ER" for receive error  ;     *
;*           =              (b) : "H_CNSHDL0RX" for receive  (input)  ;  *
;*           =              (c) : "H_CNSHDL0TX" for transmit (output) ;  *
;*           =              (d) : "H_CNSHDL1ER" for receive error  ;     *
;*           =              (e) : "H_CNSHDL1RX" for receive  (input)  ;  *
;*           =              (f) : "H_CNSHDL1TX" for transmit (output) ;  *
;*date       = 92/09/17                                            ;     *
;*author     = Hitachi, Ltd.                                       ;     *
;*history    = V1.00/V2.00                                         ;     *
;*attribute  = public                                              ;     *
;*class      = unit                                                ;     *
;*linkage    = h8 vector table top address = h'0000 for HI8-3H     ;     *
;*input      = none                                                ;     *
;*output     = none                                                ;     *
;*parameter  = @er7(B) : interrupt before "CCR"                    ;     *
;*           = @er7(L) : interrupt before "PC"(next mnemonic)      ;     *
;*end of specifications ;                                                *
;*************************************************************************
```

図２－１２　割込みベクタテーブルのコーディング例(1/2)

```
;
        .radix  d                       ;:xxxxx -> d'xxxxx
;
;-------.data.l < address >             ;: h8/3003 vector no. contents
        .data.l H_3H_CPUINI             ;: vector no.00 <reset>
        .data.l H_SHARED;H_3HINT01      ;: vector no.01 [reserve]
        .data.l H_DBGHDL;H_3HINT02      ;: vector no.02 [reserve]
        .data.l H_3HINT03               ;: vector no.03 [reserve]
        .data.l H_3HINT04               ;: vector no.04 [reserve]
        .data.l H_3HINT05               ;: vector no.05 [reserve]
        .data.l H_3HINT06               ;: vector no.06 [reserve]
        .data.l H_3HINT07               ;: vector no.07 <NMI          >
        .data.l H_3HINT08               ;: vector no.08 <TRAPA  #0    >
        .data.l H_3HINT09               ;: vector no.09 <TRAPA  #1    >
        .data.l H_3HINT10               ;: vector no.10 <TRAPA  #2    >
        .data.l H_3HINT11               ;: vector no.11 <TRAPA  #3    >
        .data.l H_3HINT12               ;: vector no.12 <IRQ0         >
        .data.l H_3HINT13               ;: vector no.13 <IRQ1         >
        .data.l H_3HINT14               ;: vector no.14 <IRQ2         >
        .data.l H_3HINT15               ;: vector no.15 <IRQ3         >
        .data.l H_3HINT16               ;: vector no.16 <IRQ4         >
        .data.l H_3HINT17               ;: vector no.17 <IRQ5         >
        .data.l H_3HINT18               ;: vector no.18 <IRQ6         >
        .data.l H_3HINT19               ;: vector no.19 <IRQ7         >
        .data.l H_3HINT20               ;: vector no.20 <WDT    wovi  >
        .data.l H_3HINT21               ;: vector no.21 <WDT    wovi  >
        .data.l H_3HINT22               ;: vector no.22 <WDT    wovi  >
        .data.l H_3HINT23               ;: vector no.23 <WDT    wovi  >
        .data.l H_3H_TIM;H_3HINT24      ;: vector no.24 <ITU0   tgai  >-option
        .data.l H_3HINT25               ;: vector no.25 <ITU0   tgbi  >
        .data.l H_3HINT26               ;: vector no.26 <ITU0   tovi  >
        .data.l H_3HINT27               ;: vector no.27 [reserve]
        .data.l H_3HINT28               ;: vector no.28 <ITU1   tgai  >
        .data.l H_3HINT29               ;: vector no.29 <ITU1   tgbi  >
        .data.l H_3HINT30               ;: vector no.30 <ITU1   tovi  >
        .data.l H_3HINT31               ;: vector no.31 [reserve]
        .data.l H_3HINT32               ;: vector no.32 <ITU2   tgai  >
        .data.l H_3HINT33               ;: vector no.33 <ITU2   tgbi  >
        .data.l H_3HINT34               ;: vector no.34 <ITU2   tovi  >
        .data.l H_3HINT35               ;: vector no.35 [reserve]
        .data.l H_3HINT36               ;: vector no.36 <ITU3   tgai  >
        .data.l H_3HINT37               ;: vector no.37 <ITU3   tgbi  >
        .data.l H_3HINT38               ;: vector no.38 <ITU3   tovi  >
        .data.l H_3HINT39               ;: vector no.39 [reserve]
        .data.l H_3HINT40               ;: vector no.40 <ITU4   tgai  >
        .data.l H_3HINT41               ;: vector no.41 <ITU4   tgbi  >
        .data.l H_3HINT42               ;: vector no.42 <ITU4   tovi  >
        .data.l H_3HINT43               ;: vector no.43 [reserve]
        .data.l H_3HINT44               ;: vector no.44 <DMAC03 dend0>
        .data.l H_3HINT45               ;: vector no.45 <DMAC03 dend1>
        .data.l H_3HINT46               ;: vector no.46 <DMAC03 dend2>
        .data.l H_3HINT47               ;: vector no.47 <DMAC03 dend3>
        .data.l H_3HINT48               ;: vector no.48 <DMAC47 dend4>
        .data.l H_3HINT49               ;: vector no.49 <DMAC47 dend5>
        .data.l H_3HINT50               ;: vector no.50 <DMAC47 dend6>
        .data.l H_3HINT51               ;: vector no.51 <DMAC47 dend7>
        .data.l H_CNSHDL0ER;H_3HINT52   ;: vector no.52 <SCI0   eri   >-option
        .data.l H_CNSHDL0RX;H_3HINT53   ;: vector no.53 <SCI0   rxi   >-option
        .data.l H_CNSHDL0TX;H_3HINT54   ;: vector no.54 <SCI0   txi   >-option
        .data.l H_3HINT55               ;: vector no.55 <SCI0   tend  >
        .data.l H_CNSHDL1ER;H_3HINT56   ;: vector no.56 <SCI1   eri   >-option
        .data.l H_CNSHDL1RX;H_3HINT57   ;: vector no.57 <SCI1   rxi   >-option
        .data.l H_CNSHDL1TX;H_3HINT58   ;: vector no.58 <SCI1   txi   >-option
        .data.l H_3HINT59               ;: vector no.59 <SCI1   tend  >
        .data.l H_3HINT60               ;: vector no.60 <ADC    adi   >
;
;*************************************;
        .end; of H3HVECTR.MAR
```

図２－１２　割込みベクタテーブルのコーディング例(2/2)

２．６　システム初期化ハンドラの登録

システム初期化ハンドラを登録する場合、システム初期化ハンドラのプログラムの先頭にラベル名『_HIPRG_SYSINI』を付け、export宣言してください。

システム初期化ハンドラを未登録にする場合、ラベル名『_HIPRG_SYSINI』に０をequate定義し、export宣言してください。

標準提供のシステム初期化ハンドラは、次のファイル名に含まれています。

・『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
・『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

標準提供のシステム初期化ハンドラは、未登録です。

２．７　タイマドライバの登録

時間管理機能を使用する場合、タイマドライバを登録してください。タイマドライバはタイマ初期設定ルーチンとタイマ割込みハンドラから構成されています。標準提供のタイマ初期設定ルーチンおよびタイマ割込みハンドラは、以下のファイルに含まれています。

・『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
・『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

タイマドライバを使用しない場合は、割込みベクタテーブル、セットアップテーブルへの登録および標準提供のタイマドライバの削除が必要です。

(1)タイマ初期設定ルーチンの登録

タイマ初期設定ルーチンを登録する場合、タイマ初期設定ルーチンのプログラムの先頭にラベル名『_HIPRG_TIMINI』を付け、export宣言してください。

タイマ初期設定ルーチンを未登録にする場合、ラベル名『_HIPRG_TIMINI』に０をequate定義し、export宣言してください。

(2)タイマ割込みハンドラの登録

タイマ割込みハンドラの先頭アドレスを割込みベクタテーブルに登録します。

詳しい内容は、「２．５．１　割込みハンドラの登録」を参照してください。

(3)タイマドライバを未登録にする方法

タイマドライバを使用しない場合、以下の設定を行なってください。

(a)割込みベクタテーブル

タイマ割込みハンドラの先頭アドレス(ラベル名『H_3H_TIM』)の外部参照宣言(import)を削除し、ベクタ番号２４に未定義割込みハンドラ(ラベル名『H_3HINT24』)を登録します。

(b)セットアップテーブル

タイマスタックサイズ(ラベル名『TIMSTKSIZ』)を０にします。

(c)タイマドライバ(タイマ初期設定ルーチン、タイマ割込みハンドラ)

①タイマ初期設定ルーチン(ラベル名『_HIPRG_TIMINI』) のプログラムを削除し、ラベル名『_HIPRG_TIMINI』に０をequate定義します。

②タイマ割込みハンドラ(ラベル名『H_3H_TIM』)のプログラムを削除します。

③ＯＳのタイマ割込み処理(ラベル名『H_timsys』)の外部参照シンボルの宣言(import)を削除します。

２．８　システム異常終了ルーチンの登録

システム異常終了ルーチンは、必ず登録してください。

システム異常終了ルーチンを登録する場合、プログラムの先頭にラベル名『_HIPRG_ABNOML』を付け、export宣言してください。

標準提供のシステム異常終了ルーチンは、次のファイルに含まれています。

- 『h3huser.mar』：H8/300Hアドバンストモード用(sampleaﾃﾞｨﾚｸﾄﾘ)
- 『h3husern.mar』：H8/300Hノーマルモード用(samplenﾃﾞｨﾚｸﾄﾘ)

ラベル名『_HIPRG_ABNOML』に０または奇数アドレスをequate定義した場合、動作は保証されません。

２．９　実行形式プログラムの作成(H8/300Hノーマルモード)

作成したソースプログラムをもとに、実行形式プログラムを生成します。
図２－１３にシステム結合の概要を示します。

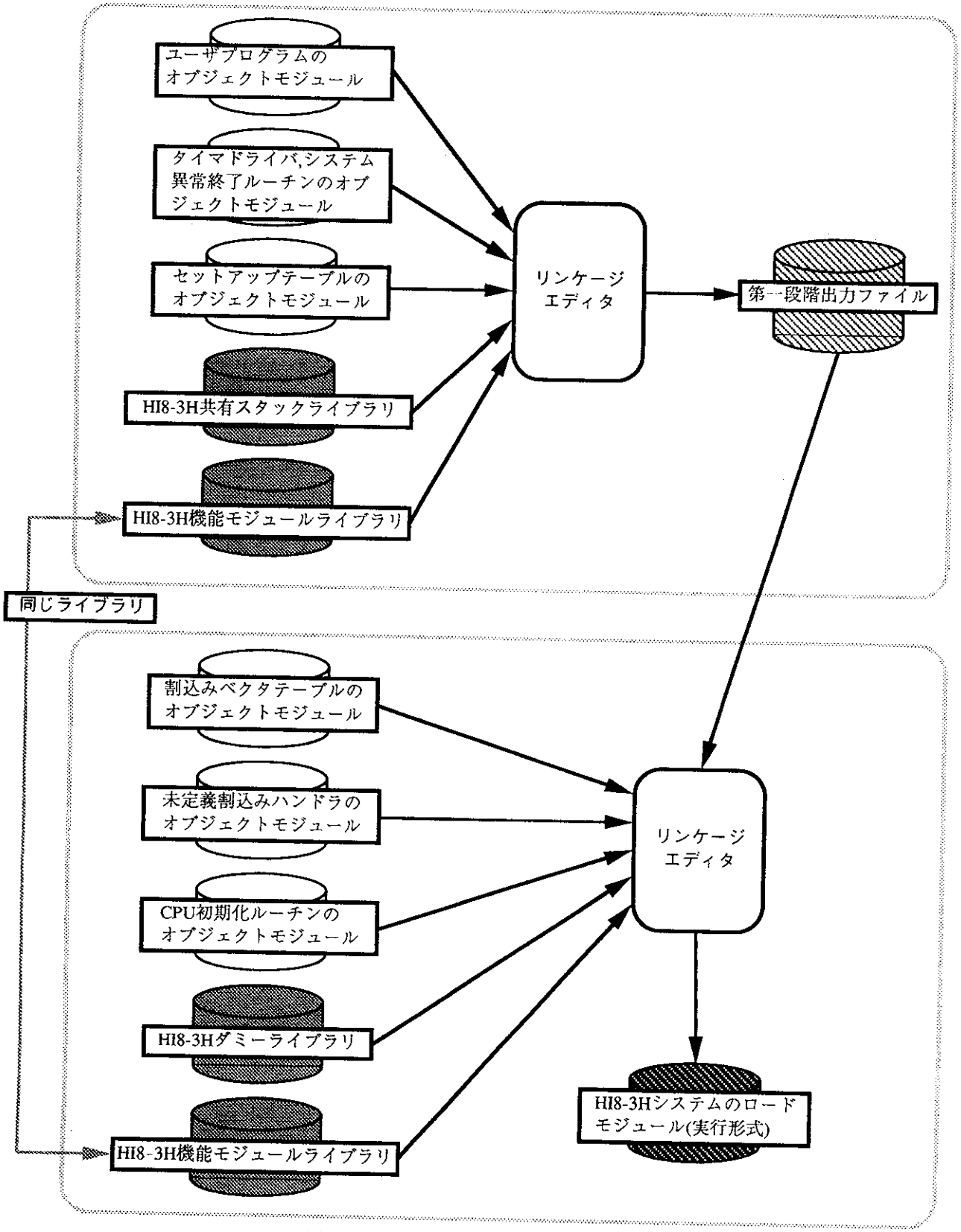


図２－１３　システム結合の概要

２．９．１　オブジェクトモジュールの作成(H8/300Hノーマルモード)

作成するオブジェクトモジュールには以下のものがあります。

・ユーザプログラム（タスク、割込みハンドラ、システム初期化ハンドラ）
・タイマドライバ、システム異常終了ルーチン
・セットアップテーブル
・割込みベクタテーブル、未定義割込みハンドラ
・CPU初期化ルーチン

標準提供のソースプログラムファイルを使用する場合、メイクファイルを利用してアセンブル／コンパイルすることができます。メイクファイルの実行方法については、「２．１１　メイクファイルの実行」を参照してください。

(1)　ユーザプログラムのアセンブル

ユーザプログラムのソースファイルをアセンブルまたはコンパイルし、オブジェクトモジュールを生成します。

ユーザプログラムをアセンブルするには次のコマンドを入力します。

アセンブルのコマンドラインオプション(CPU種別)には、ノーマルモードを指定します。

```
% asm38 △〈ユーザプログラムファイル名〉△ -cpu=300hn［△-debug］(RET)
```

ユーザプログラムをコンパイルするには次のコマンドを入力します。コンパイラのオプション(オブジェクト種類)には、ノーマルモードを指定します。

```
% ch38 △ -cpu=300hn［△-debug］△〈ユーザプログラムファイル名〉(RET)
```

(2) タイマドライバ、システム異常終了ルーチンのアセンブル

タイマドライバ、システム異常終了ルーチンのプログラムをアセンブルするには次のコマンドを入力します。アセンブルのコマンドラインオプション(CPU種別)には、ノーマルモードを指定します。

```
% asm38 △〈プログラムファイル名〉△ -cpu=300hn［△-debug］(RET)
```

タイマドライバ、システム異常終了ルーチンのプログラムをコンパイルするには次のコマンドを入力します。コンパイラのオプション(オブジェクト種類)には、ノーマルモードを指定します。

```
% ch38 △ -cpu=300hn［△-debug］△〈プログラムファイル名〉(RET)
```

標準提供のタイマドライバ、システム異常終了ルーチンのファイルを以下に示します。

・ファイル名『h3husern.mar』：H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用ﾀｲﾏﾄﾞﾗｲﾊﾞ、ｼｽﾃﾑ異常終了ﾙｰﾁﾝ
(samplenﾃﾞｨﾚｸﾄﾘ)

(3) セットアップテーブルのアセンブル

構築するシステムのセットアップテーブルをアセンブルし、オブジェクトモジュールを生成します。セットアップテーブルをアセンブルするには次のコマンドを入力します。
アセンブルのコマンドラインオプション(CPU種別)には、ノーマルモードを指定します。

```
% asm38 △ h3hsetup.mar △ -cpu=300hn [△-debug] (RET)
```

(4) 割込みベクタテーブル、未定義割込みハンドラのアセンブル

構築するシステムの割込みベクタテーブル、未定義割込みハンドラをアセンブルし、オブジェクトモジュールを生成します。割込みベクタテーブルファイルをアセンブルするには次のコマンドを入力します。アセンブルのコマンドラインオプション(CPU種別)には、ノーマルモードを指定します。

```
% asm38 △ <割込みベクタテーブルのファイル名> △ -cpu=300hn [△-debug] (RET)
% asm38 △ <未定義割込みハンドラのファイル名> △ -cpu=300hn [△-debug] (RET)
```

標準提供の割込みベクタテーブル、未定義割込みハンドラのファイルを以下に示します。

・ファイル名『h3hvectn.mar』：H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用割込みﾍﾞｸﾀﾃｰﾌﾞﾙ(samplenﾃﾞｨﾚｸﾄﾘ)
・ファイル名『h3hilint.mar』：H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用未定義割込みﾊﾝﾄﾞﾗ(samplenﾃﾞｨﾚｸﾄﾘ)

(5) CPU初期化ルーチンのアセンブル

CPU初期化ルーチンのプログラムをアセンブルするには次のコマンドを入力します。アセンブルのコマンドラインオプション(CPU種別)には、ノーマルモードを指定します。

```
% asm38 △ <CPU初期化ルーチンのファイル名> △ -cpu=300hn [△-debug] (RET)
```

標準提供のCPU初期化ルーチンのファイルを以下に示します。

・ファイル名『cpuinin.mar』：H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用CPU初期化ルーチン(samplenﾃﾞｨﾚｸﾄﾘ)

H8/300シリーズのクロスアセンブラの詳しい説明は、「Ｈ８／３００シリーズ　クロスアセンブラ　ユーザーズマニュアル」を参照してください。また、H8/300シリーズのＣコンパイラの詳しい説明は、「Ｈ８／３００シリーズ　Ｃコンパイラ　ユーザーズマニュアル」を参照してください。

２．９．２　オブジェクトモジュールの結合(H8/300Hノーマルモード)

構築するシステムに必要なオブジェクトモジュールやライブラリファイルをリンケージエディタにより結合し、HI8-3Hのロードモジュールを作成します。

リンケージエディタでユーザプログラムとカーネルのライブラリを結合するだけで、最適な実行形式ロードモジュールが生成されます。

HI8-3Hのライブラリは、５つに分かれています。

表２－７にHI8-3Hライブラリの機能を示します。

表２－７　HI8-3Hライブラリの機能

項番	機能	ファイル名	備考
1	機能モジュールライブラリ	h3hkmdl.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし(knlnﾃﾞｨﾚｸﾄﾘ)
		h3hkmdlc.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり(knlnﾃﾞｨﾚｸﾄﾘ)
2	共有スタックライブラリ (共有ｽﾀｯｸ機能)	h3hkstk.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし(knlnﾃﾞｨﾚｸﾄﾘ)
		h3hkstkc.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり(knlnﾃﾞｨﾚｸﾄﾘ)
3	ダミーライブラリ	h3hkdmy.lib	(knlnﾃﾞｨﾚｸﾄﾘ)

ＨＩ８－３Ｈのシステム結合は、以下の４つの項目にしたがって行なってください。

①共有スタック機能の使用／未使用の選択

②システムコールのパラメータチェック機能あり／なしの選択

③システム結合の順序

④メモリ配置

(1)共有スタック機能の使用／未使用の選択

『h3hkstk.lib』または『h3hkstkc.lib』（パラメータチェック機能あり）を結合すると、共有スタック機能が使用できます。

これらのライブラリを結合しないと、共有スタック機能が使用できません。

(2)システムコールのパラメータチェック機能あり／なしの選択

『h3hkmdlc.lib』,『h3hkstkc.lib』（共有スタック機能使用時のみ）を結合すると、パラメータチェック機能が組み込まれたシステムコールを使用できます。

『h3hkmdl.lib』,『h3hkstk.lib』（共有スタック機能使用時のみ）を結合すると、パラメータチェック機能が組み込まれていないシステムコールを使用できます。

(3)システム結合の順序

H18-3Hのシステム結合は、必ず２つの段階に分けて行なってください。

システム結合を２段階に分けて行なっていない場合、H18-3Hが正常に動作しない場合があります。

第1段階では、必ず全てのユーザプログラム、タイマドライバ、システム異常終了ルーチン、セットアップテーブルおよびカーネルの機能モジュールライブラリを結合し、リロケータブルロードモジュールを出力してください。

第2段階では、第1段階で出力したリロケータブルロードモジュール、割込みベクタテーブル、未定義割込みハンドラ、CPU初期化ルーチンおよびカーネルのダミーライブラリ、機能モジュールライブラリを結合してください。

以下にH18-3Hのシステム結合時に入力するファイルの順序を示します。

(a) 第１段階の入力ファイル

①ユーザプログラムのオブジェクトモジュールファイル

②タイマドライバのオブジェクトモジュールファイル

システム異常終了ルーチンのオブジェクトモジュールファイル

③セットアップテーブルのオブジェクトモジュールファイル

④共有スタックライブラリファイル(共有スタック機能を使用する場合のみ、必ず、⑤機能モジュールライブラリより前に入力してください)

⑤機能モジュールライブラリファイル

(b) 第２段階の入力ファイル

①第１段階の出力ファイル（リロケータブルロードモジュール）

②割込みベクタテーブルのオブジェクトモジュールファイル

未定義割込みハンドラのオブジェクトモジュールファイル

③CPU初期化ルーチンのオブジェクトモジュールファイル

④ダミーライブラリファイル(必ず、⑤機能モジュールライブラリより前に入力してください)

⑤機能モジュールライブラリファイル

(4)メモリ配置

HI8-3Hのプログラムおよび作業領域をメモリに配置するための注意事項を以下に示します。

以下に示すセクション以外のセクションについては、メモリ配置に対する注意事項はありません。

(a) カーネル(セクション名『hi8_3h』)

カーネルのプログラムは、偶数番地から配置してください。

(b) HI8-3Hのシステム作業領域(セクション名『hi8_3h_ram』)

システム作業領域は、偶数番地から配置してください。

(c) セットアップテーブル(セクション名『h3hsetup』)

セットアップテーブルは、偶数番地から配置してください。

(d) 割込みベクタテーブル(セクション名『h3hvectr』)

割込みベクタテーブルは、H'0000番地から配置してください。標準提供の割込みベクタテーブルは、自動的にH'0000番地に配置します。

2．9．3　リンケージエディタの実行(H8/300Hノーマルモード)

リンケージエディタを使用して、作成したオブジェクトモジュールと機能モジュールライブラリを結合してください。リンケージエディタの実行には、サブコマンドファイルを利用してください。

リンケージエディタの実行には、次のコマンドを入力します。

