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

SH7000シリーズ
HI-SHデバッガ

ユーザーズマニュアル

ルネサスマイクロコンピュータ開発環境システム

HS0700IRCN1S

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　1994.03

はじめに

本マニュアルは、「SH7000シリーズ　HI-SH7デバッガ」について解説しています。

「SH7000シリーズ　HI-SH7デバッガ」（以下、デバッガと略します）は、リアルタイム・マルチタスクOSであるHI-SH7を使用したユーザシステムの開発を支援するソフトウェアです。
本デバッガは、ユーザプログラム、HI-SH7と共にユーザシステムに組み込み、ユーザシステム上で動作します。

なお、HI-SH7の詳しい説明は、『HI-SH7ユーザーズマニュアル』、および『HI-SH7構築マニュアル』を参照してください。

＜マニュアルの構成＞

第1章では、デバッガの概要について説明しています。
第2章では、デバッガの動作環境について説明しています。
第3章では、デバッガのコマンドについて詳細に解説しています。
第4章では、デバッガのシステムの構築ついて説明しています。
付録では、メモリ容量の算出、システムの構築例、応答メッセージ、ユーザシステムのインタフェース回路例、ＡＳＣＩＩコード表、および索引を記載しています。

＜関連マニュアル＞

・HI-SH7ユーザーズマニュアル
・HI-SH7構築マニュアル
・SH7000シリーズ　ハードウェアマニュアル
・SH7000シリーズ　Cコンパイラ　ユーザーズマニュアル
・SH7000シリーズ　クロスアセンブラ　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　インタフェースソフト　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＞

[ ]	省略できることを示します。
{ }	いずれかひとつを選択することを示します。
(RET)	リターンキーの入力を示します。
△	1つ以上の空白またはタブコードを示します。
___	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。
< >	この記号で囲まれた内容を指定することを示します。
…	直前の項目を1回以上指定することを示します。
H'	整数定数の先頭に" H' "が付いているのは16進数です。
D'	整数定数の先頭に" D' "が付いているのは10進数です。

目　次

＜第４章　システムの構築＞

＜付録Ａ　メモリ容量の算出＞

＜付録Ｂ　システムの構築例＞

＜付録Ｃ　応答メッセージ＞

＜付録Ｄ　ユーザシステムのインタフェース回路例＞

＜付録Ｅ　ＡＳＣＩＩコード表＞

＜付録Ｆ　索引＞

図目次

表目次

1．　概　要

SH7000シリーズHI-SH7デバッガ（以下デバッガと略します）は、リアルタイムマルチタスクOS : HI-SH7を使用したユーザシステムの開発を支援するソフトウェア（デバッグツール）です。

本デバッガは、HI-SH7カーネル、およびユーザプログラムと共にユーザシステムに組み込み、ユーザシステム上で動作します。

ユーザは、端末とユーザシステムをシリアル回線で接続し、端末からコマンドを入力することで、CPUの制御、メモリの操作などのほか、HI-SH7カーネルに対してタスクの起動やイベントフラグのセット、タスクやセマフォ等の状態を参照することができます。これによって、HI-SH7を利用したユーザシステムの開発を効率良く行なうことができます。