```
% lnk △ -sub=〈ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ名〉(RET)
```

標準提供のサブコマンドファイルを参考に、結合したいシステムのプログラムファイル、セクションのメモリ配置などを修正／追加して、HI8-3Hのシステム結合を行なってください。

> 第一段階のシステム結合時、未定義の外部シンボル参照のエラーメッセージが表示されますが、外部シンボルが"H_DBGHDL"の場合は問題ありません。

標準提供のソースプログラムファイル、サブコマンドファイルを使用する場合、メイクファイルを利用してアセンブルから結合まで実行することができます。メイクファイルの実行方法については、「2．11　メイクファイルの実行」を参照してください。

表２－８に標準提供のサブコマンドファイル一覧を示します。

表２－８　サブコマンドファイル一覧

項番	ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ名	内　容
1	h3hlnkn.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし，共有ｽﾀｯｸ機能なしのカーネルを結合する
2	h3hlnkcn.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり，共有ｽﾀｯｸ機能なしのカーネルを結合する
3	h3hlnksn.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし，共有ｽﾀｯｸ機能ありのカーネルを結合する
4	h3hlncsn.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり，共有ｽﾀｯｸ機能ありのカーネルを結合する

図２－１４に"パラメータチェックなし，共有スタック機能なし"のサブコマンドファイル（ファイル名『h3hlnkn.sub』:samplenﾃﾞｨﾚｸﾄﾘ)を示します。

```
input   h3hsetup,h3husern,〈ユーザﾌﾟﾛｸﾞﾗﾑのｵﾌﾞｼﾞｪｸﾄﾓｼﾞｭｰﾙ〉                    <-- ①
library ../knln/h3hkmdl,../cifn/h3hcif,../cnsn/h3hcns,c38hn                  <-- ②
output  h3htmp.rel                                                           <-- ③
form    r                                                                    <-- ④
debug
end

input   h3hvectn,h3hilint,cpuinin,h3htmp.rel                                 <-- ⑤
library ../knln/h3hkdmy,../knln/h3hkmdl                                      <-- ⑥
output  h3hprgn.abs                                                          <-- ⑦
start   hi8_3h(80),&                                                         <-- ⑧
        h3hsetup,h3huser,h3hilint,h3hcns,〈ユーザﾌﾟﾛｸﾞﾗﾑのｾｸｼｮﾝ名〉,h3hc(2000),&
        hi8_3h_ram,h3hstack,h3hcns_ram,h3hmpl,h3htrc(0f710)
print   h3hprgn.map                                                          <-- ⑨
debug
exit
```

図２－１４　サブコマンドファイル『h3hlnkn.sub』

（解説）
第１段階：
①h3hsetup.obj，h3husern.objを入力します。（省略不可）
　作成したユーザプログラムのオブジェクトモジュールを入力します
②h3hkmdl.lib，h3hcif.lib，h3hcns.libを入力します。（省略不可）
　Ｃコンパイラの標準ライブラリを使用する場合、c38hn.libを入力します。
③出力ファイル名を h3htmp.relとします。
④出力ロードモジュールファイルをリロケータブル形式にします。
第２段階：
⑤h3hvectn.obj，h3hilint.obj，cpuinin.obj，h3htmp.relを入力します。（省略不可）
⑥h3hkdmy.lib，h3hkmdl.libを入力します。（省略不可）
⑦出力ファイル名を h3hprgn.absとします。
⑧カーネル(hi8_3h)をH'80番地から配置します。
　セットアップテーブル(h3hsetup)，システム初期化ハンドラ等のプログラム(h3huser)，未定義割込み処理(h3hilint)，コンソールドライバ(h3hcns)をH'2000番地から配置します。
　システム作業領域(hi8_3h_ram)，タスクスタック領域(h3hstack)，コンソールドライバ作業領域(h3hcns_ram)，メモリプール領域(h3hmpl)，トレースバッファ領域(h3htrc)をH'F710番地から配置します。
　ユーザプログラムをＣ言語で作成した場合、ユーザプログラムとＣ言語インタフェースライブラリのセクション名(h3hc)を設定します。
⑨このシステムのリンケージリストをファイルh3hprgn.mapに出力します。

リンケージエディタの詳しい説明は「Ｈシリーズ　リンケージエディタ　ユーザーズマニュアル」を参照してください。

表２－９にHI8-3Hが使用しているセクション名を示します。

表２－９　HI8-3Hのセクション名

項番	セクション名	プログラム，作業領域
1	hi8_3h	カーネル
2	h3hsetup	セットアップテーブル
3	h3huser	システム初期化ハンドラ，タイマ初期設定ルーチン，タイマ割込みハンドラ，システム異常終了ルーチン，ＣＰＵ初期化ルーチン
4	hi8_3h_ram	HI8-3Hのシステム作業領域
5	h3hstack	タスクスタック領域
6	h3hmpl	メモリプール領域
7	h3hc	Ｃ言語インタフェースライブラリ（Ｃ言語でシステムコールを使用する場合）
8	h3hcns	コンソールドライバ
9	h3hcns_ram	コンソールドライバ作業領域
１０	h3hilint	未定義割込みハンドラ
１１	h3htrc	トレースバッファ領域

２．１０　実行形式プログラムの作成(H8/300Hアドバンストモード)

作成したソースプログラムをもとに、実行形式プログラムを生成します。
図２－１５にシステム結合の概要を示します。

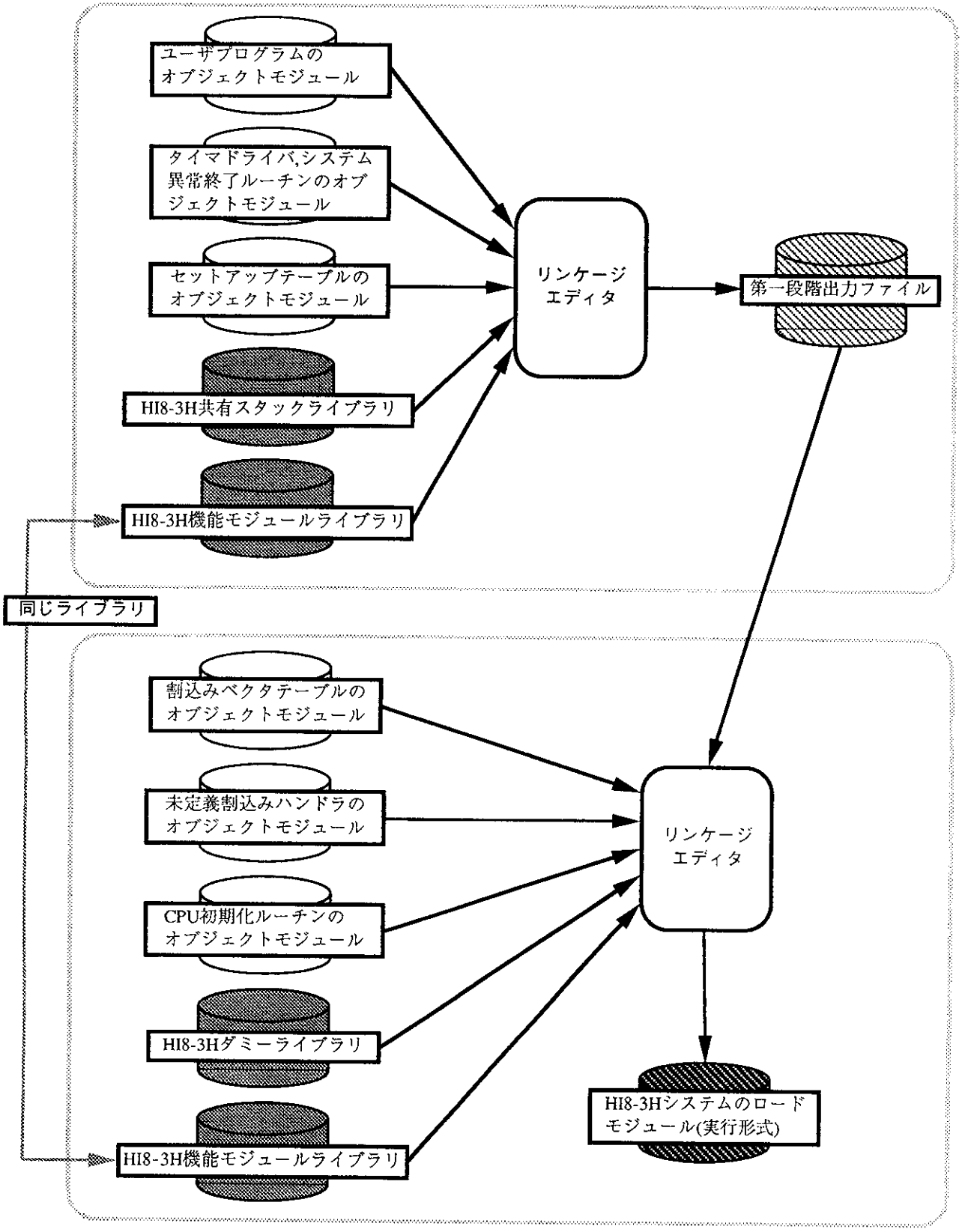


図２－１５　システム結合の概要

２．１０．１　オブジェクトモジュールの作成(H8/300Hアドバンストモード)

作成するオブジェクトモジュールには以下のものがあります。

・ユーザプログラム（タスク、割込みハンドラ、システム初期化ハンドラ）
・タイマドライバ、システム異常終了ルーチン
・セットアップテーブル
・割込みベクタテーブル、未定義割込みハンドラ
・CPU初期化ルーチン

標準提供のソースプログラムファイルを使用する場合、メイクファイルを利用してアセンブル／コンパイルすることができます。メイクファイルの実行方法については、「２．１１　メイクファイルの実行」を参照してください。

(1)　ユーザプログラムのアセンブル

ユーザプログラムのソースファイルをアセンブルまたはコンパイルし、オブジェクトモジュールを生成します。

ユーザプログラムをアセンブルするには次のコマンドを入力します。

アセンブルのコマンドラインオプション(CPU種別)には、アドバンストモードを指定します。

% asm38 △〈ユーザプログラムファイル名〉△ -cpu=300ha ［△-debug］(RET)

ユーザプログラムをコンパイルするには次のコマンドを入力します。コンパイラのオプション(オブジェクト種類)には、アドバンストモードを指定します。

% ch38 △ -cpu=300ha ［△-debug］ △〈ユーザプログラムファイル名〉(RET)

(2) タイマドライバ、システム異常終了ルーチンのアセンブル

タイマドライバ、システム異常終了ルーチンのプログラムをアセンブルするには次のコマンドを入力します。アセンブルのコマンドラインオプション(CPU種別)には、アドバンストモードを指定します。

% asm38 △〈プログラムファイル名〉△ -cpu=300ha ［△-debug］(RET)

タイマドライバ、システム異常終了ルーチンのプログラムをコンパイルするには次のコマンドを入力します。コンパイラのオプション(オブジェクト種類)には、アドバンストモードを指定します。

% ch38△ -cpu=300ha ［△-debug］ △〈プログラムファイル名〉(RET)

標準提供のタイマドライバ、システム異常終了ルーチンのファイルを以下に示します。

・ファイル名『h3huser.mar』：H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用ﾀｲﾏﾄﾞﾗｲﾊﾞ、ｼｽﾃﾑ異常終了ﾙｰﾁﾝ
（sampleaﾃﾞｨﾚｸﾄﾘ）

(3) セットアップテーブルのアセンブル

構築するシステムのセットアップテーブルをアセンブルし、オブジェクトモジュールを生成します。セットアップテーブルをアセンブルするには次のコマンドを入力します。
アセンブルのコマンドラインオプション(CPU種別)には、アドバンストモードを指定します。

% asm38 △ h3hsetup.mar △ -cpu=300ha [△-debug] (RET)

(4) 割込みベクタテーブル、未定義割込みハンドラのアセンブル

構築するシステムの割込みベクタテーブル、未定義割込みハンドラをアセンブルし、オブジェクトモジュールを生成します。割込みベクタテーブルファイルをアセンブルするには次のコマンドを入力します。アセンブルのコマンドラインオプション(CPU種別)には、アドバンストモードを指定します。

% asm38 △ 〈割込みベクタテーブルのファイル名〉 △ -cpu=300ha [△-debug] (RET)
% asm38 △ 〈未定義割込みハンドラのファイル名〉 △ -cpu=300ha [△-debug] (RET)

標準提供の割込みベクタテーブル、未定義割込みハンドラのファイルを以下に示します。

・ファイル名『h3hvectr.mar』 : H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用割込みﾍﾞｸﾀﾃｰﾌﾞﾙ(sampleaﾃﾞｨﾚｸﾄﾘ)
・ファイル名『h3hilint.mar』 : H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用未定義割込みﾊﾝﾄﾞﾗ(sampleaﾃﾞｨﾚｸﾄﾘ)

(5) CPU初期化ルーチンのアセンブル

CPU初期化ルーチンのプログラムをアセンブルするには次のコマンドを入力します。アセンブルのコマンドラインオプション(CPU種別)には、アドバンストモードを指定します。

% asm38 △ 〈CPU初期化ルーチンのファイル名〉 △ -cpu=300ha [△-debug] (RET)

標準提供のCPU初期化ルーチンのファイルを以下に示します。

・ファイル名『cpuini.mar』 : H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用CPU初期化ルーチン(sampleaﾃﾞｨﾚｸﾄﾘ)

H8/300シリーズのクロスアセンブラの詳しい説明は、「Ｈ８／３００シリーズ　クロスアセンブラ　ユーザーズマニュアル」を参照してください。また、H8/300シリーズのＣコンパイラの詳しい説明は、「Ｈ８／３００シリーズ　Ｃコンパイラ　ユーザーズマニュアル」を参照してください。

２．１０．２　オブジェクトモジュールの結合(H8/300Hアドバンストモード)

構築するシステムに必要なオブジェクトモジュールやライブラリファイルをリンケージエディタにより結合し、HI8-3Hのロードモジュールを作成します。

リンケージエディタでユーザプログラムとカーネルのライブラリを結合するだけで、最適な実行形式ロードモジュールが生成されます。

HI8-3Hのライブラリは、５つに分かれています。

表２－１０にHI8-3Hライブラリの機能を示します。

表２－１０　HI8-3Hライブラリの機能

項番	機能	ファイル名	備考
1	機能モジュールライブラリ	h3hkmdl.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし(knlaﾃﾞｨﾚｸﾄﾘ)
		h3hkmdlc.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり(knlaﾃﾞｨﾚｸﾄﾘ)
2	共有スタックライブラリ（共有ｽﾀｯｸ機能）	h3hkstk.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし(knlaﾃﾞｨﾚｸﾄﾘ)
		h3hkstkc.lib	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり(knlaﾃﾞｨﾚｸﾄﾘ)
3	ダミーライブラリ	h3hkdmy.lib	(knlaﾃﾞｨﾚｸﾄﾘ)

ＨＩ８－３Ｈのシステム結合は、以下の４つの項目にしたがって行なってください。

①共有スタック機能の使用／未使用の選択

②システムコールのパラメータチェック機能あり／なしの選択

③システム結合の順序

④メモリ配置

(1)共有スタック機能の使用／未使用の選択

『h3hkstk.lib』または『h3hkstkc.lib』（パラメータチェック機能あり）を結合すると、共有スタック機能が使用できます。

これらのライブラリを結合しないと、共有スタック機能が使用できません。

(2)システムコールのパラメータチェック機能あり／なしの選択

『h3hkmdlc.lib』.『h3hkstkc.lib』（共有スタック機能のライブラリ）を結合すると、パラメータチェック機能が組み込まれたシステムコールを使用できます。

『h3hkmdl.lib』.『h3hkstk.lib』（共有スタック機能のライブラリ）を結合すると、パラメータチェック機能が組み込まれていないシステムコールを使用できます。

(3)システム結合の順序

HI8-3Hのシステム結合は、必ず２つの段階に分けて行なってください。

システム結合を２段階に分けて行なっていない場合、HI8-3Hが正常に動作しない場合があります。

第1段階では、必ず全てのユーザプログラム、タイマドライバ、システム異常終了ルーチン、セットアップテーブルおよびカーネルの機能モジュールライブラリを結合し、リロケータブルロードモジュールを出力してください。

第2段階では、第1段階で出力したリロケータブルロードモジュール、割込みベクタテーブル、未定義割込みハンドラ、CPU初期化ルーチンおよびカーネルのダミーライブラリ、機能モジュールライブラリを結合してください。

以下にHI8-3Hのシステム結合時に入力するファイルの順序を示します。

(a) 第１段階の入力ファイル

①ユーザプログラムのオブジェクトモジュールファイル

②タイマドライバのオブジェクトモジュールファイル

システム異常終了ルーチンのオブジェクトモジュールファイル

③セットアップテーブルのオブジェクトモジュールファイル

④共有スタックライブラリファイル(共有スタック機能を使用する場合のみ、必ず、⑤機能モジュールライブラリより前に入力してください)

⑤機能モジュールライブラリファイル

(b) 第２段階の入力ファイル

①第１段階の出力ファイル（リロケータブルロードモジュール）

②割込みベクタテーブルのオブジェクトモジュールファイル

未定義割込みハンドラのオブジェクトモジュールファイル

③CPU初期化ルーチンのオブジェクトモジュールファイル

④ダミーライブラリファイル(必ず、⑤機能モジュールライブラリより前に入力してください)

⑤機能モジュールライブラリファイル

(4)メモリ配置

HI8-3Hのプログラムおよび作業領域をメモリに配置するための注意事項を以下に示します。

以下に示すセクション以外のセクションについては、メモリ配置に対する注意事項はありません。

(a) カーネル(セクション名『hi8_3h』)

カーネルのプログラムは、H'xx0000～H'xxFFFF番地の間に配置してください。配置する番地の上位アドレス"xx"は同一にしてください。また、偶数番地から配置してください。

(b) HI8-3Hのシステム作業領域(セクション名『hi8_3h_ram』)

システム作業領域は、H'xx0000～H'xxFFFF番地の間に配置してください。配置する番地の上位アドレス"xx"は同一にしてください。また、偶数番地から配置してください。

共有スタック機能を使用している場合、タスクスタック領域のセクション『h3hstack』と『hi8_3h_ram』のセクションは、同じH'xx0000～H'xxFFFF番地の範囲に配置してください。

(c) セットアップテーブル(セクション名『h3hsetup』)

セットアップテーブルは、H'xx0000～H'xxFFFF番地の間に配置してください。配置する番地の上位アドレス"xx"は同一にしてください。また、偶数番地から配置してください。

(d) 割込みベクタテーブル(セクション名『h3hvectr』)

割込みベクタテーブルは、H'000000番地から配置してください。標準提供の割込みベクタテーブルは、自動的にH'000000番地に配置します。

２．１０．３　リンケージエディタの実行(H8/300Hアドバンストモード)

リンケージエディタを使用して、作成したオブジェクトモジュールと機能モジュールライブラリを結合してください。リンケージエディタの実行には、サブコマンドファイルを利用してください。

リンケージエディタの実行には、次のコマンドを入力します。

```
% lnk△ -sub=<ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ名>(RET)
```

標準提供のサブコマンドファイルを参考に、結合したいシステムのプログラムファイル、セクションのメモリ配置などを修正／追加して、HI8-3Hのシステム結合を行なってください。

> 第一段階のシステム結合時、未定義の外部シンボル参照のエラーメッセージが表示されますが、外部シンボルが"H_DBGHDL"の場合は問題ありません。

標準提供のソースプログラムファイル、サブコマンドファイルを使用する場合、メイクファイルを利用してアセンブルから結合まで実行することができます。メイクファイルの実行方法については、「２．１１　メイクファイルの実行」を参照してください。

表２－１１に標準提供のサブコマンドファイル一覧を示します。

表２－１１　サブコマンドファイル一覧

項番	ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ名	内　容
1	h3hlnk.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし，共有ｽﾀｯｸ機能なしのカーネルを結合する
2	h3hlnkc.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり，共有ｽﾀｯｸ機能なしのカーネルを結合する
3	h3hlnks.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能なし，共有ｽﾀｯｸ機能ありのカーネルを結合する
4	h3hlnkcs.sub	ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能あり，共有ｽﾀｯｸ機能ありのカーネルを結合する

図２－１６に"パラメータチェックなし，共有スタック機能なし"のサブコマンドファイル（ファイル名『h3hlnk.sub』:sampleaﾃﾞｨﾚｸﾄﾘ）を示します。

```
input    h3hsetup,h3huser,〈ユーザﾌﾟﾛｸﾞﾗﾑのｵﾌﾞｼﾞｪｸﾄﾓｼﾞｭｰﾙ〉          <-- ①
library  ../knla/h3hkmdl,../cifa/h3hcif,../cnsa/h3hcns,c38ha      <-- ②
output   h3htmp.rel                                               <-- ③
form     r                                                        <-- ④
debug
end

input    h3hvectr,h3hilint,cpuini,h3htmp.rel                      <-- ⑤
library  ../knla/h3hkdmy,../knla/h3hkmdl                          <-- ⑥
output   h3hprg.abs                                               <-- ⑦
start    hi8_3h,h3hsetup,h3huser,〈ユーザﾌﾟﾛｸﾞﾗﾑのｾｸｼｮﾝ名〉,h3hc,&    <-- ⑧
         hi8_3h_ram,h3hstack,h3hmpl(100)
print    h3hprg.map                                               <-- ⑨
debug
exit
```

図２－１６　サブコマンドファイル『h3hlnk.sub』

（解説）
第１段階：
①h3hsetup.obj，h3huser.objを入力します。（省略不可）
　作成したユーザプログラムのオブジェクトモジュールを入力します
②h3hkmdl.lib，h3hcif.lib，h3hcns.libを入力します。（省略不可）
　Ｃコンパイラの標準ライブラリを使用する場合、c38ha.libを入力します。
③出力ファイル名を h3htmp.relとします。
④出力ロードモジュールファイルをリロケータブル形式にします。
第２段階：
⑤h3hvectr.obj，h3hilint.obj，cpuini.obj，h3htmp.relを入力します。（省略不可）
⑥h3hkdmy.lib，h3hkmdl.libを入力します。（省略不可）
⑦出力ファイル名を h3hprg.absとします。
⑧カーネル(hi8_3h)を先頭に，セットアップテーブル(h3hsetup)，システム初期化ハンドラ等のプログラム(h3huser)，ユーザプログラム，OS作業領域(hi8_3h_ram)，タスクスタック領域(h3hstack)，メモリプール領域(h3hmpl)をH'100番地から順に配置します。
　ユーザプログラムをＣ言語で作成した場合、ユーザプログラムとＣ言語インタフェースライブラリのセクション名(h3hc)を設定します。
⑨このシステムのリンケージリストをファイルh3hprg.mapに出力します。

リンケージエディタの詳しい説明は「Ｈシリーズ　リンケージエディタ　ユーザーズマニュアル」を参照してください。

表２－１２にHI8-3Hが使用しているセクション名を示します。

表２－１２　HI8-3Hのセクション名

項番	セクション名	プログラム，作業領域
1	hi8_3h	カーネル
2	h3hsetup	セットアップテーブル
3	h3huser	システム初期化ハンドラ，タイマ初期設定ルーチン，タイマ割込みハンドラ，システム異常終了ルーチン，ＣＰＵ初期化ルーチン
4	hi8_3h_ram	HI8-3Hのシステム作業領域
5	h3hstack	タスクスタック領域
6	h3hmpl	メモリプール領域
7	h3hc	Ｃ言語インタフェースライブラリ（Ｃ言語でシステムコールを使用する場合）
8	h3hcns	コンソールドライバ
9	h3hcns_ram	コンソールドライバ作業領域
１０	h3hilint	未定義割込みハンドラ
１１	h3htrc	トレースバッファ領域

２．１１　メイクファイルの実行

H18-3Hでは、システムを自動的に構築するメイクファイルを提供しています。UNIXシステムのメイク機能を使用し、必要なソースプログラムのみのアセンブル／コンパイルとシステムの結合を行なうことができます。

以下に標準提供のメイクファイルを示します。

・システム構築用メイクファイル
『h3hlnk.mak』　:H8/300Hアドバンストモード用メイクファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
『h3hlnkn.mak』:H8/300Hノーマルモード用メイクファイル(samplenﾃﾞｨﾚｸﾄﾘ)

・アセンブルリスト生成用メイクファイル
『h3hlnkl.mak』　:H8/300Hアドバンストモード用メイクファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
『h3hlnkln.mak』:H8/300Hノーマルモード用メイクファイル(samplenﾃﾞｨﾚｸﾄﾘ)

(1)システム構築用メイクファイルの実行

システム構築用メイクファイルは、標準提供のソースプログラムをアセンブルし、機能モジュールライブラリとの結合を行ないます。

H18-3Hの機能モジュールライブラリとの結合は、サブコマンドファイルにしたがって行ないます。

システム構築用メイクファイルの実行には、次のコマンドを入力します。

% make △ -f △ 〈メイクファイル名〉［△SUB=〈サブコマンドファイル名〉］ (RET)

〈メイクファイル名〉には、表２―１３のシステム構築用メイクファイルを指定します。

表２―１３　システム構築用メイクファイル

項番	〈ﾒｲｸﾌｧｲﾙ名〉	内　容
1	h3hlnk.mak	H8/300Hアドバンストモード用メイクファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
2	h3hlnkn.mak	H8/300Hノーマルモード用メイクファイル(samplenﾃﾞｨﾚｸﾄﾘ)

〈メイクファイル名〉以降のパラメータを省略すると、システム結合時、自動的にパラメータチェック機能なしのサブコマンドファイルが使用されます。

省略時に使用されるサブコマンドファイルを表２―１４に示します。

表２―１４　省略時に使用されるサブコマンドファイル

項番	〈ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ名〉	内　容
1	h3hlnk.sub	H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ(sampleaﾃﾞｨﾚｸﾄﾘ)
2	h3hlnkn.sub	H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用ｻﾌﾞｺﾏﾝﾄﾞﾌｧｲﾙ(samplenﾃﾞｨﾚｸﾄﾘ)

(2)アセンブルリスト生成用メイクファイルの実行

アセンブルリスト生成用メイクファイルは、標準提供のソースプログラムのアセンブルリストファイルの生成を行ないます。

アセンブルリスト生成用メイクファイルの実行には、次のコマンドを入力します。

% make △ -f △ 〈メイクファイル名〉(RET)

〈メイクファイル名〉には、表２―１５のアセンブルリスト生成用メイクファイルを指定します。

表２―１５　アセンブルリスト生成用メイクファイル

項番	〈ﾒｲｸﾌｧｲﾙ名〉	内　容
1	h3hlnk1.mak	H8/300Hアドバンストモード用メイクファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
2	h3hlnk1n.mak	H8/300Hノーマルモード用メイクファイル(samplenﾃﾞｨﾚｸﾄﾘ)

(3)メイクファイルの変更

メイクファイルを使用してユーザプログラムのアセンブルおよびコンパイルを行なう場合、図２－１７の追加を行なってください。

```
#3333333333333333333333333333333333333333333333333333333333333333333333333333333
#HHH    H18-3H makefile Ver2.00 for UNIX                                        HHH
#HHH    Copyright (c)  Hitachi, Ltd. 1993.                                      HHH
#HHH    Licensed Material of Hitachi, Ltd.                                      HHH
#3333333333333333333333333333333333333333333333333333333333333333333333333333333
SUFFIXES = .abs .obj .c .mar .src .lis .sub
.SUFFIXES: $(SUFFIXES)

####### User MACRO define. ##############
DELIMIT=-
OUTPUT_OPTION=$(DELIMIT)nolis$(DELIMIT)debug                    ←――――――――――――  ①
SUB=h3hlnk.sub

####### Assembly section. ###############
AS=asm38
ASFLAGS=
COMPILE.s=$(AS) $(ASFLAGS)
.mar.obj:
        $(COMPILE.s) $< $(OUTPUT_OPTION)
.src.obj:
        $(COMPILE.s) $< $(OUTPUT_OPTION)

####### C language section. #############
CC=ch38
CFLAGS=
CPPFLAGS=$(DELIMIT)CPU=300HA
COMPILE.c=$(CC) $(CFLAGS) $(CPPFLAGS)
.c.obj:
        $(COMPILE.c) $(OUTPUT_OPTION) $<
```

図２－１７　メイクファイルの変更(『h3hlnk.mak』の変更方法)(1/2)

```
####### Linkage section. ###############
h3hprg.abs:     h3hsetup.obj h3huser.obj h3hilint.obj h3hvectr.obj cpuini.obj  \
                <ユーザプログラムのオブジェクトファイル>                          ←―― ②
        rm -f h3hprg.abs
        lnk $(DELIMIT)sub=$(SUB)
        rm h3htmp.rel

####### Target section. ################
h3hsetup.obj:   setup.cmn
h3huser.obj:
h3hilint.obj:
h3hvectr.obj:
cpuini.obj:
<ユーザプログラムのオブジェクトファイル> : <オブジェクト更新に必要なファイル>   ←―― ③
```

図２－１７　メイクファイルの変更(『h3hlnk.mak』の変更方法)(2/2)

(解説)
①必要なオプションを指定します。(デバッグオプションの指定例)
②H18-3Hのシステム結合に必要なユーザプログラムのオブジェクトファイルを指定します。
③作成するユーザプログラムのオブジェクトファイルおよびオブジェクトの更新に必要なソースファイルを指定します。

なお、メイク機能の詳しい説明は、ホストコンピュータ上のＵＮＩＸのマニュアル等を参照してください。

２．１２　システムの起動

システムの結合で作成したロードモジュールをユーザ実機(ターゲットシステム)にダウンロードするには、実機デバッグ装置H8/300H シリーズE7000を使用する方法と、H8/300H 内蔵PROMまたはEPROM に書き込み、ハードウェア環境に搭載する方法があります。

２．１２．１　E7000を使用してロードモジュールのダウンロード

(1) E7000を使用してのロードモジュールのダウンロード

以下にE7000を使用してユーザ実機(ターゲットシステム)にロードモジュールをダウンロードする方法を説明します。

①E7000システムを起動します。

②E7000のFTPコマンドでデータ転送するホストシステムと開局してください。

開局後、BINコマンドでバイナリコードのロードモジュールが転送できるように設定してください。

```
: FTP △ <ホスト名>(RET)

FTP> BIN (RET)
```

③E7000のLAN_LOADコマンドを使用して、ホストシステムからロードモジュールをロードします。

```
FTP> LAN_LOAD;R:<ホストシステムのファイル名> (RET)
```

④ホストシステムとのFTP接続を終了します。

```
FTP> BYE (RET)
```

(2) システムの起動方法

ロードモジュールのスタートアドレスからプログラムを実行すると、HI8-3Hが起動します。

スタートアドレスには、CPU初期化ルーチンの先頭アドレス 『H_3H_CPUINI』(ベクタ番号 0のアドレス)を指定します。

E7000のコマンドを使用した起動方法を次に示します。

```
: RESET (RET)
: GO (RET)
```

なお、E7000についての詳細は、「E7000 H8/3003.H8/3002.H8/3042シリーズ　エミュレータユーザーズマニュアル」を参照してください。

２．１２．２　ハードウェア環境への搭載(Power ONによる起動)

電源をＯＮした状態からシステムを起動するには、HI8-3HのロードモジュールをＲＯＭに書き込み、ハードウェア環境に実装することにより、実現します。

ロードモジュールをフォーマット変換し、モトローラＳタイプ形式ロードモジュールを作成します。

ホストシステムで次のコマンドを入力します。

% cnvs △＜ロードモジュールファイル名＞(RET)

この操作で作成したモトローラＳタイプ形式ロードモジュールをＥＰＲＯＭに書き込み、ハードウェア環境に実装します。

２．１３　システムの異常終了

HI8-3Hは、システムに異常が発生した場合、システム異常終了ルーチンを起動します。

HI8-3Hのシステム異常終了ルーチンには、異常終了原因に応じたプログラムを記述します。（標準提供のHI8-3Hのシステム異常終了ルーチンは、無限ループに入ります。）

この処理中、HI8-3Hの動作は保証できませんので注意してください。

HI8-3Hのシステム起動時と通常動作時のシステム異常終了の原因を示します。

(1)システム起動時のシステム異常終了

HI8-3Hのパラメータチェック有りの機能モジュールライブラリを組み込んだシステムは、次の内容をチェックします。不正が発生した場合、スタックにエラー情報を積んでHI8-3Hのシステム異常終了ルーチンに移行します。

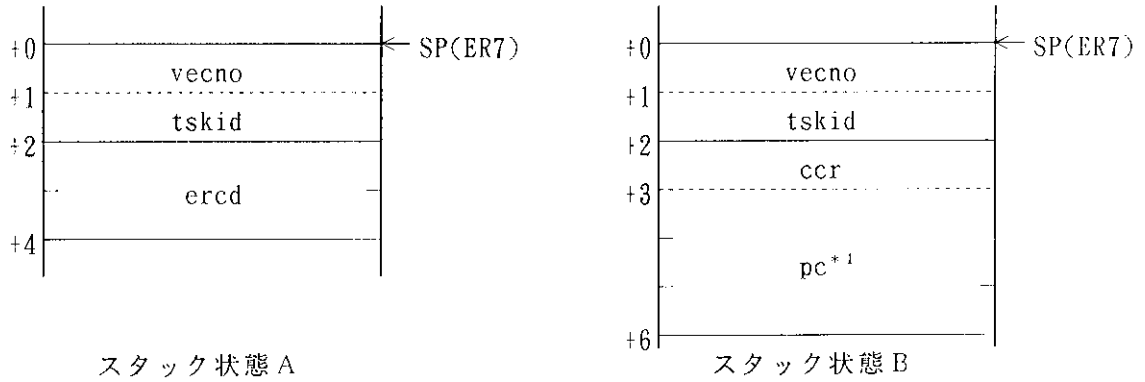
・セットアップ情報のエラー
・タイマ未サポート

(2)通常動作時のシステム異常終了

通常動作中に次の異常が起った場合、スタックにエラー情報を積んでHI8-3Hのシステム異常終了ルーチンに移行します。

・非タスク部からext_tskシステムコール発行
・タスク部からret_intシステムコール発行
・未定義割込みの発生

図２－１８にシステム異常終了時のスタックの状態を示します。



【注】*1　H8/300Hノーマルモードの場合、下位１６ビットが有効になります。

図２－１８　スタックの状態

図２－１８のスタック状態Ａは、セットアップ情報のエラー，タイマ未サポート，非タスク部からext_tskシステムコール発行，タスク部からret_intシステムコール発行の異常が起きたときのスタック状態です。スタック状態Ｂは、未定義割込み発生の異常が起きたときのスタック状態です。

表２－１６にシステム異常終了となる原因とスタックに積まれる値を示します。

表２－１６　システム異常終了となる原因

項番	原　因	vecno	tskid	ercd/ccr.pc
1	セットアップ情報のエラー	H'00	H'00	H'0000～H'0FFF
2	タイマ未サポート	H'00	H'00	H'F9ED
3	非タスク部からの ext_tskシステムコール発行	H'00	H'00	H'FFEB
4	タスク部からの ret_intシステムコール発行	H'00	tskid	H'FFBB
5	未定義割込みの発生	割込みベクタ番号	tskid*	発生時のCCR.PC

【注】*　タスク部で発生した場合はtskid が設定され、非タスク部で発生した場合は０が設定されます。

表２－１７にセットアップ情報不正内容とercdを示します。

表２－１７　セットアップ情報不正内容一覧

項番	不正項目		ercd
1	アドレス不正 （０か奇数）	OS用スタックポインタ(_HI_OS_SP)が０または奇数	H'0101
		タイマ割込み用スタックポインタ(_HI_TIM_SP)が ０または奇数	H'0102
		OS作業領域の先頭アドレス (セクション名 「hi8_3h_ram」)が０または奇数	H'0103
		TIMCB 領域(_HI_TIMCB)が０または奇数	H'0104
		TCB 領域(_HI_TCB)が０または奇数	H'0105
		FLGCB 領域(_HI_FLGCB)が０または奇数	H'0106
		SEMCB 領域(_HI_SEMCB)が０または奇数	H'0107
		MBXCB 領域(_HI_MBXCB)が０または奇数	H'0108
		MPLCB 領域(_HI_MPLCB)が０または奇数	H'0109
		トレーススタックポインタ(_HI_TRC_SP)が０または奇数	H'010A
		トレース管理領域(TBACB)が０または奇数	H'010B
2	ルーチンアドレス不正	システム初期化ハンドラの先頭アドレス (_HIPRG_SYSINI)が奇数	H'0201
		タイマ初期設定ルーチンの先頭アドレス (_HIPRG_TIMINI)が奇数	H'0202
3	設定値不正 （範囲外）	カーネル割込みマスクレベル(IMASK)が４以上	H'0301
		優先度定義数(MAXPRI)が32以上	H'0302
		タスク定義数(TSKCNT)が256 以上	H'0303
		イベントフラグ定義数(FLGCNT)が256 以上	H'0304
		セマフォ定義数(SEMCNT)が256 以上	H'0305
		メイルボックス定義数(MBXCNT)が256 以上	H'0306
		メモリプール定義数(MPLCNT)が256 以上	H'0307
4	セットアップテーブル アドレス不正 （０か奇数）	タスク定義テーブル(_HI_TDT)が０または奇数	H'0401
		メモリプール定義テーブル(_HI_MPLDT)が０または奇数	H'0402
		未定義割込みハンドラ(_HI_ILT)が０または奇数	H'0403
		トレースバッファ情報テーブル(INITRC)が０または奇数	H'0404
5	セットアップテーブル 項目不正	タスク初期優先度(ITSKPRI)が ０または優先度定義数より大きい値	H'0501
		タスク先頭アドレス(TSKADR)が０または奇数	H'0502
		タスクスタックポインタ(ITSKSP)が０または奇数	H'0503
		メモリブロック長(BLKLEN)が０または奇数 または65532バイト以上	H'0504
		メモリプールアドレス(MPLADR)が０または奇数	H'0505
		トレースバッファアドレス(TRACE BUFFER ADDRESS)が０ または奇数	H'0506

3．C言語プログラムの構築方法

本章では、C言語で作成したユーザプログラムをコンパイル，結合する方法を説明します。

3．1　実行形式プログラムの作成

3．1．1　オブジェクトモジュールの作成

C言語で記述したタスクや割込みハンドラなどのユーザプログラムを、他のオブジェクトモジュールやライブラリファイルと結合可能にするために、コンパイルしてオブジェクトモジュールを生成します。

コンパイル時は、コンパイラオプションとしてオブジェクト種類にH8/300Hアドバンストモード(-cpu=300ha)またはH8/300Hノーマルモード(-cpu=300hn)を指定してください。

% ch38 △ -cpu={300ha/300hn} [△-debug] △ <ﾕｰｻﾞﾌﾟﾛｸﾞﾗﾑﾌｧｲﾙ名>(RET)

また、割込みハンドラ等のユーザプログラムでアセンブラ埋め込み機能を使用している場合、コンパイラオプションとしてオブジェクト形式にアセンブリソースプログラム出力(-code=asmcode)を指定し、出力されたアセンブリソースプログラムをアセンブルしてください。

% ch38 △ -cpu={300ha/300hn} △-code=asmcode [△-debug] △ <ﾕｰｻﾞﾌﾟﾛｸﾞﾗﾑﾌｧｲﾙ名>(RET)
% asm38 △ <ｱｾﾝﾌﾞﾘｿｰｽﾌﾟﾛｸﾞﾗﾑﾌｧｲﾙ名> △ -cpu={300ha/300hn} [△-debug] (RET)

Ｈ８／３００シリーズ Ｃコンパイラの詳しい使用方法は、「Ｈ８／３００シリーズ　Ｃコンパイラ　ユーザーズマニュアル」を参照してください。

3．1．2　オブジェクトモジュールの結合

コンパイラによって作成されたオブジェクトモジュールを他のオブジェクトモジュールやライブラリファイルと結合することによりH18-3Hを作成します。

C言語でH18-3Hのシステムコールを使用する場合は、入力ファイルに以下のC言語インタフェースライブラリおよびCコンパイラの標準ライブラリを追加してください。

・C言語インタフェースライブラリ
　『h3hcif.lib』　:H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用C言語ｲﾝﾀﾌｪｰｽﾗｲﾌﾞﾗﾘﾌｧｲﾙ(cifaﾃﾞｨﾚｸﾄﾘ)
　『h3hcif.lib』　:H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用C言語ｲﾝﾀﾌｪｰｽﾗｲﾌﾞﾗﾘﾌｧｲﾙ(cifnﾃﾞｨﾚｸﾄﾘ)

・標準ライブラリ
　『c38ha.lib』　:H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用標準ﾗｲﾌﾞﾗﾘﾌｧｲﾙ
　『c38hn.lib』　:H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用標準ﾗｲﾌﾞﾗﾘﾌｧｲﾙ

Ｃ言語インタフェースライブラリを結合する場合、サブコマンドファイルにＣ言語インタフェースライブラリのセクション『h3hc』（セクションの属性は、code）を追加してください。

リンケージエディタの詳しい使用方法は、「Ｈシリーズ　リンケージエディタ　ユーザーズマニュアル」を参照してください。

図３－１に、Ｃ言語で記述したプログラムを結合する場合のリンケージエディタのサブコマンドファイルの例を示します。

・システム結合用サブコマンドファイル
『h3hlnk.sub』:H8/300Hアドバンストモード用サブコマンドファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
『h3hlnkn.sub』:H8/300Hノーマルモード用サブコマンドファイル(samplenﾃﾞｨﾚｸﾄﾘ)