図１－１にデバッガの概要を示します。

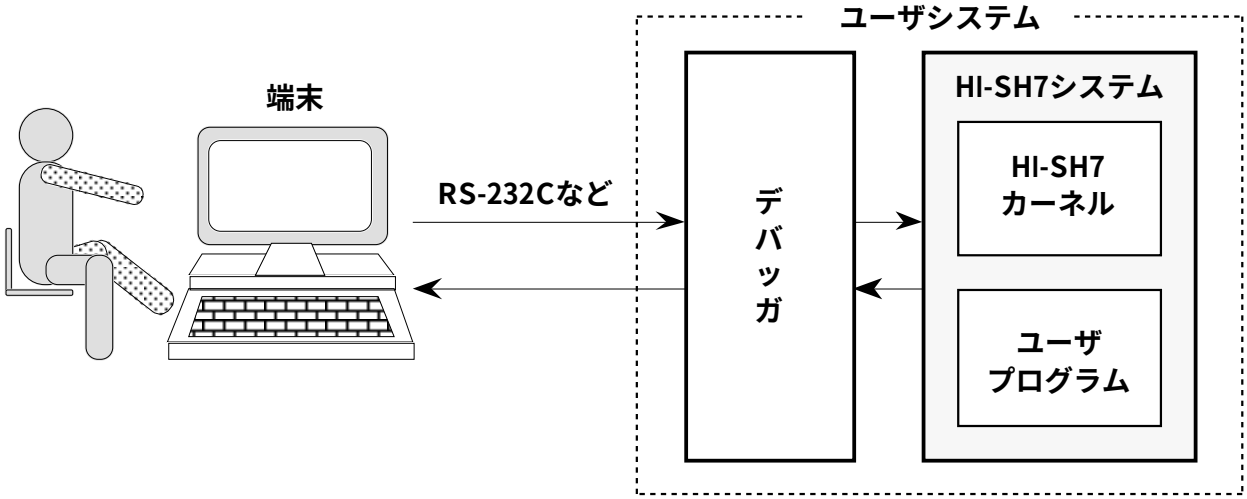


図１－１　デバッガの概要

１．１　特　長

(1)　組込み型デバッガ

本デバッガは、ユーザシステム上に組み込んで動作させるため、インサーキットエミュレータなどのハードウェアを必要としません。

(2)　機能選択可能なデバッガ

ユーザが必要とするデバッグ機能をセットアップテーブルで選択し、最適なデバッガを構築することができます。これにより、デバッガを組み込むために必要なユーザシステムのROM/RAM容量を小さくすることができます。

(3)　リアルタイムマルチタスク環境におけるデバッグの実現

・HI-SH7システムが動作中でも、デバッグコマンドを入力することができます。
・HI-SH7カーネルの各オブジェクトの状態を参照することができます。
・HI-SH7カーネルに対して、各種システムコールを要求することができます。
・各タスクが保有しているレジスタを表示／変更することができます。
・ブレークポイントの設定条件として、タスクを指定することができます。

２．　動作環境

２．１　システム構成

図２－１にシステム構成を示します。

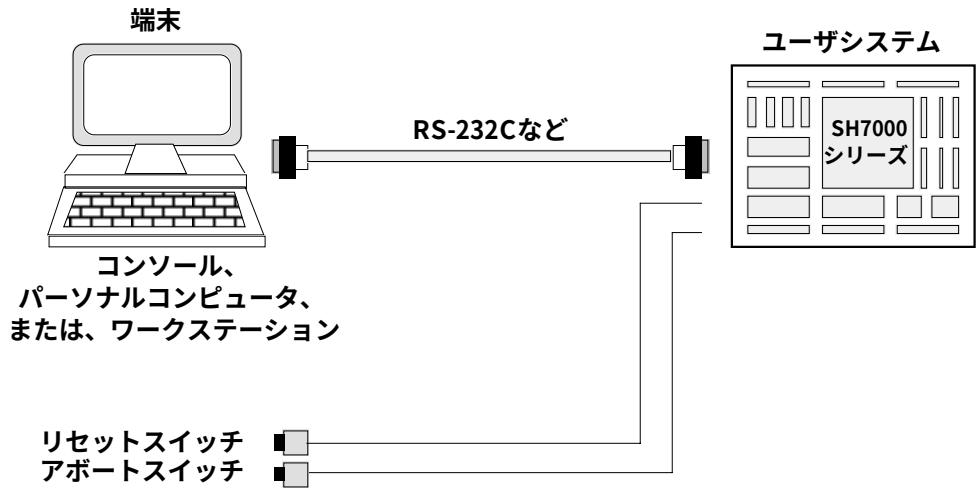


図２－１　システム構成

コマンドを入出力するための端末と、SH7000シリーズマイコンを搭載したユーザシステムが必要です。端末とユーザシステムは、RS-232Cなどのシリアル回線で接続します。

また、ユーザシステムが暴走した際に、リセットまたはアボート（ユーザプログラムの強制中断）したい場合には、リセットスイッチまたはアボートスイッチが必要です。

端末として使用する機器には、ASCIIコードのシリアル送受信が行なえるコンソール、パーソナルコンピュータ、またはワークステーションが必要です。

本デバッガのホスト機能（ファイルのロード／セーブ／メモリとのベリファイ）を使用する場合には、Hシリーズインタフェースソフトが動作するパーソナルコンピュータ、またはワークステーションが必要です。詳細は、『**Hシリーズインタフェースソフト　ユーザーズマニュアル**』を参照してください。