```
input   h3hsetup,h3huser,<ﾕｰｻﾞﾌﾟﾛｸﾞﾗﾑのｵﾌﾞｼﾞｪｸﾄﾓｼﾞｭｰﾙ>          ←①
library ../knla/h3hkmdl,../cifa/h3hcif,../cnsa/h3hcns,c38ha   ←②
output  h3htmp.rel
form    r
debug
end

input   h3hvectr,h3hilint,cpuini,h3htmp.rel
library ../knla/h3hkdmy,../knla/h3hkmdl
output  h3hprg.abs
start   hi8_3h,h3hsetup,h3huser,<ﾕｰｻﾞﾌﾟﾛｸﾞﾗﾑのｾｸｼｮﾝ名>,h3hc,&  ←③
        hi8_3h_ram,h3hstack,h3hmpl(100)
print   h3hprg.map
debug
exit
```

図３－１　サブコマンドファイルの例(『h3hlnk.sub』)

(解説)
①ユーザプログラムのオブジェクトモジュールを入力します。
②Ｃインタフェースライブラリ(h3hcif.lib)と標準ライブラリ(c38ha.lib)を入力します。
③Ｃインタフェースライブラリ(h3hc)とユーザプログラムのセクション名を追加します。

３．２　メイクファイルの実行

H8-3Hでは、Ｃ言語インタフェースライブラリを自動的に生成するメイクファイルを提供しています。UNIXシステムのメイク機能を使用し、必要なソースプログラムのみをアセンブルし、Ｃ言語インタフェースライブラリを生成することができます。
以下に標準提供のメイクファイルを示します。

・Ｃ言語インタフェースライブラリ生成用メイクファイル
『h3hcif.mak』　:H8/300Hアドバンストモード用メイクファイル(cifaﾃﾞｨﾚｸﾄﾘ)
『h3hcif_n.mak』:H8/300Hノーマルモード用メイクファイル(cifnﾃﾞｨﾚｸﾄﾘ)

・アセンブルリスト生成用メイクファイル
『h3hcifl.mak』　:H8/300Hアドバンストモード用メイクファイル(cifaﾃﾞｨﾚｸﾄﾘ)
『h3hciln.mak』:H8/300Hノーマルモード用メイクファイル(cifnﾃﾞｨﾚｸﾄﾘ)

(1)C言語インタフェースライブラリ生成用メイクファイルの実行

C言語インタフェースライブラリ生成用メイクファイルは、C言語インタフェースのソースプログラムをアセンブルし、ライブラリファイルの生成を行ないます。

C言語インタフェースライブラリ生成用メイクファイルの実行には、次のコマンドを入力します。

% make △ -f △ 〈メイクファイル名〉(RET)

〈メイクファイル名〉には、表3—1のC言語インタフェースライブラリ生成用メイクファイルを指定します。

表3—1　C言語インタフェースライブラリ生成用メイクファイル

項番	〈ﾒｲｸﾌｧｲﾙ名〉	内　容
1	h3hcif.mak	H8/300Hアドバンストモード用メイクファイル(cifaﾃﾞｨﾚｸﾄﾘ)
2	h3hcif_n.mak	H8/300Hノーマルモード用メイクファイル(cifnﾃﾞｨﾚｸﾄﾘ)

(2)アセンブルリスト生成用メイクファイルの実行

アセンブルリスト生成用メイクファイルは、C言語インタフェースのソースプログラムのアセンブルリストファイルの生成を行ないます。

アセンブルリスト生成用メイクファイルの実行には、次のコマンドを入力します。

% make △ -f △ 〈メイクファイル名〉(RET)

〈メイクファイル名〉には、表3—2のアセンブルリスト生成用メイクファイルを指定します。

表3—2　アセンブルリスト生成用メイクファイル

項番	〈ﾒｲｸﾌｧｲﾙ名〉	内　容
1	h3hcifl.mak	H8/300Hアドバンストモード用メイクファイル(cifaﾃﾞｨﾚｸﾄﾘ)
2	h3hcifln.mak	H8/300Hノーマルモード用メイクファイル(cifnﾃﾞｨﾚｸﾄﾘ)

なお、メイクファイルの変更の詳細については、「2．11　メイクファイルの実行」を参照してください。

付録Ａ．メモリ容量の算出

セットアップテーブル(ROM)のサイズおよび使用するメモリ(RAM)容量は以下の算出表から求めてください。

システム初期化ハンドラ、タイマ初期設定ルーチンのスタックサイズを求める場合は、割込みハンドラ用スタックサイズの算出表を使用してください。（カーネル割込みレベルと同一の割込みレベルを持つ割込みハンドラの算出表で算出してください）

Ａ．１　セットアップテーブルサイズの算出

表Ａ－１にセットアップテーブルサイズ(ROM)の算出表を示します。本算出表を使用して、テーブル(ROM) のサイズを求めてください。

本算出表は、H8/300Hノーマルモード、H8/300Hアドバンストモードの両方で使用できます。

表Ａ－１　セットアップテーブルサイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
ｼｽﾃﾑ定数の定義	8	8	常に必要です
ﾀｽｸ定義ﾃｰﾌﾞﾙ	１０×ﾀｽｸ定義数		常に必要です
ﾒﾓﾘﾌﾟｰﾙ定義ﾃｰﾌﾞﾙ	８×ﾒﾓﾘﾌﾟｰﾙ定義数		ﾒﾓﾘﾌﾟｰﾙ を使用する時必要です
ﾄﾚｰｽﾊﾞｯﾌｧ情報ﾃｰﾌﾞﾙ	8		ﾄﾚｰｽ機能を使用する時必要です
合計			

Ａ．２　OS作業領域の算出

表Ａ－２にOS作業領域の算出表を示します。本算出表を使用して、OSの使用する作業領域のサイズを求めてください。

本算出表は、H8/300Hノーマルモード、H8/300Hアドバンストモードの両方で使用できます。

表Ａ－２　OS作業領域の算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
ｼｽﾃﾑ管理ﾃｰﾌﾞﾙ	10＋(優先度定義数×4)		常に必要です
ﾀｽｸ管理ﾌﾞﾛｯｸ	１８×ﾀｽｸ定義数		常に必要です
ｲﾍﾞﾝﾄﾌﾗｸﾞ管理ﾌﾞﾛｯｸ	４×ｲﾍﾞﾝﾄﾌﾗｸﾞ定義数		ｲﾍﾞﾝﾄﾌﾗｸﾞを使用する時必要です
ｾﾏﾌｫ管理ﾌﾞﾛｯｸ	６×ｾﾏﾌｫ定義数		ｾﾏﾌｫを使用する時必要です
ﾒｲﾙﾎﾞｯｸｽ管理ﾌﾞﾛｯｸ	８×ﾒｲﾙﾎﾞｯｸｽ定義数		ﾒｲﾙﾎﾞｯｸｽを使用する時必要です
ﾒﾓﾘﾌﾟｰﾙ管理ﾌﾞﾛｯｸ	６×ﾒﾓﾘﾌﾟｰﾙ定義数		ﾒﾓﾘﾌﾟｰﾙ を使用する時必要です
ﾀｲﾏ管理ﾌﾞﾛｯｸ	１０		時間管理を使用する時必要です
ﾄﾚｰｽﾊﾞｯﾌｧ管理ﾌﾞﾛｯｸ	8		ﾄﾚｰｽ機能を使用する時必要です
合計			

A．3　OS用スタック領域サイズの算出

表A－3，A－4にOSのスタック領域サイズの算出表を示します。本算出表を使用して、OSのスタック領域のサイズを求めてください。

表A－3　OS用スタック領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OSが独自に使用する ｽﾀｯｸｻｲｽﾞ	4	4	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0の 割込み用ｽﾀｯｸｻｲｽﾞ	4		プライオリティレベル0の 割込みを使用する時必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ1の 割込み用ｽﾀｯｸｻｲｽﾞ	4		プライオリティレベル1の 割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用 ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

表A－4　OS用スタック領域サイズの算出表(ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OSが独自に使用する ｽﾀｯｸｻｲｽﾞ	8 *1	8 *1	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0の 割込み用ｽﾀｯｸｻｲｽﾞ	4		プライオリティレベル0の 割込みを使用する時必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ1の 割込み用ｽﾀｯｸｻｲｽﾞ	4		プライオリティレベル1の 割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用 ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

【注】 *1　カーネル割込みマスクレベル(IMASK)を3に設定した場合で割込みを2レベル(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0,1)使用するとき、OSが独自に使用するスタックサイズは4バイトで算出できます。

A．4　タイマ割込み用スタック領域サイズの算出

表A－5，A－6にタイマ割込み用スタック領域サイズの算出表を示します。本算出表を使用して、タイマ割込み用スタック領域のサイズを求めてください。

本算出表は、カーネル割込みマスクレベルを2，3に設定したとき、使用してください。カーネル割込みマスクレベルを0，1に設定した場合、タイマ割込みは使用できません。

表A－5　タイマ割込み用スタック領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
ﾀｲﾏ 割込みが独自に使用するｽﾀｯｸｻｲｽﾞ	28	28	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ1の割込み用ｽﾀｯｸｻｲｽﾞ	6		ﾀｲﾏ 割込みより高いレベルの割込み（NMIを除く）を使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

表A－6　タイマ割込み用スタック領域サイズの算出表(ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
ﾀｲﾏ 割込みが独自に使用するｽﾀｯｸｻｲｽﾞ	30	30	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ1の割込み用ｽﾀｯｸｻｲｽﾞ	6		ﾀｲﾏ 割込みより高いレベルの割込み（NMIを除く）を使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

A．5　タスクスタック領域サイズの算出

表A－7にタスクスタック領域サイズの算出表を示します。本算出表を使用して、タスクID毎のスタック領域のサイズを求めてください。タスク用スタック領域全体のサイズは、各タスクID毎に求めたスタックサイズの総和になります。なお、共有スタック機能を使用する場合は、同じスタック領域を使用するタスクの中で、最も大きなサイズを指定してください。

共有スタック機能を使用する場合は、すべてのスタック領域の最終アドレスから上位アドレス側に８バイト分の共有スタック機能用の領域を確保してください。

タスクをＣ言語で記述した場合、タスクが独自に使用するスタックサイズはコンパイルリストに表される関数のフレームサイズから求めてください。

表Ａ－７　タスクスタック領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
ﾀｽｸ が独自に使用するｽﾀｯｸｻｲｽﾞ	—		
OSが使用するｽﾀｯｸｻｲｽﾞ	３２	３２	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ０の割込み用ｽﾀｯｸｻｲｽﾞ	4		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ０の割込みを使用する時必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	4 *1		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するサイズ	１６		Ｃ言語でタスクを記述しているとき必要です
合計			

【注】*1　カーネル割込みマスクレベル(IMASK)を３に設定した場合で、割込みを２レベル(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0,1)使用するとき、ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用スタックサイズを０バイトで算出できます。

A．６　割込みハンドラ用スタック領域サイズの算出

表Ａ－８～Ａ－１７に割込みハンドラ用スタック領域サイズの算出表を示します。

本算出表を使用して、個々の割込みハンドラのスタック領域のサイズを求めてください。割込みハンドラのスタック領域は、割込みレベル毎に共有できます。同じ割込みレベルの割込みハンドラで最大に使用するときのスタックサイズを確保してください。

割込みハンドラをＣ言語で記述した場合、割込みハンドラが独自に使用するスタックサイズはコンパイルリストに表される関数のフレームサイズから求めてください。

(1)カーネル割込みマスクレベル(IMASK)を０，１に設定した場合

表Ａ－８，Ａ－９にカーネル割込みマスクレベル(IMASK)を０，１に設定した場合の割込みハンドラのスタック領域サイズを示します。

割込みハンドラの割込みレベルは、カーネル割込みマスクレベルより高くなるため、システムコール(ret_int含む)を発行できません。

表Ａ－８　割込ﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 0)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	4		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

表Ａ－９　割込ﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 1)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

(2)カーネル割込みマスクレベル(IMASK)を２に設定した場合

表Ａ－１０～Ａ－１３にカーネル割込みマスクレベル(IMASK)を２に設定した場合の割込みハンドラのスタック領域サイズを示します。

プライオリティレベル１の割込みハンドラの割込みレベルは、カーネル割込みマスクレベルより高くなるため、システムコール(ret_int含む)を発行できません。

表Ａ－１０　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 0)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ﾉｰﾏﾙﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
OSが使用するｽﾀｯｸｻｲｽﾞ	２６	２６	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	6		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するｻｲｽﾞ	１６		Ｃ言語で割込みﾊﾝﾄﾞﾗを記述しているとき必要です
合計			

表Ａ－１１　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 1)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ﾉｰﾏﾙﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

表Ａ－１２　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 0)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
OSが使用するｽﾀｯｸｻｲｽﾞ	２８	２８	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	6		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するサイズ	１６		Ｃ言語で割込みﾊﾝﾄﾞﾗを記述しているとき必要です
合計			

表Ａ－１３　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 1)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

(3)カーネル割込みマスクレベル(IMASK)を３に設定した場合

表Ａ－１４～Ａ－１７にカーネル割込みマスクレベル(IMASK)を３に設定した場合の割込みハンドラ用スタック領域サイズの算出表を示します。

表Ａ－１４　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 0)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ﾉｰﾏﾙﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
OSが使用するｽﾀｯｸｻｲｽﾞ	２６	２６	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	6		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するｻｲｽﾞ	１６		Ｃ言語で割込みﾊﾝﾄﾞﾗを記述しているとき必要です
合計			

表Ａ－１５　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 1)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ﾉｰﾏﾙﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
OSが使用するｽﾀｯｸｻｲｽﾞ	２６	２６	常に必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するｻｲｽﾞ	１６		Ｃ言語で割込みﾊﾝﾄﾞﾗを記述しているとき必要です
合計			

表Ａ－１６　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 0)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
OSが使用するｽﾀｯｸｻｲｽﾞ	２８	２８	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	6		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するサイズ	１６		Ｃ言語で割込みﾊﾝﾄﾞﾗを記述しているとき必要です
合計			

表Ａ－１７　割込みﾊﾝﾄﾞﾗ(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ 1)用ｽﾀｯｸ領域ｻｲｽﾞの算出表(ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
割込みﾊﾝﾄﾞﾗ が独自に使用するｽﾀｯｸｻｲｽﾞ	－		
OSが使用するｽﾀｯｸｻｲｽﾞ	２８	２８	常に必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
ﾄﾚｰｽ用ｽﾀｯｸｻｲｽﾞ	4		ﾄﾚｰｽ機能を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
C言語ｲﾝﾀﾌｪｰｽが使用するｻｲｽﾞ	１６		Ｃ言語で割込みﾊﾝﾄﾞﾗを記述しているとき必要です
合計			

Ａ．７　メモリプール領域サイズの算出

表Ａ－１８に、メモリプール領域サイズの算出表を示します。

本算出表を使用して、メモリプールID毎のメモリプール領域サイズを求めてください。メモリプール領域全体のサイズは、各メモリプールID毎に求めた領域サイズの総和になります。

本算出表は、H8/300Hノーマルモード、H8/300Hアドバンストモードの両方で使用できます。

表Ａ－１８　メモリプール領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
メモリプール領域	(ﾒﾓﾘﾌﾞﾛｯｸｻｲｽﾞ＋４)×ﾒﾓﾘﾌﾞﾛｯｸ数		ﾒﾓﾘﾌﾞﾛｯｸ毎に管理領域(４ﾊﾞｲﾄ)が必要です
合計			

A．８　トレース機能用スタック領域サイズの算出

表Ａ－１９～Ａ－２１に、トレース機能用スタック領域サイズの算出表を示します。本スタック領域は、トレース機能を使用する場合にのみ必要です。

本算出表を使用して、トレース機能が使用するスタック領域のサイズを求めてください。

(1)カーネル割込みマスクレベルを０．１に設定した場合

表Ａ－１９にカーネル割込みマスクレベルを０．１に設定した場合のトレース機能用スタック領域サイズの算出表を示します。

表Ａ－１９　トレース機能用スタック領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OSが使用するｽﾀｯｸｻｲｽﾞ	２６	２６	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ０の割込み用ｽﾀｯｸｻｲｽﾞ	４		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ０の割込みを使用する時必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	４		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	４		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	６		未定義割込みが発生する時必要です
合計			

(2)カーネル割込みマスクレベルを２に設定した場合

表Ａ－２０にカーネル割込みマスクレベルを２に設定した場合のトレース機能用スタック領域サイズの算出表を示します。

表Ａ－２０　トレース機能用スタック領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OSが使用するｽﾀｯｸｻｲｽﾞ	２６	２６	常に必要です
ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込み用ｽﾀｯｸｻｲｽﾞ	４		ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ１の割込みを使用する時必要です
NMI用ｽﾀｯｸｻｲｽﾞ	４		NMI を使用する時必要です
未定義割込み用ｽﾀｯｸｻｲｽﾞ	６		未定義割込みが発生する時必要です
合計			

(3)カーネル割込みマスクレベルを3に設定した場合

表A－21にカーネル割込みマスクレベルを3に設定した場合のトレース機能用スタック領域サイズの算出表を示します。

表A－21　トレース機能用スタック領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OSが使用するｽﾀｯｸｻｲｽﾞ	２６	２６	常に必要です
NMI用ｽﾀｯｸｻｲｽﾞ	4		NMI を使用する時必要です
未定義割込み用 ｽﾀｯｸｻｲｽﾞ	6		未定義割込みが発生する時必要です
合計			

A．9　トレースバッファ領域サイズの算出

表A－22に、トレースバッファ領域サイズの算出表を示します。本算出表を使用して、トレースバッファ領域のサイズを求めてください。

本算出表は、H8/300Hノーマルモード、H8/300Hアドバンストモードの両方で使用できます。

表A－22　トレースバッファ領域サイズの算出表(ﾉｰﾏﾙﾓｰﾄﾞ/ｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ両用)

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
トレースバッファ 管理領域	１６	１６	
トレースエントリ 情報領域	２４×トレース情報 取得数		
合計			

A．１０　HI8-3H作業領域の算出

表Ａ－２３にHI8-3H作業領域の算出表を示します。本算出表を使用して、HI8-3Hが使用するRAMの使用量を求めてください。