【注意】

本デバッガをユーザシステムに搭載（ダウンロード）する場合には、インサーキットエミュレータ等が必要です。または、本デバッガをROMに書き込み、ユーザシステムに搭載します。

２．２　ハードウェア構成

図２－２にハードウェア構成を示します。本デバッガは、SHマイコンに内蔵されているSCIとUBCを使用して動作します。周辺回路には、シリアル回線ドライバ／レシーバ、チャタリング防止回路、および外部ROM、外部RAMが必要となります。

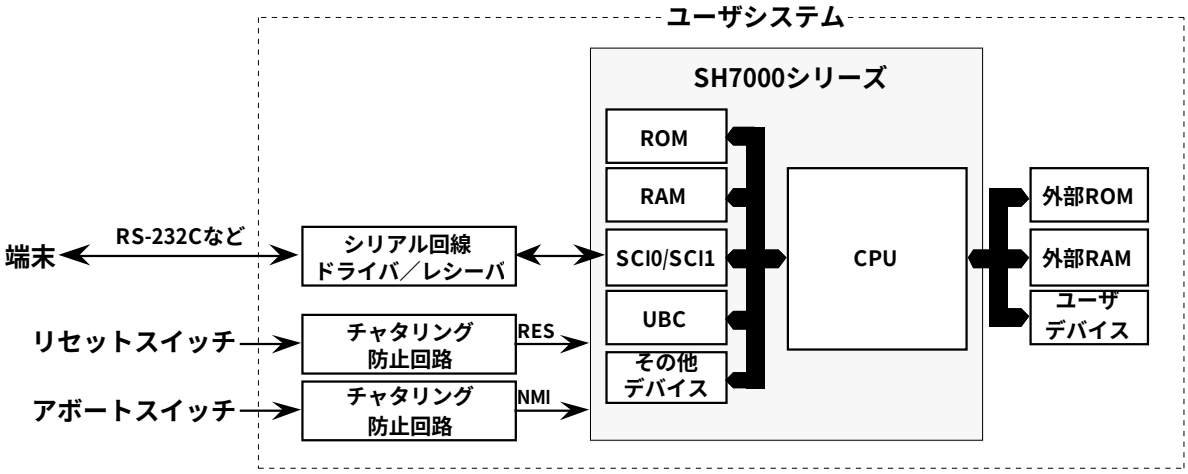


図２－２　ハードウェア構成

(1)　シリアル回線ドライバ／レシーバ

端末とユーザシステムをRS-232Cなどで接続するには、シリアル回線の仕様に合った電圧レベルに変換する回路（シリアル回線ドライバ／レシーバ）が必要です。**「付録Ｄ　ユーザシステムのインタフェース回路例」**を参照してください。

(2)　チャタリング防止回路

リセットスイッチ（RESET割込み）とアボートスイッチ（NMI割込み）は、チャタリング防止回路を介して、SHマイコンに入力する必要があります。**「付録Ｄ　ユーザシステムのインタフェース回路例」**を参照してください。

(3)　ROM/RAM

ユーザシステムに本デバッガを組み込むと、SHマイコンの内蔵ROM、内蔵RAMだけでは、メモリ容量が不足します。ユーザシステムに、外部ROM、外部RAMを用意してください。必要メモリ容量の詳細は、**「付録A　メモリ容量の算出」**を参照してください。

なお、システム構築時に、使用するコマンドを選択することで、デバッガのメモリ容量を小さくすることができます。詳細は、**「４．３　デバッガセットアップテーブルの作成」**を参照してください。

(4)　SCI

本デバッガは、SHマイコン内蔵のSCI（シリアルコミュニケーションインタフェース）を使用して、コマンドの入出力（シリアル回線の送受信）を行ないます。デバッガが使用するデバイス（SCI0またはSCI1）は、システム構築時に選択することができます。詳細は、**「４．３．７　コンソールドライバ情報の定義」**を参照してください。なお、選択したSCIは、ユーザプログラムから使用することはできなくなります。