表Ａ－２３　HI8-3H作業領域の算出表

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OS作業領域	—		表A-2を参照してください
OS用ｽﾀｯｸ領域	—		表A-3, A-4を参照してください
ﾀｲﾏ 割込み用ｽﾀｯｸ領域	—		表A-5, A-6を参照してください
ﾀｽｸｽﾀｯｸ領域(全体)*1	—		表A-7を参照してください
割込みﾊﾝﾄﾞﾗ用ｽﾀｯｸ領域(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0)*2	—		表A-8, A-10, A-12, A-14, A-16を参照してください
割込みﾊﾝﾄﾞﾗ用ｽﾀｯｸ領域(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ1)*2	—		表A-9, A-11, A-13, A-15, A-17を参照してください
ﾒﾓﾘﾌﾟｰﾙ領域(全体)	—		表A-18を参照してください
ﾄﾚｰｽ機能用ｽﾀｯｸ領域	—		表A-19, A-20, A-21を参照してください
ﾄﾚｰｽﾊﾞｯﾌｧ領域	—		表A-22を参照してください
NMI 割込みﾊﾝﾄﾞﾗ用ｽﾀｯｸ領域	—		
ｼｽﾃﾑ初期化ﾊﾝﾄﾞﾗ用ｽﾀｯｸ領域	—		
CPU初期化ﾙｰﾁﾝ用ｽﾀｯｸ領域*3	—		
ﾀｲﾏ初期設定ﾙｰﾁﾝ用ｽﾀｯｸ領域	—		
その他（　　）	—		
その他（　　）	—		
その他（　　）	—		
その他（　　）	—		
その他（　　）	—		
その他（　　）	—		
合計			

【注】 *1　共有スタック機能を使用する場合、全てのタスクスタック領域に共有スタック用の領域が必要になります。

*2　割込みハンドラのスタック領域は、割込みレベル毎に共有できます。同じ割込みレベルの割込みハンドラで最大に使用するときのスタックサイズを確保してください。

*3　ＣＰＵ初期化ルーチンは、HI8-3Hのシステム起動前に実行されるため、スタック領域として NMI割込みハンドラ用スタック領域以外の任意のRAM領域と共有することができます。

付録Ｂ．　コンソールドライバの登録方法

Ｂ．１　コンソールドライバの登録

標準提供のコンソールドライバを登録する場合は、以下の作業を行なってください。

・コンソールドライバのequate 定義
・タスク定義テーブルへの登録
・ベクタテーブルへの登録

Ｂ．１．１　コンソールドライバのequate 定義

コンソールドライバを使用するためには、ユーザシステムに組み込む際のタスクIDや、コンソールが使用するメイルボックスID、SCI 割込みハンドラのスタック容量をequate 定義してください。
また、使用するH8/3003,H8/3042内蔵のSCI レジスタアドレスをequate 定義してください。

標準提供のコンソールドライバ（ファイル名『h3hcns_0.mar』『h3hcns_1.mar』:H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ用、『h3hcnsn0.mar』『h3hcnsn1.mar』:H8/300Hﾉｰﾏﾙﾓｰﾄﾞ用）には、コンソールドライバの登録に必要な値をequate 定義しています。

表Ｂ－１に標準提供のコンソールドライバのequate 定義を示します。
H8/3003,H8/3042内蔵のSCI の詳しい説明は、H8/3003,H8/3042のハードウェア　マニュアルを参照してください。また、他のH8/300Hシリーズを使用する場合、各CPUのハードウェア　マニュアルを参照してください。

表Ｂ－１　コンソールドライバのequate 定義

項番	ラベル名	内容	設定値	
			SCI0	SCI1
1	CMBXID	コンソールドライバ用 メイルボックスID	H'1	H'2
2	CTSKID	コンソールドライバのタスクID	H'1	H'2
3	CNSHDL_LVL	SCI割込みハンドラの割込みレベル	H'80(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0)	H'80(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0)
4	SMR	SCIのSMR(ｼﾘｱﾙ･ﾓｰﾄﾞ･ﾚｼﾞｽﾀ)アドレス	H'0FFFFB0*1	H'0FFFFB8*1
5	BRR	SCIのBRR(ﾋﾞｯﾄﾚｰﾄ･ﾚｼﾞｽﾀ)アドレス	H'0FFFFB1*1	H'0FFFFB9*1
6	SCR	SCIのSCR(ｼﾘｱﾙ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ)アドレス	H'0FFFFB2*1	H'0FFFFBA*1
7	TDR	SCIのTDR(ﾄﾗﾝｽﾐｯﾄ･ﾃﾞｰﾀ･ﾚｼﾞｽﾀ)アドレス	H'0FFFFB3*1	H'0FFFFBB*1
8	SSR	SCIのSSR(ｼﾘｱﾙ･ｽﾃｰﾀｽ･ﾚｼﾞｽﾀ)アドレス	H'0FFFFB4*1	H'0FFFFBC*1
9	RDR	SCIのRDR(ﾚｼｰﾌﾞ･ﾃﾞｰﾀ･ﾚｼﾞｽﾀ)アドレス	H'0FFFFB5*1	H'0FFFFBD*1
10	CSTACK	SCI割込みハンドラ用スタック領域の 最終アドレス	結合時に決定	結合時に決定

【注】*1　H8/300Hアドバンストモードのアドレス値です。H8/300Hノーマルモードの場合は、下位16ビットのアドレス値になります。

(1) CMBXID(コンソールドライバのメイルボックスID)

コンソールドライバが入出力要求メッセージを受信するためのメイルボックスIDを設定します。このメイルボックスは、コンソールドライバ専用でなければなりません。

(2) CTSKID(コンソールドライバのタスクID)

コンソールドライバのタスクIDを設定します。セットアップテーブルのタスク定義テーブルにコンソールを登録する時、ここで指定したIDと矛盾がないようにしてください。

(3) CNSHDL_LVL(SCI 割込みハンドラの割込みレベル)

コンソールドライバのSCI 割込みの割込みレベルを設定します。

・プライオリティレベル0の割込みとして使用する場合は、H'80を設定してください。
割込みハンドラは、割込みマスクビットI=1,UI=0で実行されます。

・プライオリティレベル1の割込みとして使用する場合は、H'C0を設定してください。
割込みハンドラは、割込みマスクビットI=1,UI=1で実行されます。

(4) SMR(ｼﾘｱﾙ･ﾓｰﾄﾞ･ﾚｼﾞｽﾀ)

SCI のSMRのアドレスを設定します。H8/300Hアドバンストモードの場合は、24ビットのアドレスで記述してください。H8/300Hノーマルモードのときは、下位16ビットのアドレスで記述してください。

(5) BRR(ﾋﾞｯﾄﾚｰﾄ･ﾚｼﾞｽﾀ)

SCI のBRRのアドレスを設定します。H8/300Hアドバンストモードの場合は、24ビットのアドレスで記述してください。H8/300Hノーマルモードのときは、下位16ビットのアドレスで記述してください。

(6) SCR(ｼﾘｱﾙ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ)

SCI のSCRのアドレスを設定します。H8/300Hアドバンストモードの場合は、24ビットのアドレスで記述してください。H8/300Hノーマルモードのときは、下位16ビットのアドレスで記述してください。

(7) TDR(ﾄﾗﾝｽﾐｯﾄ･ﾃﾞｰﾀ･ﾚｼﾞｽﾀ)

SCI のTDRのアドレスを設定します。H8/300Hアドバンストモードの場合は、24ビットのアドレスで記述してください。H8/300Hノーマルモードのときは、下位16ビットのアドレスで記述してください。

(8) SSR(ｼﾘｱﾙ･ｽﾃｰﾀｽ･ﾚｼﾞｽﾀ)

SCI のSSRのアドレスを設定します。H8/300Hアドバンストモードの場合は、24ビットのアドレスで記述してください。H8/300Hノーマルモードのときは、下位16ビットのアドレスで記述してください。

(9) RDR(ﾚｼｰﾌﾞ･ﾃﾞｰﾀ･ﾚｼﾞｽﾀ)

SCI のRDRのアドレスを設定します。H8/300Hアドバンストモードの場合は、24ビットのアドレスで記述してください。H8/300Hノーマルモードのときは、下位16ビットのアドレスで記述してください。

(10) CSTACK(SCI割込みハンドラ用スタックポインタ)

SCI の割込みハンドラで使われるスタック領域の最終アドレスを設定します。

割込みハンドラ独自で使用するスタックサイズは4バイトです。

コンソールドライバの割込みハンドラ用スタックサイズは、「付録Ａ　メモリ容量の算出」を参照してください。

また、CSTACKの後に、割込み発生時のスタックポインタ(SP)を退避する４バイトのエリアを確保してください。

Ｂ．１．２　タスク定義テーブルへの登録

セットアップテーブルのタスク定義テーブルに、コンソールドライバのタスクの先頭アドレスとスタックポインタを登録してください。

・ファイル名『h3hsetup.mar』:H8/300Hｱﾄﾞﾊﾞﾝｽﾄﾓｰﾄﾞ(sampleaﾃﾞｨﾚｸﾄﾘ)
／H8/300Hﾉｰﾏﾙﾓｰﾄﾞ(samplenﾃﾞｨﾚｸﾄﾘ)

表Ｂ－２にコンソールドライバをセットアップテーブルのタスク定義テーブルに登録するために必要な情報を示します。

表Ｂ－２　タスク定義情報一覧

項番	タスク情報	設定値
1	登録／起動要求(IMOD)	ユーザシステムに合せて設定してください
2	初期優先度(ITSKPRI)	ユーザシステムに合せて設定してください
3	タスク先頭アドレス(ITSKADR)	SCI0用は、『H_CNSDRV_0』を設定してください SCI1用は、『H_CNSDRV_1』を設定してください
4	タスクスタックポインタ(ITSKSP)	コンソールドライバのタスク独自で使用するスタックサイズは４バイトです

コンソールドライバのタスク用スタックサイズは、「付録Ａ　メモリ容量の算出」を参照してください。

Ｂ．１．３　割込みベクタテーブルへの登録

コンソールドライバは、SCI 割込みを使用しますので、割込みハンドラの先頭アドレスをベクタテーブルへ登録してください。

(1)H8/300Hアドバンストモードの場合

標準提供のベクタファイル(ファイル名『h3hvectr.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)は、H8/3003を標準として作成しています。

表Ｂ－３に、H8/3003 のSCI 割込みの種類を示します。

表Ｂ－３　H8/3003 のSCI 割込みの種類

項番	割込み要因	SCI0		SCI1	
		ﾍﾞｸﾀ番号	ｱﾄﾞﾚｽ	ﾍﾞｸﾀ番号	ｱﾄﾞﾚｽ
1	ERi割込み(受信エラー割込み)	52	H'D0～H'D3	56	H'E0～H'E3
2	Rxi割込み(入力割込み)	53	H'D4～H'D7	57	H'E4～H'E7
3	Txi割込み(出力割込み)	54	H'D8～H'DB	58	H'E8～H'EB

(a)SCI0用コンソールドライバのソースプログラムファイル（ファイル名『h3hcns_0.mar』:cnsaﾃﾞｨﾚｸﾄﾘ）を使用する場合、割込みベクタテーブルに以下の割込みハンドラを登録してください。

・SCI0のERi 割込みには、ラベル名『H_CNSHDL0ER 』を設定します。
・SCI0のRxi 割込みには、ラベル名『H_CNSHDL0RX 』を設定します。
・SCI0のTxi 割込みには、ラベル名『H_CNSHDL0TX 』を設定します。

(b)SCI1用コンソールドライバのソースプログラムファイル（ファイル名『h3hcns_1.mar』:cnsaﾃﾞｨﾚｸﾄﾘ）を使用する場合、割込みベクタテーブルに以下の割込みハンドラを登録してください。

・SCI1のERi 割込みには、ラベル名『H_CNSHDL1ER 』を設定します。
・SCI1のRxi 割込みには、ラベル名『H_CNSHDL1RX 』を設定します。
・SCI1のTxi 割込みには、ラベル名『H_CNSHDL1TX 』を設定します。

(2)H8/300Hノーマルモードの場合

標準提供のベクタファイル(ファイル名『h3hvectn.mar』:samplenﾃﾞｨﾚｸﾄﾘ)は、H8/3042を標準として作成しています。

表Ｂ－４に、H8/3042 のSCI 割込みの種類を示します。

表Ｂ－４　H8/3042 のSCI 割込みの種類

項番	割込み要因	SCI0		SCI1	
		ﾍﾞｸﾀ番号	ｱﾄﾞﾚｽ	ﾍﾞｸﾀ番号	ｱﾄﾞﾚｽ
1	ERi割込み(受信エラー割込み)	52	H'68～H'69	56	H'70～H'71
2	Rxi割込み(入力割込み)	53	H'6A～H'6B	57	H'72～H'73
3	Txi割込み(出力割込み)	54	H'6C～H'6D	58	H'74～H'75

(a)SCI0用コンソールドライバのソースプログラムファイル（ファイル名『h3hcnsn0.mar』:cnsnﾃﾞｨﾚｸﾄﾘ）を使用する場合、割込みベクタテーブルに以下の割込みハンドラを登録してください。

・SCI0のERi 割込みには、ラベル名『H_CNSHDL0ER 』を設定します。
・SCI0のRxi 割込みには、ラベル名『H_CNSHDL0RX 』を設定します。
・SCI0のTxi 割込みには、ラベル名『H_CNSHDL0TX 』を設定します。

(b)SCI1用コンソールドライバのソースプログラムファイル（ファイル名『h3hcnsn1.mar』:cnsnﾃﾞｨﾚｸﾄﾘ）を使用する場合、割込みベクタテーブルに以下の割込みハンドラを登録してください。

・SCI1のERi 割込みには、ラベル名『H_CNSHDL1ER 』を設定します。
・SCI1のRxi 割込みには、ラベル名『H_CNSHDL1RX 』を設定します。
・SCI1のTxi 割込みには、ラベル名『H_CNSHDL1TX 』を設定します。

B．2　メイクファイルの実行

H18-3Hでは、コンソールドライバのライブラリファイルを自動的に生成するメイクファイルを提供しています。UNIXシステムのメイク機能を使用し、必要なソースプログラムのみをアセンブルし、コンソールドライバのライブラリファイルを生成することができます。

以下に、標準提供のメイクファイルを示します。

・コンソールドライバライブラリ生成用メイクファイル
『h3hcns.mak』　　H8/300Hアドバンストモード用メイクファイル(cnsaﾃﾞｨﾚｸﾄﾘ)
『h3hcns_n.mak』　　H8/300Hノーマルモード用メイクファイル(cnsnﾃﾞｨﾚｸﾄﾘ)
・アセンブルリス生成用メイクファイル
『h3hcnsl.mak』　　H8/300Hアドバンストモード用メイクファイル(cnsaﾃﾞｨﾚｸﾄﾘ)
『h3hcnsln.mak』　　H8/300Hノーマルモード用メイクファイル(cnsnﾃﾞｨﾚｸﾄﾘ)

(1)コンソールドライバライブラリ生成用メイクファイルの実行

コンソールドライバライブラリ生成用メイクファイルは、コンソールドライバのソースプログラムをアセンブルし、ライブラリファイルの生成を行ないます。

コンソールドライバライブラリ生成用メイクファイルの実行には、次のコマンドを入力します。

% make△-f△〈メイクファイル名〉(RET)

〈メイクファイル名〉には、表B－5のコンソールドライバライブラリ生成用メイクファイルを指定します。

表B－5　コンソールドライバライブラリ生成用メイクファイル

項番	〈ﾒｲｸﾌｧｲﾙ名〉	内　容
1	h3hcns.mak	H8/300Hアドバンストモード用メイクファイル(cnsaﾃﾞｨﾚｸﾄﾘ)
2	h3hcns_n.mak	H8/300Hノーマルモード用メイクファイル(cnsnﾃﾞｨﾚｸﾄﾘ)

(2)アセンブルリスト生成用メイクファイルの実行

アセンブルリスト生成用メイクファイルは、コンソールドライバのソースプログラムのアセンブルリストファイルの生成を行ないます。

アセンブルリスト生成用メイクファイルの実行には、次のコマンドを入力します。

% make△-f△〈メイクファイル名〉(RET)

〈メイクファイル名〉には、表B－6のアセンブルリスト生成用メイクファイルを指定します。

表B－6　アセンブルリスト生成用メイクファイル

項番	〈ﾒｲｸﾌｧｲﾙ名〉	内　容
1	h3hcnsl.mak	H8/300Hアドバンストモード用メイクファイル(cnsaﾃﾞｨﾚｸﾄﾘ)
2	h3hcnsln.mak	H8/300Hノーマルモード用メイクファイル(cnsnﾃﾞｨﾚｸﾄﾘ)

付録Ｃ．　標準提供のHI8-3Hシステム作成例

Ｃ．１　概　要

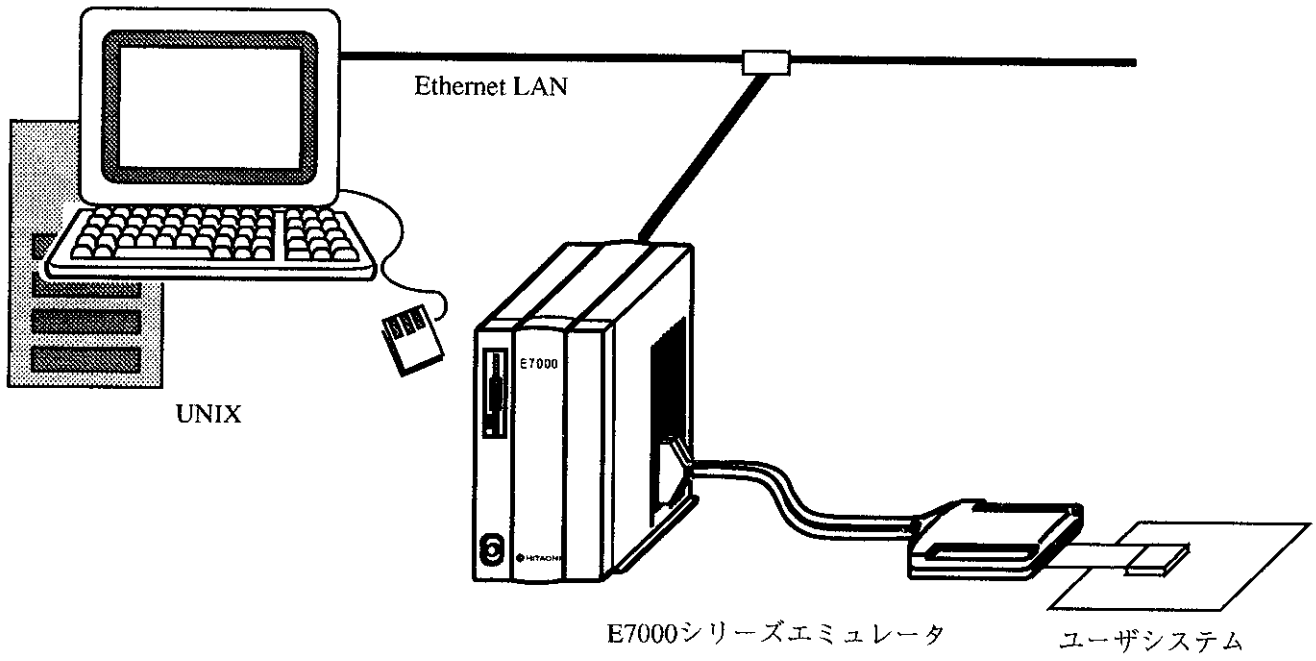
Ｈ Ｉ ８ － ３ Ｈで標準提供しているコンソールドライバを使用して，実際にHI8-3Hのシステムを作成する例を示します。本例では、H8/300H アドバンストモード用のHI8-3Hのシステムの作成を、UNIXシステム上で行ないます。

標準提供のコンソールドライバを使用する場合は，本例を参考にしてユーザシステムを開発してください。

Ｃ．２　HI8-3Hのシステム構成

Ｃ．２．１　ハードウェア構成

本例では、ホストシステムにUNIX，実機デバッグ装置にH8/300Hシリーズ用E7000を使用します。
以下にハードウェア構成を示します。



図Ｃ－１　ハードウェア構成

C．2．2　ソフトウェア構成

本例では、ホストシステム(UNIX)上に、以下のソフトウェアが必要です。

ユーティリティソフトウェア：

①Ｈ８／３００シリーズ　クロスアセンブラ
②Ｈ８／３００シリーズ　Ｃコンパイラ
③Ｈ８／３００シリーズ　シミュレータデバッガ
④Ｈシリーズ　ライブラリアン
⑤Ｈシリーズ　リンケージエディタ
⑥Ｈシリーズ　オブジェクトコンバータ
なお、②，③は本例では使用しておりません。

表Ｃ－１に本例で使用するH8-3Hのシステムファイル一覧を示します。

表Ｃ－１　H8-3Hのシステムファイル一覧

項番	ﾃﾞｨﾚｸﾄﾘ	ﾌｧｲﾙ名	内　容
1	samplea	cpuini.mar	CPU初期化ルーチンのソースプログラムファイル
2		h3hilint.mar	未定義割込みハンドラのソースプログラムファイル
3		h3hlnk.mak	H8-3Hシステム作成MAKEファイル
4		h3hlnk.sub	結合用サブコマンドファイル(ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能無し)
5		h3hsetup.mar. setup.cmn	セットアップテーブルのソースプログラムファイル
6		h3huser.mar	ﾀｲﾏﾄﾞﾗｲﾊﾞ、ｼｽﾃﾑ異常終了ﾙｰﾁﾝ、ｼｽﾃﾑ初期化ﾊﾝﾄﾞﾗのソースプログラムファイル
7		h3hvectr.mar	割込みベクタテーブルのソースプログラムファイル
8	knla	h3hkdmy.lib	カーネルダミーライブラリ
9		h3hkmdl.lib	カーネル機能モジュールライブラリ(ﾊﾟﾗﾒｰﾀﾁｪｯｸ機能無し)
10	cifa	h3hcif.lib	Ｃ言語インタフェースライブラリのライブラリファイル
11	cnsa	h3hcns.cmn	コンソールドライバの共通ソースプログラムファイル
12		h3hcns.mak	コンソールドライバライブラリ作成MAKEファイル
13		h3hcns.sub	ライブラリ作成用サブコマンドファイル
14		h3hcns_0.mar	コンソールドライバのソースプログラムファイル(H8/3003内蔵SCI0)
15		h3hcns_1.mar	コンソールドライバのソースプログラムファイル(H8/3003内蔵SCI1)

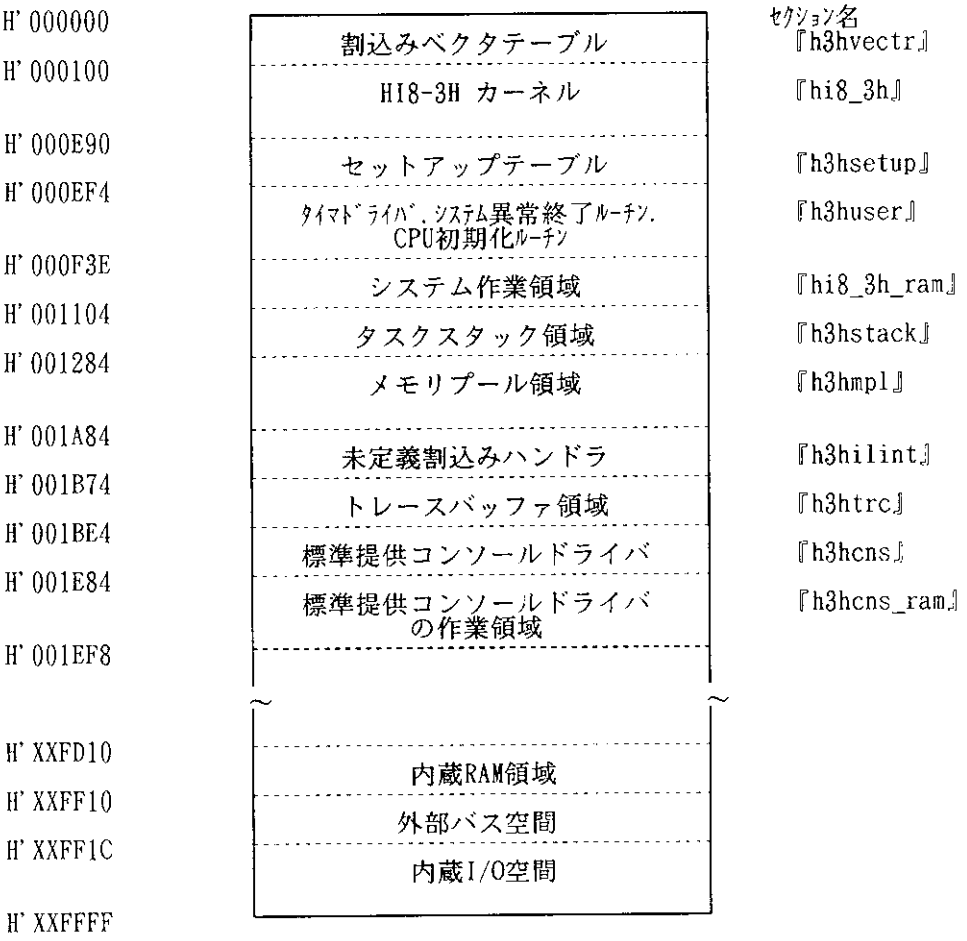
表Ｃ－２　に標準提供のHI8-3Hのシステム概要を示します。

表Ｃ－２　HI8-3Hのシステム概要

項番	項目	内 容
1	MPU	H8/300Hアドバンストモード（H8/3003の例）
2	ﾒﾓﾘ	ROMなし，内蔵RAM 512バイト，外部ROM/RAM使用します
3	ﾀｽｸ	タスクID= 1～ 5までの５個を使用します タスクID= 1は、標準提供のコンソールドライバを登録しています タスクID= 2は、標準提供のコンソールドライバを登録しています タスクID= 3,4はタスクを未登録に設定しています タスクID= 4,5は共有スタック領域を使用する設定にしています
4	ｲﾍﾞﾝﾄﾌﾗｸﾞ	イベントフラグID= 1～ 4までの 4個を登録しています
5	ｾﾏﾌｫ	セマフォID= 1～ 4までの 4個を登録しています
6	ﾒｲﾙﾎﾞｯｸｽ	メイルボックスID= 1～ 4までの 4個を登録しています
7	ﾒﾓﾘﾌﾟｰﾙ	メモリプールID= 1～ 4までの 4個を登録しています メモリプールID= 1～ 4は、メモリブロック数=32個，メモリブロック長=12バイトを定義しています
8	割込み	プライオリティレベル０はコンソールドライバのSCI割込みに使用します プライオリティレベル１はのシステムクロックのタイマ割込みに使用します
9	ｶｰﾈﾙ割込みﾏｽｸﾚﾍﾞﾙ	セットアップテーブルに設定されているカーネル割込みマスクレベルには、３を設定しています カーネル割込みマスクレベルより高いレベルの割込みは使用していません
１０	ｶｰﾈﾙ	パラメータチェック機能を組み込みます
１１	ｼｽﾃﾑ初期化ﾊﾝﾄﾞﾗ	使用しません
１２	CPU初期化ﾙｰﾁﾝ	ｼｽﾃﾑｺﾝﾄﾛｰﾙﾚｼﾞｽﾀ(SYSCR)を設定し、割込みレベルを2レベル使用可能にしています
１３	ｼｽﾃﾑ異常終了ﾙｰﾁﾝ	カーネル割込みマスクレベルの状態で無限ループする処理が登録されています

C．3　メモリマップ

標準提供のリンケージエディタのサブコマンドファイル『h3hlnk.sub』を使用した場合、図C－2に示す配置になります。



図C－2　HI8-3Hのシステムメモリマップ

C．4　使用する作業領域の算出

本例で使用する作業領域を示します。各領域のサイズは、「付録A　メモリ容量の算出」を参照してください。表C－3に本例が使用する作業領域の算出表を示します。

表C－3　作業領域の算出表

内　訳	計算式	容量(ﾊﾞｲﾄ)	備　考
OS作業領域	10＋(31×4) ＋18×5 ＋ 4×4 ＋ 6×4 ＋ 8×4 ＋ 6×4 ＋10 ＋8	338	「付録A　メモリ容量の算出」を参照してください
OS用ｽﾀｯｸ領域	8＋4＋4＋4＋6	26	付録Aを参照してください
ﾀｲﾏ 割込み用ｽﾀｯｸ領域	30＋6＋4＋6	46	付録Aを参照してください
ﾄﾚｰｽ のｽﾀｯｸ領域	26＋4＋4＋4＋6	44	付録Aを参照してください
全ﾀｽｸ のｽﾀｯｸ領域*1	82＋82＋110＋110	384	付録Aを参照してください
割込みﾊﾝﾄﾞﾗのｽﾀｯｸ領域(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ0)*2	56＋56	112	ｺﾝｿｰﾙﾄﾞﾗｲﾊﾞのSCI0,SCI1割込みﾊﾝﾄﾞﾗ用ｽﾀｯｸ領域
割込みﾊﾝﾄﾞﾗのｽﾀｯｸ領域(ﾌﾟﾗｲｵﾘﾃｨﾚﾍﾞﾙ1)	0	0	未使用です
NMI割込みﾊﾝﾄﾞﾗのｽﾀｯｸ領域	0	0	未使用です
ｼｽﾃﾑ初期化ﾊﾝﾄﾞﾗのｽﾀｯｸ領域	0	0	未登録です
CPU初期化ﾙｰﾁﾝのｽﾀｯｸ領域	0	0	スタック未使用です
ﾀｲﾏ初期設定ﾙｰﾁﾝのｽﾀｯｸ領域	0	0	スタック未使用です
コンソールドライバの作業領域	2＋2	4	コンソールドライバが独自に使用する領域です(SCI0,SCI1用)
メモリプール領域	32×(12＋4)×4	2048	メモリプール機能が使用する領域です
トレースバッファ領域	16＋(4×24)	112	トレース機能が使用する領域です
合計		3114	

【注】*1　全てのタスクスタック領域に、共有スタック用の領域（8バイト）を加算しています。
*2　割込みハンドラのスタック領域は、割込みレベル毎に共有できます。同じ割込みレベルの割込みハンドラで最大に使用するときのスタックサイズを確保してください。
本例では、割込みハンドラのスタック領域を共有していません。

C．5　セットアップテーブル

標準提供のセットアップテーブル(ファイル名『h3hsetup.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)は、本例で修正なしに使用できるよう作成してあります。

図C－3にセットアップテーブルのリストを示します。次にセットアップテーブルの内容を説明します。

```
;3333333333333333333333333333333333333333333333333333333333333333333333333333333333333333
;HHH                                                                              HHH
;HHH    H18-3H Version (uITRON V2.01/02)                                          HHH
;HHH    H18-3H/kernel setup table Ver2.00                                         HHH
;HHH                                                                              HHH
;HHH    Copyright (c) Hitachi, Ltd. 1993.                                         HHH
;HHH    Licensed Material of Hitachi, Ltd.                                        HHH
;HHH                                                                              HHH
;3333333333333333333333333333333333333333333333333333333333333333333333333333333333333333
        .program        h3hsetup
        .heading        "### h3hsetup.mar : H18-3H setupfile V2.00 ###"
;
;#########################################################################
;###                                                                   ###
;###   User define section                                             ###  <---- ユーザ定義部
;###                                                                   ###
;#########################################################################
        .section        h3hsetup,code,align=2
;
        .radix  d                       ;:xxxxx -> d'xxxxx
;
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%    VALUE define section                                                %%%  <---- 定数定義部
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;------ Usage ---------------------------------------------------------------
;LABEL                   VALUE   ;:[ RANGE ]     ;: COMMENT
;----------------------------------------------------------------------------
IMASK:          .equ    03      ;:[0.....3]     ;: Max interrupt level  <---- ｶｰﾈﾙ割込みﾏｽｸﾚﾍﾞﾙ=3
MAXPRI:         .equ    31      ;:[0....31]     ;: Max low priority     <---- 優先度定義数=31
FLGCNT:         .equ    4       ;:[0...255]     ;: Eventflag definition count  <- ｲﾍﾞﾝﾄﾌﾗｸﾞ定義数=4
SEMCNT:         .equ    4       ;:[0...255]     ;: Semaphore definition count  <- ｾﾏﾌｫ定義数=4
MBXCNT:         .equ    4       ;:[0...255]     ;: Mailbox definition count  <- ﾒｲﾙﾎﾞｯｸｽ定義数=4
;
OSSTKSIZ:       .equ    8+4+4+4+6;:[8.....]     ;: OS stack size     <---- OSｽﾀｯｸｻｲｽﾞ=26
TIMSTKSIZ:      .equ    30+6+4+6;:[0.30...]     ;: Timer stack size  <---- ﾀｲﾏｽﾀｯｸｻｲｽﾞ=46
TRCSTKSIZ:      .equ    26+4+4+4+6;:[0.26...]   ;: Trace stack size  <---- ﾄﾚｰｽｽﾀｯｸｻｲｽﾞ=44
;
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%    TASK define section                                                 %%%  <---- タスク登録部
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;------ Usage ---------------------------------------------------------------
;               TASK_TOP_LABEL                  ;: COMMENT
;----------------------------------------------------------------------------
        .import H_CNSDRV0                       ;: console (SCI0)
        .import H_CNSDRV1                       ;: console (SCI1)
;
```

図C－3　セットアップテーブルのリスト(1／3)

```
;------ Usage ---------------------------------------------------
;                .res.b  SIZE +TSKSTKSIZ ;[RANGE];: COMMENT
;TSK?_SP_LABEL: .equ    $                      ;: COMMENT
;----------------------------------------------------------------
TSKSTKSIZ:      .equ    32+4+4+4+4+6+16 ;[32...];: Task minimum stack size← タスク最小スタックサイズ
        .section        h3hstack, stack, align=2
                .res.b  ( 4) +TSKSTKSIZ ;[32...];: tskid1 stack area← タスク用スタック領域(ID=1)
TSK1_SP:        .equ    $                                        ← タスクスタックポインタ
                .res.b  8                                        ← 共有スタック用管理領域
                .res.b  ( 4) +TSKSTKSIZ ;[32...];: tskid2 stack area← タスク用スタック領域(ID=2)
TSK2_SP:        .equ    $                                        ← タスクスタックポインタ
                .res.b  8                                        ← 共有スタック用管理領域
                .res.b  (32) +TSKSTKSIZ ;[32...];: tskid3 stack area← タスク用スタック領域(ID=3)
TSK3_SP:        .equ    $                                        ← タスクスタックポインタ
                .res.b  8                                        ← 共有スタック用管理領域
                .res.b  (32) +TSKSTKSIZ ;[32...];: tskid4 stack area← タスク用スタック領域(ID=4)
TSK4_SP:        .equ    $                                        ← タスクスタックポインタ
                .res.b  8                                        ← 共有スタック用管理領域
;
;------ Usage ---------------------------------------------------
;LABEL          .data.b IMOD, IPRI              ;: COMMENT
;               .data.l TSK_TOP, TSK_SP         ;: COMMENT
;----------------------------------------------------------------
NOEXS:          .equ    0                       ;: initial mode = NO EXIST
RDY:            .equ    1                       ;: initial mode = READY
DMT:            .equ    (-1)                    ;: initial mode = DORMANT
TDTLEN:         .equ    10;<- Not Change !      ;: TDT Length
        .section        h3hsetup, code, align=2
_HI_TDT:        .equ    $-TDTLEN                ;: Task define table← タスク定義テーブル
TDT_TOP:        .equ    $                       ;:
tdt_id1:        .data.b RDY, 1                  ;: init. mode, init. priority← タスク情報
                .data.l H_CNSDRV0, TSK1_SP      ;: top address, stack pointer   (タスクID=1)
tdt_id2:        .data.b RDY, 1                  ;: init. mode, init. priority← タスク情報
                .data.l H_CNSDRV1, TSK2_SP      ;: top address, stack pointer   (タスクID=2)
tdt_id3:        .data.b NOEXS, 3                ;: init. mode, init. priority← タスク情報
                .data.l 0, TSK3_SP              ;: top address, stack pointer   (タスクID=3)
tdt_id4:        .data.b NOEXS, 4                ;: init. mode, init. priority← タスク情報
                .data.l 0, TSK4_SP              ;: top address, stack pointer   (タスクID=4)
tdt_id5:        .data.b NOEXS, 5                ;: init. mode, init. priority← タスク情報
                .data.l 0, TSK4_SP              ;: top address, stack pointer   (タスクID=5)
TDT_BTM:
TSKCNT:         .equ    (TDT_BTM-TDT_TOP)/TDTLEN                ← タスク定義数=5
                                ;:[0...255]     ;: Task definition count
;
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%    MEMORYPOOL define section                            %%%← メモリプール登録部
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;------ Usage ---------------------------------------------------
;MB?_CNT_LABEL: .equ    VALUE   ;:[  RANGE  ]   ;: COMMENT
;MB?_LEN_LABEL: .equ    VALUE   ;:[  RANGE  ]   ;: COMMENT
;----------------------------------------------------------------
MB1_CNT:        .equ    32      ;:[0...65535]   ;: memory block count   ← メモリブロック数=32
MB1_LEN:        .equ    12      ;:[2...65530]   ;: memory block length  ← メモリブロック長=12
;                                                                         (ID=1)
MB2_CNT:        .equ    32      ;:[0...65535]   ;: memory block count   ← メモリブロック数=32
MB2_LEN:        .equ    12      ;:[2...65530]   ;: memory block length  ← メモリブロック長=12
;                                                                         (ID=2)
MB3_CNT:        .equ    32      ;:[0...65535]   ;: memory block count   ← メモリブロック数=32
MB3_LEN:        .equ    12      ;:[2...65530]   ;: memory block length  ← メモリブロック長=12
;                                                                         (ID=3)
MB4_CNT:        .equ    32      ;:[0...65535]   ;: memory block count   ← メモリブロック数=32
MB4_LEN:        .equ    12      ;:[2...65530]   ;: memory block length  ← メモリブロック長=12
;                                                                         (ID=4)
```

図C－３　セットアップテーブルのリスト(２／３)