(5)　UBC

本デバッガは、SHマイコン内蔵のUBC（ユーザブレークコントローラ）を使用して、ハードウェアブレークを実現しています。

(6)　その他デバイス

本デバッガは、SHマイコン内蔵のSCI、UBC以外のデバイスは、一切使用していません。

２．３　ソフトウェア構成

２．３．１　オフラインモードとオンラインモード

本デバッガは、オフラインモードとオンラインモードの2つのモードから、コマンドを入力、実行することができます。

オフラインモードとは、ユーザプログラムが停止している状態のことです。このモードでは、デバッガ（オフラインデバッガ）がCPUを占有するため、実行コマンド（GO/STEP/STEP_OVER/STEP_UBC）が入力されるまで、ユーザプログラムは動作しません。

オンラインモードとは、ユーザプログラムが実行している状態（マルチタスク環境）のことです。このモードでは、デバッガ（オンラインデバッガタスク）がタスクとして動作するため、ユーザタスクは並列に動作します。

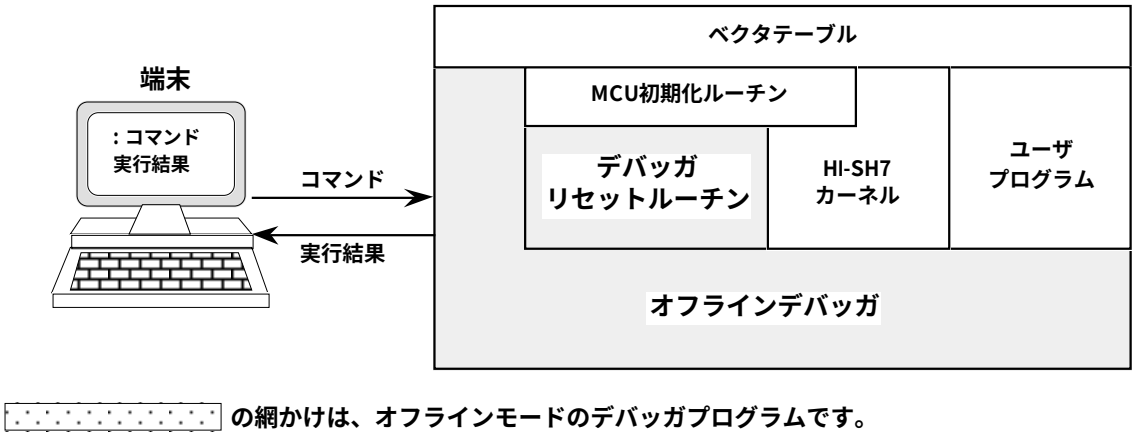
なお、デバッガは、オフラインモード／オンラインモードのプロンプト（コマンド入力促進記号）として、以下の文字を表示します。