```
;------ Usage ---------------------------------------------------------
;MPL?_TOP_LABEL:.res.b  MEMORYPOOL_SIZE          ;: COMMENT
;----------------------------------------------------------------------
        .section        h3hmpl.data.align=2
MPL1_TOP:       .res.b  MB1_CNT * (MB1_LEN + 4) ;: mplid1 memorypool area <---- ﾒﾓﾘﾌﾟｰﾙ領域(ID=1)
MPL2_TOP:       .res.b  MB2_CNT * (MB2_LEN + 4) ;: mplid2 memorypool area <---- ﾒﾓﾘﾌﾟｰﾙ領域(ID=2)
MPL3_TOP:       .res.b  MB3_CNT * (MB3_LEN + 4) ;: mplid3 memorypool area <---- ﾒﾓﾘﾌﾟｰﾙ領域(ID=3)
MPL4_TOP:       .res.b  MB4_CNT * (MB4_LEN + 4) ;: mplid4 memorypool area <---- ﾒﾓﾘﾌﾟｰﾙ領域(ID=4)
;
;------ Usage ---------------------------------------------------------
;LABEL          .data.w BLKCNT, BLKLEN          ;: COMMENT
;               .data.l MEMORYPOOL_TOP          ;: COMMENT
;----------------------------------------------------------------------
MPLDTLEN:       .equ    8;<- Not Change !       ;: MPLDT Length
        .section        h3hsetup.code.align=2
_HI_MPLDT:      .equ    $-MPLDTLEN              ;: MemoryPool define table <---- ﾒﾓﾘﾌﾟｰﾙ定義ﾃｰﾌﾞﾙ
MPLDT_TOP:      .equ    $                       ;:
mpldt_id1:      .data.w MB1_CNT, MB1_LEN        ;: Memory block count, length <-- ﾒﾓﾘﾌﾟｰﾙ情報
                .data.l MPL1_TOP                ;: Memorypool top address          (ID=1)
mpldt_id2:      .data.w MB2_CNT, MB2_LEN        ;: Memory block count, length <-- ﾒﾓﾘﾌﾟｰﾙ情報
                .data.l MPL2_TOP                ;: Memorypool top address          (ID=2)
mpldt_id3:      .data.w MB3_CNT, MB3_LEN        ;: Memory block count, length <-- ﾒﾓﾘﾌﾟｰﾙ情報
                .data.l MPL3_TOP                ;: Memorypool top address          (ID=3)
mpldt_id4:      .data.w MB4_CNT, MB4_LEN        ;: Memory block count, length <-- ﾒﾓﾘﾌﾟｰﾙ情報
                .data.l MPL4_TOP                ;: Memorypool top address          (ID=4)
MPLDT_BTM:
MPLCNT:         .equ    (MPLDT_BTM-MPLDT_TOP)/MPLDTLEN                  <---- ﾒﾓﾘﾌﾟｰﾙ定義数=4
                                ;:[0...255]     ;: Memorypool definition count
;
;------ Usage ---------------------------------------------------------
;TRC_CNT:.equ   TRACE COUNT                     ;: COMMENT              <---- ﾄﾚｰｽ機能登録部
;TRC_BUF:.equ   TRACE BUFFER ADDRESS            ;: COMMENT
;----------------------------------------------------------------------
        .section        h3htrc,data,align=2
TRC_CNT:        .equ    4                       ;: trace count          <---- 最大ﾄﾚｰｽ取得数
TRC_BUF:        .res.b  16+(TRC_CNT*24)         ;: trace buffer address <---- ﾄﾚｰｽﾊﾞｯﾌｧ領域
;
;------ Usage ---------------------------------------------------------
;INITRC         .data.l TRACE BUFFER ADDRESS    ;: COMMENT
;               .data.w TRACE COUNT             ;: COMMENT
;----------------------------------------------------------------------
        .section        h3hsetup.code.align=2
INITRC:         .equ    $                       ;:                      <---- ﾄﾚｰｽﾊﾞｯﾌｧ定義ﾃｰﾌﾞﾙ
                .data.l TRC_BUF                 ;: trace buffer address <---- ﾄﾚｰｽﾊﾞｯﾌｧｱﾄﾞﾚｽ
                .data.w TRC_CNT                 ;: trace count          <---- 最大ﾄﾚｰｽ取得数
;
        .include        "setup.cmn"             <---------- H18-3Hｼｽﾃﾑ定義部
;
;*********************************************;
        .end; of H3HSETUP.MAR
```

図Ｃ－３　セットアップテーブルのリスト(３／３)

(1)セットアップテーブルの定数定義部

(a)『IMASK』(カーネル割込みマスクレベル)には、３を定義しています。

(b)『MAXPRI』(優先度定義数)には、３１を定義しています。

(c)『FLGCNT』(イベントフラグ定義数)には、４を定義しています。

(d)『SEMCNT』(セマフォ定義数)には、４を定義しています。

(e)『MBXCNT』(メイルボックス定義数)には、４を定義しています。

(f)『OSSTKSIZ』(ＯＳスタックサイズ)には、２６バイトを定義しています。

(g)『TIMSTKSIZ』(タイマスタックサイズ)には、４６バイトを定義しています。

(h)『TRCSTKSIZ』(トレーススタックサイズ)には、４４バイトを定義しています。

(2)セットアップテーブルのタスク登録部

タスクを使用するために必要な情報を登録します。

(a)『TSKSTKSIZ』(タスク最小スタックサイズ)には、７０バイトを定義しています。

(b)『TSKx_SP』(タスクスタック領域) には、タスク(ID=1～5)が使用するスタック領域を確保しています。共有スタック機能を使用するため、全てのスタック領域のスタックポインタから上位アドレス側に８バイトを共有スタック管理領域として確保します。

タスクID=1が使用するスタック領域の最終アドレスに、ラベル『TSK1_SP』を設定します。

タスクID=2が使用するスタック領域の最終アドレスに、ラベル『TSK2_SP』を設定します。

タスクID=3が使用するスタック領域の最終アドレスに、ラベル『TSK3_SP』を設定します。

タスクID=4が使用するスタック領域の最終アドレスに、ラベル『TSK4_SP』を設定します。

タスクID=5が使用するスタック領域の最終アドレスに、ラベル『TSK4_SP』を設定します。

(タスクID=4,5は、同一のスタック領域(共有スタック)を使用します)

(c)『_HI_TDT』(タスク定義テーブル)に、５つのタスクの情報を定義しています。

①タスクID= 1

タスクID= 1には、コンソールドライバ(H8/3003内蔵SCI0用)を定義しています。

登録／起動要求には、実行可能状態を設定しています。

初期優先度には、1(最高優先度)を設定しています。

タスク先頭アドレスには、コンソールドライバのラベル『H_CNSDRV0』を設定しています。

タスクスタックポインタには、タスクID= 1のスタックポインタ『TSK1_SP』を設定しています。

②タスクID= 2

タスクID= 2には、コンソールドライバ(H8/3003内蔵SCI1用)を定義しています。

登録／起動要求には、実行可能状態を設定しています。

初期優先度には、1(最高優先度)を設定しています。

タスク先頭アドレスには、コンソールドライバのラベル『H_CNSDRV1』を設定しています。

タスクスタックポインタには、タスクID= 2のスタックポインタ『TSK2_SP』を設定しています。

③タスクID=3

タスクID=3は、登録／起動要求に0(ラベル『NOEXS』)を設定し、未登録にしています。

未登録ですが、初期優先度(優先度= 3)、タスクスタックポインタ(ラベル『TSK3_SP』)は設定しています。

④タスクID=4

タスクID=4は、登録／起動要求に0(ラベル『NOEXS』)を設定し、未登録にしています。

未登録ですが、初期優先度(優先度= 4)、タスクスタックポインタ(ラベル『TSK4_SP』)は設定しています。タスクID=4は、共有スタック領域(タスクID=5と同一のスタック領域)を使用します。

⑤タスクID=5

タスクID=5は、登録／起動要求に0(ラベル『NOEXS』)を設定し、未登録にしています。

未登録ですが、初期優先度(優先度= 5)、タスクスタックポインタ(ラベル『TSK4_SP』)は設定しています。タスクID=5は、共有スタック領域(タスクID=4と同一のスタック領域)を使用します。

(d)『TSKCNT』(タスク定義数)には、タスク定義テーブルのサイズから求めたタスクの総数を設定しています。

(3)セットアップテーブルのメモリプール登録部

メモリプールを使用するために必要な情報を登録します。

(a)『MBx_CNT』(メモリブロック数)には、メモリプールID=1～4のメモリプールが使用するメモリブロックの数を設定しています。

メモリプールID=1のメモリブロック数『MB1_CNT』に32を設定しています。

メモリプールID=2のメモリブロック数『MB2_CNT』に32を設定しています。

メモリプールID=3のメモリブロック数『MB3_CNT』に32を設定しています。

メモリプールID=4のメモリブロック数『MB4_CNT』に32を設定しています。

(b)『MBx_LEN』(メモリブロック長)には、メモリプールID=1～4のメモリプールが使用するメモリブロックの長さを設定しています。

メモリプールID=1のメモリブロック長『MB1_LEN』に12を設定しています。

メモリプールID=2のメモリブロック長『MB2_LEN』に12を設定しています。

メモリプールID=3のメモリブロック長『MB3_LEN』に12を設定しています。

メモリプールID=4のメモリブロック長『MB4_LEN』に12を設定しています。

(c)『MPLx_TOP』(メモリプール領域)には、メモリプールID=1～4までのメモリプール領域を確保しています。

メモリプールID=1のメモリプール領域の先頭アドレスにラベル『MPL1_TOP』を設定しています。

メモリプールID=2のメモリプール領域の先頭アドレスにラベル『MPL2_TOP』を設定しています。

メモリプールID=3のメモリプール領域の先頭アドレスにラベル『MPL3_TOP』を設定しています。

メモリプールID=4のメモリプール領域の先頭アドレスにラベル『MPL4_TOP』を設定しています。

(d)『_HI_MPLDT』(メモリプール定義テーブル)

①メモリプールID=1

使用するメモリブロック数には、『MB1_CNT』を設定しています。

使用するメモリブロック長には、『MB1_LEN』を設定しています。

メモリプール領域の先頭アドレスには、『MPL1_TOP』を設定しています。

②メモリプールID=2

使用するメモリブロック数には、『MB2_CNT』を設定しています。

使用するメモリブロック長には、『MB2_LEN』を設定しています。

メモリプール領域の先頭アドレスには、『MPL2_TOP』を設定しています。

③メモリプールID=3

使用するメモリブロック数には、『MB3_CNT』を設定しています。

使用するメモリブロック長には、『MB3_LEN』を設定しています。

メモリプール領域の先頭アドレスには、『MPL3_TOP』を設定しています。

④メモリプールID=4

使用するメモリブロック数には、『MB4_CNT』を設定しています。

使用するメモリブロック長には、『MB4_LEN』を設定しています。

メモリプール領域の先頭アドレスには、『MPL4_TOP』を設定しています。

(e)『MPLCNT』(メモリプール定義数)には、メモリプール定義テーブルのサイズから求めたメモリプールの総数を設定しています。

C．6　割込みベクタテーブル

ユーザシステムで，割込みに対し独自の処理に制御を移すための割込みハンドラの先頭アドレスを登録するテーブルです。(ファイル名『h3hvectr.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)

(1)リセット割込みには、CPU初期化ルーチンを設定しています。

(2)H8/3003内蔵のITUのTGAI0割込みには、タイマドライバを設定しています。

(3)H8/3003内蔵SCI0, SCI1のeri割込み、rxi割込み、txi割込みには、コンソールドライバを設定しています。

(4)使用しない割込みベクタには、未定義割込みハンドラを設定しています。

標準提供の割込みベクタテーブル『h3hvectr.mar』は、本例でそのまま使用できます。

図C－4に割込みベクタテーブルを示します。

```
;3333333333333333333333333333333333333333333333333333333333333333333333333333333333333333
;HHH                                                                              HHH
;HHH    HI8-3H Version (uITRON V2.01/02)                                          HHH
;HHH    HI8-3H vector table for H8/3003 Ver2.00                                   HHH
;HHH                                                                              HHH
;HHH    Copyright (c) Hitachi, Ltd. 1993.                                         HHH
;HHH    Licensed Material of Hitachi, Ltd.                                        HHH
;HHH                                                                              HHH
;3333333333333333333333333333333333333333333333333333333333333333333333333333333333333333
        .program        h3hvectr
        .heading        "### h3hvectr.mar : for H8/3003 ###"
        .section        h3hvectr,code,locate=0
;
        .import         H_3HINT01,H_3HINT02,H_3HINT03,H_3HINT04,H_3HINT05
        .import         H_3HINT06,H_3HINT07,H_3HINT08,H_3HINT09,H_3HINT10
        .import         H_3HINT11,H_3HINT12,H_3HINT13,H_3HINT14,H_3HINT15
        .import         H_3HINT16,H_3HINT17,H_3HINT18,H_3HINT19,H_3HINT20
```

図C－4　割込みベクタテーブル『h3hvectr.mar』(1／3)

```
        .import         H_3HINT21,H_3HINT22,H_3HINT23,H_3HINT24,H_3HINT25
        .import         H_3HINT26,H_3HINT27,H_3HINT28,H_3HINT29,H_3HINT30
        .import         H_3HINT31,H_3HINT32,H_3HINT33,H_3HINT34,H_3HINT35
        .import         H_3HINT36,H_3HINT37,H_3HINT38,H_3HINT39,H_3HINT40
        .import         H_3HINT41,H_3HINT42,H_3HINT43,H_3HINT44,H_3HINT45
        .import         H_3HINT46,H_3HINT47,H_3HINT48,H_3HINT49,H_3HINT50
        .import         H_3HINT51,H_3HINT52,H_3HINT53,H_3HINT54,H_3HINT55
        .import         H_3HINT56,H_3HINT57,H_3HINT58,H_3HINT59,H_3HINT60
;
        .import         H_3H_CPUINI     ;: in 'cpuini'
        .import         H_3H_TIM        ;: in 'h3huser'
        .import         H_CNSHDL0ER     ;: in 'h3hcns_0'
        .import         H_CNSHDL0RX     ;: in 'h3hcns_0'
        .import         H_CNSHDL0TX     ;: in 'h3hcns_0'
        .import         H_CNSHDL1ER     ;: in 'h3hcns_1'
        .import         H_CNSHDL1RX     ;: in 'h3hcns_1'
        .import         H_CNSHDL1TX     ;: in 'h3hcns_1'
;
        .import         H_SHARED        ;: not change
        .import         H_DBGHDL        ;: not change
;
;************************************************************************
;*specifications ;                                                      *
;*name       = h3hvectr : h8/3003 vector table for HI8-3H            ;  *
;*function   = 1. h8 interrupt handler address define for hi8        ;  *
;*           = 2. h8 exception handler address define for hi8        ;  *
;*           = 3. hi8 system standard support module                 ;  *
;*           =    (1) reset    : "H_300H_INI"  for power on          ;  *
;*           = 4. hi8 system standard support module                 ;  *
;*           =    (1) ITU0 tgai: "H_3H_TIM"    for system timer      ;  *
;*           = 5. hi8 system standard support i/o interrupt handler  ;  *
;*           =    (1) for console i/o driver [sci use]               ;  *
;*           =             (a) : "H_CNSHDL0ER" for receive error     ;  *
;*           =             (b) : "H_CNSHDL0RX" for receive  (input)  ;  *
;*           =             (c) : "H_CNSHDL0TX" for transmit (output) ;  *
;*           =             (d) : "H_CNSHDL1ER" for receive error     ;  *
;*           =             (e) : "H_CNSHDL1RX" for receive  (input)  ;  *
;*           =             (f) : "H_CNSHDL1TX" for transmit (output) ;  *
;*date       = 92/09/17                                              ;  *
;*author     = Hitachi, Ltd.                                         ;  *
;*history    = V1.00/V2.00                                           ;  *
;*attribute  = public                                                ;  *
;*class      = unit                                                  ;  *
;*linkage    = h8 vector table top address = h'0000 for HI8-3H       ;  *
;*input      = none                                                  ;  *
;*output     = none                                                  ;  *
;*parameter  = @er7(B) : interrupt before "CCR"                      ;  *
;*           = @er7(L) : interrupt before "PC"(next mnemonic)        ;  *
;*end of specifications ;                                            *
;************************************************************************
;
        .radix  d                       ;:xxxxx -> d'xxxxx
;
;-------.data.l < address >             ;: h8/3003 vector no. contents
        .data.l H_3H_CPUINI             ;: vector no.00 <reset>
        .data.l H_SHARED;H_3HINT01      ;: vector no.01 [reserve]
        .data.l H_DBGHDL;H_3HINT02      ;: vector no.02 [reserve]
        .data.l H_3HINT03               ;: vector no.03 [reserve]
        .data.l H_3HINT04               ;: vector no.04 [reserve]
        .data.l H_3HINT05               ;: vector no.05 [reserve]
        .data.l H_3HINT06               ;: vector no.06 [reserve]
        .data.l H_3HINT07               ;: vector no.07 <NMI        >
        .data.l H_3HINT08               ;: vector no.08 <TRAPA  #0  >
        .data.l H_3HINT09               ;: vector no.09 <TRAPA  #1  >
        .data.l H_3HINT10               ;: vector no.10 <TRAPA  #2  >
        .data.l H_3HINT11               ;: vector no.11 <TRAPA  #3  >
        .data.l H_3HINT12               ;: vector no.12 <IRQ0       >
        .data.l H_3HINT13               ;: vector no.13 <IRQ1       >
```

図C－4　割込みベクタテーブル『h3hvectr.mar』（2／3）

```
        .data.l H_3HINT14               ;: vector no.14 <IRQ2        >
        .data.l H_3HINT15               ;: vector no.15 <IRQ3        >
        .data.l H_3HINT16               ;: vector no.16 <IRQ4        >
        .data.l H_3HINT17               ;: vector no.17 <IRQ5        >
        .data.l H_3HINT18               ;: vector no.18 <IRQ6        >
        .data.l H_3HINT19               ;: vector no.19 <IRQ7        >
        .data.l H_3HINT20               ;: vector no.20 <WDT    wovi >
        .data.l H_3HINT21               ;: vector no.21 <WDT    wovi >
        .data.l H_3HINT22               ;: vector no.22 <WDT    wovi >
        .data.l H_3HINT23               ;: vector no.23 <WDT    wovi >
        .data.l H_3H_TIM;H_3HINT24      ;: vector no.24 <ITU0   tgai >-option
        .data.l H_3HINT25               ;: vector no.25 <ITU0   tgbi >
        .data.l H_3HINT26               ;: vector no.26 <ITU0   tovi >
        .data.l H_3HINT27               ;: vector no.27 [reserve]
        .data.l H_3HINT28               ;: vector no.28 <ITU1   tgai >
        .data.l H_3HINT29               ;: vector no.29 <ITU1   tgbi >
        .data.l H_3HINT30               ;: vector no.30 <ITU1   tovi >
        .data.l H_3HINT31               ;: vector no.31 [reserve]
        .data.l H_3HINT32               ;: vector no.32 <ITU2   tgai >
        .data.l H_3HINT33               ;: vector no.33 <ITU2   tgbi >
        .data.l H_3HINT34               ;: vector no.34 <ITU2   tovi >
        .data.l H_3HINT35               ;: vector no.35 [reserve]
        .data.l H_3HINT36               ;: vector no.36 <ITU3   tgai >
        .data.l H_3HINT37               ;: vector no.37 <ITU3   tgbi >
        .data.l H_3HINT38               ;: vector no.38 <ITU3   tovi >
        .data.l H_3HINT39               ;: vector no.39 [reserve]
        .data.l H_3HINT40               ;: vector no.40 <ITU4   tgai >
        .data.l H_3HINT41               ;: vector no.41 <ITU4   tgbi >
        .data.l H_3HINT42               ;: vector no.42 <ITU4   tovi >
        .data.l H_3HINT43               ;: vector no.43 [reserve]
        .data.l H_3HINT44               ;: vector no.44 <DMAC03 dend0>
        .data.l H_3HINT45               ;: vector no.45 <DMAC03 dend1>
        .data.l H_3HINT46               ;: vector no.46 <DMAC03 dend2>
        .data.l H_3HINT47               ;: vector no.47 <DMAC03 dend3>
        .data.l H_3HINT48               ;: vector no.48 <DMAC47 dend4>
        .data.l H_3HINT49               ;: vector no.49 <DMAC47 dend5>
        .data.l H_3HINT50               ;: vector no.50 <DMAC47 dend6>
        .data.l H_3HINT51               ;: vector no.51 <DMAC47 dend7>
        .data.l H_CNSHDL0ER;H_3HINT52   ;: vector no.52 <SCI0   eri  >-option
        .data.l H_CNSHDL0RX;H_3HINT53   ;: vector no.53 <SCI0   rxi  >-option
        .data.l H_CNSHDL0TX;H_3HINT54   ;: vector no.54 <SCI0   txi  >-option
        .data.l H_3HINT55               ;: vector no.55 <SCI0   tend >
        .data.l H_CNSHDL1ER;H_3HINT56   ;: vector no.56 <SCI1   eri  >-option
        .data.l H_CNSHDL1RX;H_3HINT57   ;: vector no.57 <SCI1   rxi  >-option
        .data.l H_CNSHDL1TX;H_3HINT58   ;: vector no.58 <SCI1   txi  >-option
        .data.l H_3HINT59               ;: vector no.59 <SCI1   tend >
        .data.l H_3HINT60               ;: vector no.60 <ADC    adi  >
;
;*************************************;
        .end; of H3HVECTR.MAR
```

図Ｃ－４　割込みベクタテーブル『h3hvectr.mar』（３／３）

C．7　タイマドライバ、システム異常終了ルーチン、CPU初期化ルーチン、システム初期化ハンドラ

標準提供のタイマドライバ、システム異常終了ルーチン、CPU初期化ルーチン、システム初期化ハンドラのプログラム（ファイル名『h3huser.mar』.『cpuini.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）は、以下のように登録されています。

(1)タイマドライバ（ファイル名『h3huser.mar』）

(a)タイマ初期設定ルーチン

タイマ初期設定ルーチンは、H8/3003内蔵のITU(ｲﾝﾃｸﾞﾚｰﾃｯﾄﾞﾀｲﾏﾊﾟﾙｽﾕﾆｯﾄ)をシステムのタイマとして使用できるように初期化しています。

(b)タイマ割込みハンドラ

タイマ割込みハンドラは、タイマ割込みをクリア（タイマ割込みリセット処理）し、ＯＳのタイマ割込み処理にジャンプしています。

(2)システム異常終了ルーチン（ファイル名『h3huser.mar』）

標準提供のシステム異常終了ルーチンは、カーネル割込みマスクレベルで無限ループする処理が登録されています。

(3)CPU初期化ルーチン（ファイル名『cpuini.mar』）

標準提供のCPU初期化ルーチンは、割込みベクタテーブルのリセット割込みに登録されています。

(4)システム初期化ハンドラ（ファイル名『h3huser.mar』）

標準提供のシステム初期化ハンドラは、未登録に設定されています。

C．8　HI8-3Hのシステム構築手順

HI8-3Hをユーザシステムに搭載するには，以下の２通りの方法があります。

(1)　HI8-3Hの実行形式ロードモジュールをH8/300Hシリーズ E7000のロードコマンド(LAN_LOAD)を用いて，ユーザシステム上へロードします。

(2)　HI8-3Hの実行形式ロードモジュールを ROM化してユーザシステムに実装します。

C．9　HI8-3Hの実行形式ロードモジュールの生成

標準提供ファイルをもとにHI8-3Hのロードモジュールを生成します。

HI8-3Hのロードモジュールの生成手順を以下に示します。

(1)　セットアップテーブルのアセンブル

(2)　割込みベクタテーブルのアセンブル

(3)　未定義割込みハンドラのアセンブル

(4)　コンソールドライバのアセンブル、ライブラリファイルの作成

(5)　タイマドライバ、システム異常終了ルーチン、CPU初期化ルーチン、システム初期化ハンドラのアセンブル

(6)　HI8-3Hのシステム結合

以上の手順で，HI8-3Hのロードモジュールを作成します。生成されるロードモジュールを以下に示します。

・HI8-3Hのシステム実行形式ロードモジュール：『h3hprg.abs』

また、標準提供のメイクファイルを使用して、(1)～(6)のアセンブルおよび結合を行なうことができます。メイクファイルの実行については、「C．9．4　コンソールドライバのアセンブル、ライブラリファイルの作成」および「C．9．7　メイクファイルの実行」を参照してください。

C．9．1　セットアップテーブルのアセンブル

本例では、標準提供のセットアップテーブル（ファイル名『h3hsetup.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）をそのまま使用します。

『h3hsetup.mar』の存在するディレクトリで、次のコマンドを実行します。アセンブルのコマンドラインオプションとして、CPU種別にH8/300Hアドバンストモードを指定します（省略可）。

```
% asm38△h3hsetup.mar［△-cpu=300ha］［△-debug］(RET)
```

この操作でセットアップテーブルのオブジェクトモジュール『h3hsetup.obj』が生成されます。

C．9．2　割込みベクタテーブルのアセンブル

本例では、標準提供の割込みベクタテーブル（ファイル名『h3hvectr.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）をそのまま使用します。

『h3hvectr.mar』の存在するディレクトリで、次のコマンドを実行します。アセンブルのコマンドラインオプションとして、CPU種別にH8/300Hアドバンストモードを指定します（省略可）。

```
% asm38△h3hvectr.mar［△-cpu=300ha］［△-debug］(RET)
```

この操作で割込みベクタテーブルのオブジェクトモジュール『h3hvectr.obj』が生成されます。

C．9．3　未定義割込みハンドラのアセンブル

本例では、標準提供の未定義割込みハンドラ（ファイル名『h3hilint.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）をそのまま使用します。

『h3hilint.mar』の存在するディレクトリで、次のコマンドを実行します。アセンブルのコマンドラインオプションとして、CPU種別にH8/300Hアドバンストモードを指定します（省略可）。