・オフラインモード　':'　（オフラインデバッガが表示するプロンプトです）
・オンラインモード　'#'　（オンラインデバッガタスクが表示するプロンプトです）

次に、オフラインモードとオンラインモードのソフトウェア構成を示します。

２．３．２　オフラインモードのソフトウェア構成

図２－３にオフラインモードのソフトウェア構成を示します。



の網かけは、オフラインモードのデバッガプログラムです。

図２－３　オフラインモードのソフトウェア構成

(1)　デバッガリセットルーチン

デバッガリセットルーチンは、CPUリセットで起動するMCU初期化ルーチンからコールし、デバッガの作業領域と、コマンド入出力用のSCIを初期化します。

その後、オフラインデバッガを起動、またはMCU初期化ルーチンへリターンします（HI-SH7カーネルおよびユーザプログラムを実行します）。

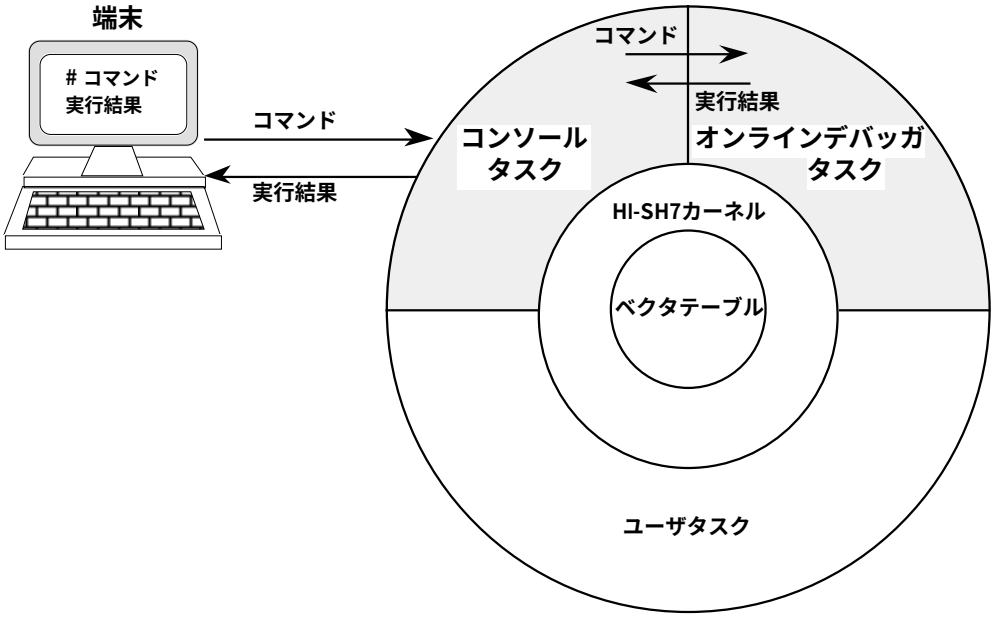
(2)　オフラインデバッガ

オフラインデバッガは、端末からコマンドを入力して、そのコマンドを実行します。また、コマンドの実行結果を端末に出力します。

なお、オフラインデバッガは、CPUの割込みマスクレベルを最大値（15）に設定します。したがって、RESETとNMI以外の割込みはすべてマスクされています。

２．３．３　オンラインモードのソフトウェア構成

図２－４にオンラインモードのソフトウェア構成を示します。



の網かけは、オンラインモードのデバッガプログラムです。

図２－４　オンラインモードのソフトウェア構成

(1)　コンソールタスク[*1]

コンソールタスクは、端末からコマンドを入力して、そのコマンドをオンラインデバッガタスクへ送信します。また、オンラインデバッガタスクから実行結果を受信して、その実行結果を端末に出力します。

コンソールタスクとオンラインデバッガタスク間は、ＯＳの資源であるメイルボックスを利用してメッセージ（コマンドと実行結果）を送受信します。

なお、コンソールタスクは、端末との入出力、およびオンラインデバッガタスクとの送受信を行なう必要があるときのみ実行（RUN）状態となり、これ以外のときは待ち（WAIT）状態となります。したがって、コンソールタスクはユーザタスクと並列に動作します。

(2)　オンラインデバッガタスク

オンラインデバッガタスクは、コンソールタスクからコマンドを受信して、そのコマンドを実行します。また、コマンドの実行結果をコンソールタスクへ送信します。

なお、オンラインデバッガタスクは、コマンドの実行、およびコンソールタスクとの送受信を行なう必要があるときのみ実行（RUN）状態となり、これ以外のときは待ち（WAIT）状態となります。したがって、オンラインデバッガタスクはユーザタスクと並列に動作します。

【注】*1　本デバッガ専用のコンソールドライバ（コンソールタスク）であり、HI-SH7付属のコンソールドライバとは関係ありません。

２．４　シリアルインタフェース仕様

表２－１に端末とユーザシステム間のシリアルインタフェース仕様を示します。

表２－１　端末とユーザシステム間のシリアルインタフェース仕様

項番	項　目	仕　様
1	使用するデバイス	SCI0またはSCI1
2	転送速度*1	110 / 150 / 300 / 600 / 1200 / 2400 / 4800 / 9600 / 19200 / 31250 / 38400 bps
3	データ長	8ビット
4	パリティ	なし
5	スタートビット	1ビット
6	ストップビット	1ビット
7	制御方式	XON/XOFF制御

【注】*1　デバッガのホスト機能（ファイルのロード／セーブ／メモリとのベリファイ）を使用する場合には、9600bpsの転送速度を使用してください。

なお、使用するデバイス（SCI0/SCI1）、および転送速度（bps）は、共通ヘッダファイルで定義します。詳細は、**「４．３．７　コンソールドライバ情報の定義」**を参照してください。

２．５　割込み／例外の環境

本デバッガは、SHマイコンの割込み／例外機能を利用して動作します。デバッガが使用する割込み／例外は、ユーザプログラムからは使用することができなくなります（RESET割込みを除く）。

次に、割込み／例外の環境を表わすベクタテーブルの内容について説明します。

２．５．１　ベクタテーブルの内容

図２－５にベクタテーブルの内容を示します。

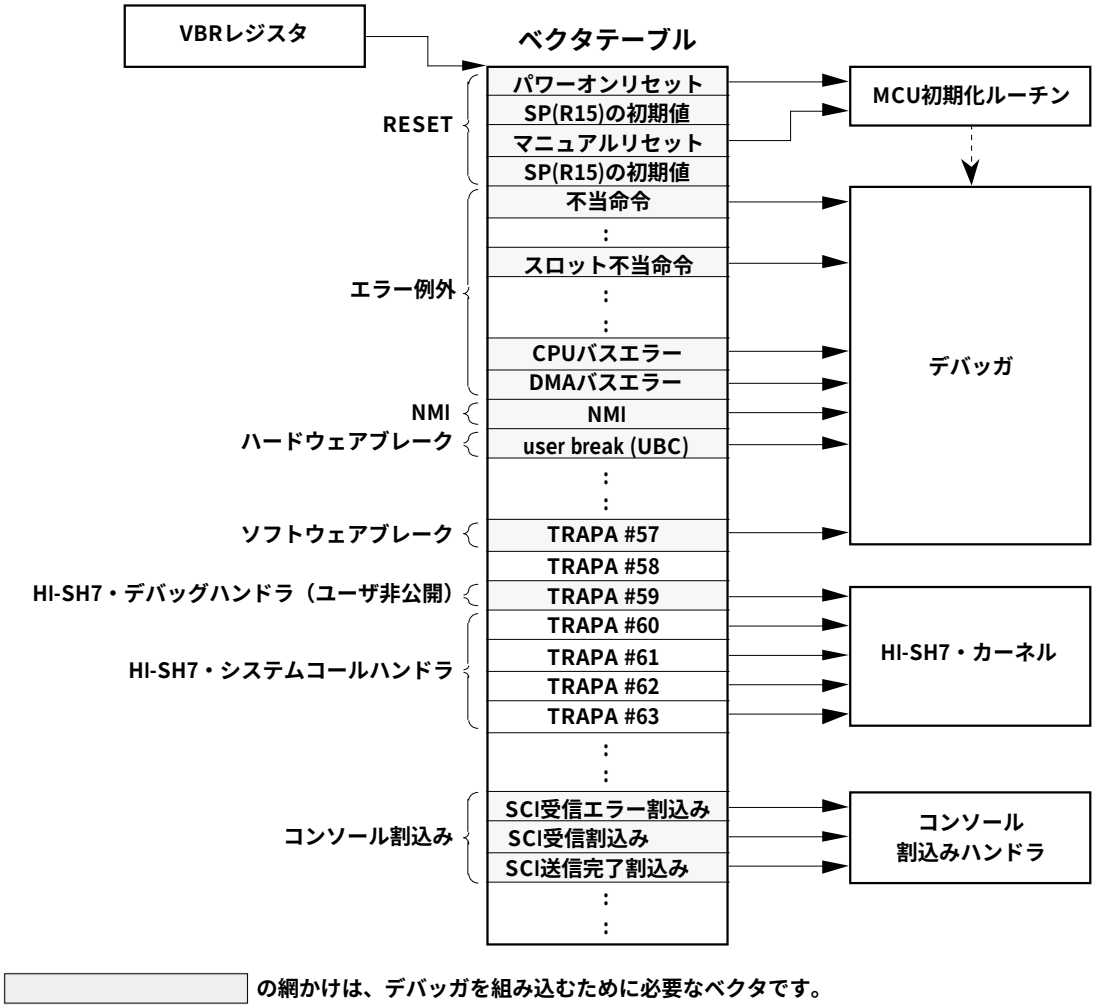


図２－５　ベクタテーブルの内容

(1)　RESET

パワーオンリセット、またはマニュアルリセットにより、MCU初期化ルーチンを起動します。なお、MCU初期化ルーチンからは、デバッガリセットルーチンをコールします。

(2)　エラー例外

システムにエラー（不当命令／スロット不当命令／CPUバスエラー／DMAバスエラー）が発生したとき、デバッガを起動します。なお、エラー例外に対する処理プログラムを起動したい場合は、ユーザプログラムで使用してもかまいません。