```
% asm38△h3hilint.mar［△-cpu=300ha］［△-debug］(RET)
```

この操作で未定義割込みハンドラのオブジェクトモジュール『h3hilint.obj』が生成されます。

C．9．4　コンソールドライバのアセンブル、ライブラリファイルの作成

本例では、標準提供のコンソールドライバ（ファイル名『h3hcns.cmn』，『h3hcns_0.mar』，『h3hcns_1.mar』:cnsaﾃﾞｨﾚｸﾄﾘ）をそのまま使用します。

コンソールドライバのソースプログラムファイルをアセンブルし、システムと結合可能なライブラリファイル『h3hcns.lib』を生成します。

ライブラリファイルには以下のオブジェクトモジュールが含まれます。

①　『h3hcns_0.obj』　：H8/3003内蔵SCI0用コンソールドライバ

②　『h3hcns_1.obj』　：H8/3003内蔵SCI1用コンソールドライバ

以下にアセンブル、ライブラリファイルの作成方法を示します。

標準提供のメイクファイル（ファイル名『h3hcns.mak』:cnsaﾃﾞｨﾚｸﾄﾘ）を実行して、コンソールドライバのライブラリファイルを作成します。

コンソールドライバのソースプログラムが存在するディレクトリで次のコマンドを入力し、メイクファイルを実行します。

```
% make△-f△h3hcns.mak (RET)
```

この操作で、コンソールドライバのライブラリファイル『h3hcns.lib』が作成されます。

C．9．5　タイマドライバ、システム異常終了ルーチン、CPU初期化ルーチン、システム初期化ハンドラのアセンブル

本例では、標準提供のタイマドライバ、システム異常終了ルーチン、CPU初期化ルーチン、システム初期化ハンドラ（ファイル名『h3huser.mar』，『cpuini.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）をそのまま使用します。

『h3huser.mar』，『cpuini.mar』の存在するディレクトリで、次のコマンドを実行します。アセンブルのコマンドラインオプションとして、CPU種別にH8/300Hアドバンストモードを指定します（省略可）。

```
% asm38△h3huser.mar［△-cpu=300ha］［△-debug］(RET)
% asm38△cpuini.mar［△-cpu=300ha］［△-debug］(RET)
```

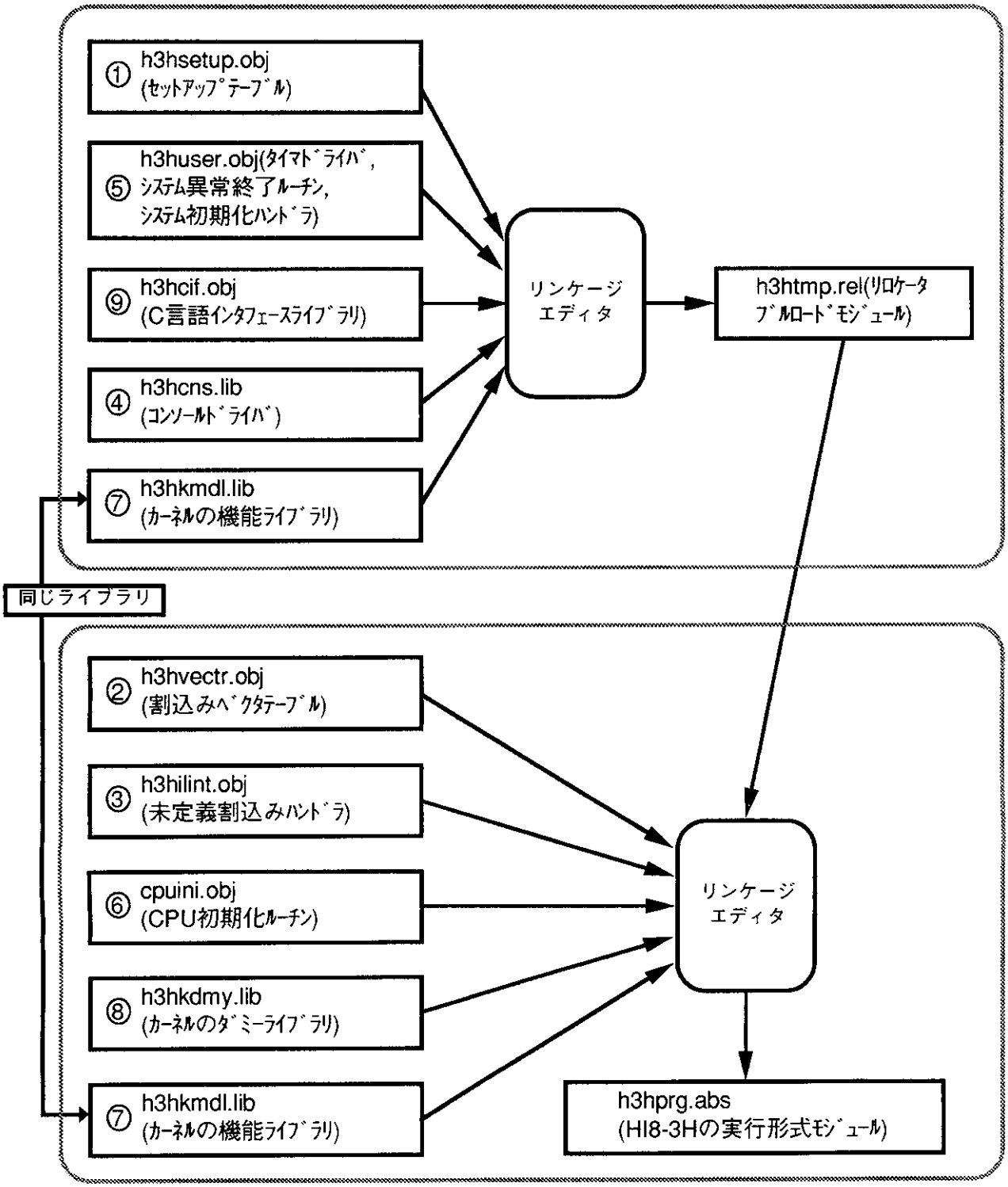
この操作でタイマドライバ、システム異常終了ルーチン、CPU初期化ルーチン、システム初期化ハンドラのオブジェクトモジュール『h3huser.obj』，『cpuini.obj』が生成されます。

C．9．6　HI8-3Hのシステム結合

以上、作成した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をもとに，HI8-3Hのシステムの実行形式ロードモジュール『h3hprg.abs』を生成します。

必要に応じて，モトローラSタイプ形式ロードモジュールに変換してください。

図C－5にシステム結合の概要を示します。



図C－5　システム結合の概要

次に，HI8-3Hのシステム結合方法を示します。
以下に示したリンケージエディタの入力ファイルを用意します。

① 『h3hsetup.obj』：セットアップテーブルのオブジェクトファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
② 『h3hvectr.obj』：割込みベクタテーブルのオブジェクトファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
③ 『h3hilint.obj』：未定義割込みハンドラのオブジェクトファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
④ 『h3hcns.lib』　：コンソールドライバのライブラリファイル(cnsaﾃﾞｨﾚｸﾄﾘ)
⑤ 『h3huser.obj』：タイマドライバ、システム異常終了ルーチン、システム初期化ハンドラ
(sampleaﾃﾞｨﾚｸﾄﾘ)
⑥ 『cpuini.obj』　：CPU初期化ルーチンのオブジェクトファイル(sampleaﾃﾞｨﾚｸﾄﾘ)
⑦ 『h3hkmdl.lib』：カーネルの機能ライブラリファイル(knlaﾃﾞｨﾚｸﾄﾘ)
⑧ 『h3hkdmy.lib』：カーネルのダミーライブラリファイル(knlaﾃﾞｨﾚｸﾄﾘ)
⑨ 『h3hcif.lib』　：Ｃ言語インタフェースライブラリのライブラリファイル(cifaﾃﾞｨﾚｸﾄﾘ)

ファイルの準備が完了した後、次のコマンドを入力します。

```
% lnk △ -sub=h3hlnk.sub (RET)
```

リンケージエディタを起動し、サブコマンドファイル（ファイル名『h3hlnk.sub』:sampleaﾃﾞｨﾚｸﾄﾘ）の内容にしたがって，各ファイルを結合します。そして、HI8-3Hの実行形式モジュール『h3hprg.abs』を生成します。

HI8-3Hのサブコマンドファイル『h3hlnk.sub』を図Ｃ－６に示します。

```
input   h3hsetup,h3huser
library ../knla/h3hkmdl,../cifa/h3hcif,../cnsa/h3hcns
output  h3htmp.rel
form    r
debug
end

input   h3hvectr,h3hilint,cpuini,h3htmp.rel
library ../knla/h3hkdmy,../knla/h3hkmdl
output  h3hprg.abs
start   hi8_3h,h3hsetup,h3huser,&
        hi8_3h_ram,h3hstack,h3hmpl(100)
print   h3hprg.map
debug
exit
```

図Ｃ－６　HI8-3Hのサブコマンドファイル『h3hlnk.sub』

C．9．7　メイクファイルの実行

標準提供のメイクファイル（ファイル名『h3hlnk.mak』:sampleaﾃﾞｨﾚｸﾄﾘ）は、HI8-3Hを構築するためのメイクファイルです。メイクファイルは、コンソールドライバを除く以下のソースプログラムをアセンブルします。

アセンブルするソースプログラムファイル

① セットアップテーブル（ファイル名『h3hsetup.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）

② 割込みベクタテーブル（ファイル名『h3hvectr.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）

③ 未定義割込みハンドラ（ファイル名『h3hilint.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）

④ タイマドライバ、システム異常終了ルーチン、システム初期化ハンドラ
（ファイル名『h3huser.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）

⑤ CPU初期化ルーチン（ファイル名『cpuini.mar』:sampleaﾃﾞｨﾚｸﾄﾘ）

このメイクファイルは上記のソースプログラムファイルをアセンブルし、指定されたサブコマンドファイルにしたがって、機能モジュールライブラリとの結合を行ないます。

メイクファイルの実行には、次のコマンドを入力します。

```
% make△-f△h3hlnk.mak△sub=h3hlnk.sub(RET)
```

メイクファイル以降のパラメータを省略すると、結合時に自動的に『h3hlnk.sub』がサブコマンドファイルとなります。メイクファイルの実行により、HI8-3Hの実行形式モジュール『h3hprg.abs』が生成されます。

C．9．8　実行形式ロードモジュールのロード

HI8-3Hの実行形式ロードモジュール『h3hprg.abs』を、E7000のロードコマンド(LAN_LOAD)を用いてターゲットシステム上へロードします。

FTPコマンドでデータ転送するホストシステムと開局してください。

開局後、BINコマンドでバイナリコードのロードモジュールが転送できるように設定してください。

```
: FTP <ホスト名>(RET)
FTP> BIN (RET)
FTP> LAN_LOAD;R:h3hprg.abs (RET)
```

この操作でホストシステムから実行形式ロードモジュール『h3hprg.abs』をターゲットシステムにロードします。

なお、E7000についての詳細は、「E7000　H8/3003.H8/3002.H8/3042シリーズ　エミュレータ　ユーザーズマニュアル」を参照してください。

C．9．9　実行形式ロードモジュールの実装

HI8-3Hのロードモジュール『h3hprg.abs』をフォーマット変換し，モトローラＳタイプ形式ロードモジュール『h3hprg.mot』を作成します。

コマンドライン上で次のコマンドを実行します。

```
% cnvs△h3hprg.abs (RET)
```

この操作で生成されたモトローラＳタイプ形式ロードモジュール『h3hprg.mot』をＲＯＭ化し，ユーザシステムに実装します。

C．9．10　HI8-3Hの起動

システムを起動するには，E7000のGOコマンドを使用します。
システムのスタートアドレスには，ＣＰＵ初期化ルーチンの先頭アドレス『H_3H_CPUINI』を指定します。

E7000のコマンドライン上で次のコマンドを実行します。

```
: RESET (RET)
: GO (RET)
```

なお、E7000についての詳細は、「E7000　H8/3003,H8/3002,H8/3042シリーズ　エミュレータ　ユーザーズマニュアル」を参照してください。

付録D. H8-3H V1.0からV2.0への移行

D. 1 H8-3H V2.0への移行方法

H8-3H V1.0用のアプリケーションは、V2.0用のアドバンストモードで実行させることができます。H8-3H V2.0への移行は、以下の方法で行なってください。

(1)ファイルの差し替え

以下の①～④のファイルを除く全てのファイルを、H8-3H V2.0のファイル(H8/300Hアドバンストモード用)に差し替えてください。

①セットアップテーブル(ファイル名『h3hsetup.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)

②割込みベクタテーブル(ファイル名『h3hvectr.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)

③CPU初期化ルーチン(ファイル名『cpuini.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)

④システム初期化ハンドラ，タイマハンドラ，システム異常終了ルーチン
(ファイル名『h3huser.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)

②の割込みベクタテーブルは、H8-3H V1.0のファイルを使用できます。

H8-3H V1.0で③，④のファイルを修正している場合はV2.0のファイルに差し替え、V1.0のファイルと同じ修正を行なってください。ファイルを修正していない場合は、V2.0のファイルに差し替えてください。

(2)セットアップテーブルの変更

H8-3H V1.0 のセットアップテーブル(ファイル名『h3hsetup.mar』:sampleaﾃﾞｨﾚｸﾄﾘ)をV2.0で使用するためには、共有スタック機能およびトレース機能を設定するための変更が必要になります。

図D－1にセットアップテーブルの変更箇所を示します

(a)共有スタック機能

共有スタック機能を使用する場合は、以下の変更が必要になります。共有スタック機能を使用しない場合は、セットアップテーブルの変更は必要ありません。

①共有スタック用の領域

共有スタック機能を使用する場合、すべてのタスクスタック領域に共有スタック用の領域（８バイト）を追加してください。共有スタック用の領域は、スタックの最終アドレス（スタックポインタ）から上位アドレス側に追加してください。

②共有スタック領域を使用するタスクの登録

共有スタック領域を使用するタスクは、タスク定義テーブルに同一のスタックポインタを設定してください。

(b)トレース機能

トレース機能の使用または未使用を設定するため、以下の変更が必要になります。

①トレース用スタックサイズ

トレース機能を使用する場合、トレース用スタックサイズを設定してください。トレース用スタックサイズは、「付録Ａ．メモリ容量の算出」を参照してください。

トレース機能を使用しない場合、トレース用スタックサイズに０を設定してください。

②トレース機能登録部

トレース機能を使用する場合、トレース機能で取得する情報の数の設定、トレース機能で取得する情報を退避するバッファ領域を確保してください。トレースバッファ領域のサイズは、「付録Ａ．メモリ容量の算出」を参照してください。また、トレースバッファ情報テーブルにトレースバッファ領域の先頭アドレス、トレース機能で取得する情報の数を設定してください。

トレース機能を使用しない場合、トレースバッファ領域を確保する必要はありません。また、トレースバッファ情報テーブルには、バッファ領域の先頭アドレス、取得する情報の数に０を設定してください。

③タスク、割込みハンドラのスタックサイズ

トレース機能を使用する場合、タスク用スタック領域サイズにトレース機能用のスタックサイズ（４バイト）を加算してください。また、作成した割込みハンドラのスタック領域サイズにトレース機能用のサイズ（４バイト）を加算してください。

トレース機能を使用しない場合、タスク用スタック領域サイズ、割込みハンドラのスタック領域サイズに４バイトを加算する必要はありません。

```
30 行目 FLGCNT:          .equ    4        ;:[0...255]     ;; Eventflag definition count
31 行目 SEMCNT:          .equ    4        ;:[0...255]     ;; Semaphore definition count
32 行目 MBXCNT:          .equ    4        ;:[0...255]     ;; Mailbox definition count
33 行目 ;
34 行目 OSSTKSIZ:        .equ    8+4+4+4+6;:[8.....]      ;; OS stack size
35 行目 TIMSTKSIZ:       .equ    30+6+4+6;:[0,30...]      ;; Timer stack size
36 行目 TRCSTKSIZ:       .equ    26+4+4+4+6;:[0,26...]    ;; Trace stack size  ← トレース用スタックサイズ (b)-①
                                        .
                                        .
51 行目 TSKSTKSIZ:       .equ    32+4+4+4+4+6+16;[32...];; Task minimum stack size
                                                          ↑―― タスク用スタックサイズ (b)-③
52 行目 .section         h3hstack,stack,align=2
53 行目          .res.b  ( 4) +TSKSTKSIZ ;[32...];: tskid1 stack area
54 行目TSK1_SP: .equ     $
55 行目          .res.b  8  ← 共有スタック用領域 (a)-①
56 行目          .res.b  ( 4) +TSKSTKSIZ ;[32...];: tskid2 stack area
57 行目TSK2_SP: .equ     $
58 行目          .res.b  8  ← 共有スタック用領域 (a)-①
```

図Ｄ―１　セットアップテーブルの変更箇所（１／２）

```
59 行目          .res.b  (32) +TSKSTKSIZ ;[32...];: tskid3 stack area
60 行目TSK3_SP: .equ    $
61 行目          .res.b  8  ← 共有ｽﾀｯｸ用領域 (a)-①
62 行目          .res.b  (32) +TSKSTKSIZ ;[32...];: tskid4 stack area
63 行目TSK4_SP: .equ    $
64 行目          .res.b  8  ← 共有ｽﾀｯｸ用領域 (a)-①
                                        .
                                        .
81 行目tdt_id3: .data.b NOEXS, 3                ;: init. mode, init. priority
82 行目          .data.l 0, TSK3_SP             ;: top address, stack pointer
83 行目 tdt_id4:.data.b NOEXS, 4                ;: init. mode, init. priority
84 行目          .data.l 0, TSK4_SP             ;: top address, stack pointer
85 行目tdt_id5:.data.b NOEXS, 5                 ;: init. mode, init. priority   ← 共有ｽﾀｯｸ領域
86 行目          .data.l 0, TSK4_SP             ;: top address, stack pointer      の使用例 (a)-②
                                                                                    (ﾀｽｸID=4と同一
                                                                                     のｽﾀｯｸ領域)
                                        .
                                        .
133行目 mpldt_id4:      .data.w MB4_CNT, MB4_LEN        ;: Memory block count, length
134行目                 .data.l MPL4_TOP                ;: Memorypool top address
135行目 MPLDT_BTM:
136行目 MPLCNT:         .equ    (MPLDT_BTM-MPLDT_TOP)/MPLDTLEN
137行目                                 ;:[0...255]     ;: Memorypool definition count
138行目 ;
139行目 ;------ Usage ----------------------------------------------------------
140行目 ;TRC_CNT:.equ   TRACE COUNT                     ;: COMMENT
141行目 ;TRC_BUF:.equ   TRACE BUFFER ADDRESS            ;: COMMENT
142行目 ;------------------------------------------------------------------------
143行目          .section        h3htrc.data.align=2
144行目 TRC_CNT:        .equ    4                       ;: trace count          ← 取得する
145行目 TRC_BUF:        .res.b  16+(TRC_CNT*24)         ;: trace buffer address    情報数
                                                                                   ﾊﾞｯﾌｧ領域
                                                                                   (b)-②
146行目 ;
147行目 ;------ Usage ----------------------------------------------------------
148行目 ;INITRC         .data.l TRACE BUFFER ADDRESS    ;: COMMENT
149行目 ;               .data.w TRACE COUNT             ;: COMMENT
150行目 ;------------------------------------------------------------------------
151行目          .section        h3hsetup.code.align=2
152行目 INITRC:         .equ    $                       ;:                      ← ﾄﾚｰｽﾊﾞｯﾌｧ
153行目                 .data.l TRC_BUF                 ;: trace buffer address    情報ﾃｰﾌﾞﾙ
154行目                 .data.w TRC_CNT                 ;: trace count             (b)-②
155行目 ;
156行目          .include        "setup.cmn"
157行目 ;
158行目 ;*********************************************;
159行目          .end; of H3HSETUP.MAR
```

図D—1　セットアップテーブルの変更箇所（２／２）

(3)オブジェクトモジュール、実行形式ロードモジュールの作成

(1)ファイルの差し替え、(2)セットアップテーブルの変更後、オブジェクトモジュール、実行形式ロードモジュールの作成を行ないます。

オブジェクトモジュール、実行形式ロードモジュールの作成方法については、「２．１０　実行形式ロードモジュールの作成」を参照してください。

付録E．ＡＳＣＩＩコード表

E．1　ＡＳＣＩＩコード表

上位４ビット / 下位４ビット	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

付録Ｆ．索引

Ｆ．１　五十音順・索引

五十音順・索引

ア　行

カ　行

サ　行

タ　行

ナ　行

ハ　行

マ　行

ヤ　行

ラ　行

ワ　行

アルファベット順・索引

A

B

C

E

F

H

I

M

O

R

S

T

UNIX HI8-3H
構築マニュアル

ルネサス エレクトロニクス株式会社
神奈川県川崎市中原区下沼部1753 〒211-8668

ADJ-702-149A