(3)　NMI

NMI割込みが発生したとき、デバッガを起動します。なお、アボート機能（ユーザプログラムの強制中断）を必要としない場合は、ユーザプログラムで使用してもかまいません。

(4)　ハードウェアブレーク

本デバッガは、ハードウェアブレークを実現するため、UBC（ユーザブレークコントローラ）の割込み機能を使用します。

(5)　ソフトウェアブレーク

本デバッガは、ソフトウェアブレークを実現するため、TRAPA #57 命令例外を使用します。したがって、ユーザプログラムでは、TRAPA #57 命令を使用することはできまん。

(6)　HI-SH7・デバッグハンドラ

本デバッガは、タスクやトレースに対するコマンドを実現するため、HI-SH7のデバッグハンドラ（ユーザ非公開）を使用します。

(7)　コンソール割込み

本デバッガは、オンラインモードにおけるコマンド入出力を行なうため、SCIの送受信割込みを使用します。

２．６　リセット時の動作

本デバッガは、CPUリセット時に、デバッガに対するリセット処理が必要となります。

次に、CPUリセット時に起動、実行するMCU初期化ルーチン、およびデバッガリセットルーチンについて説明します。

２．６．１　MCU初期化ルーチンの内容

図２－６にMCU初期化ルーチンの内容を示します。

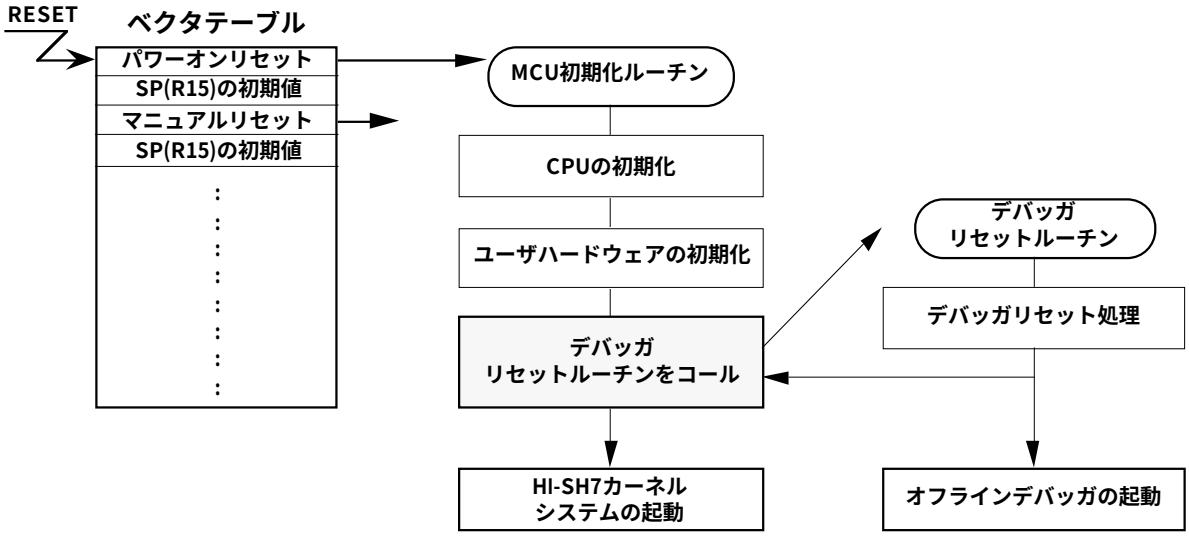


図２－６　MCU初期化ルーチンの内容

MCU初期化ルーチンは、パワーオンリセット、またはマニュアルリセットにより起動します。

HI-SH7システムのMCU初期化ルーチンでは、CPUの初期化、ユーザハードウェアの初期化の後、HI-SH7カーネルのシステム起動を行なっています。

本デバッガを組み込むには、HI-SH7カーネルのシステム起動を行なう前に、デバッガリセットルーチンをコールする必要があります。

デバッガリセットルーチンは、デバッガに対するリセット処理を行なった後、オフラインデバッガを起動するか、またはMCU初期化ルーチンにリターンします。

２．６．２　デバッガリセットルーチンの内容

図２－７にデバッガリセットルーチンの内容を示します。

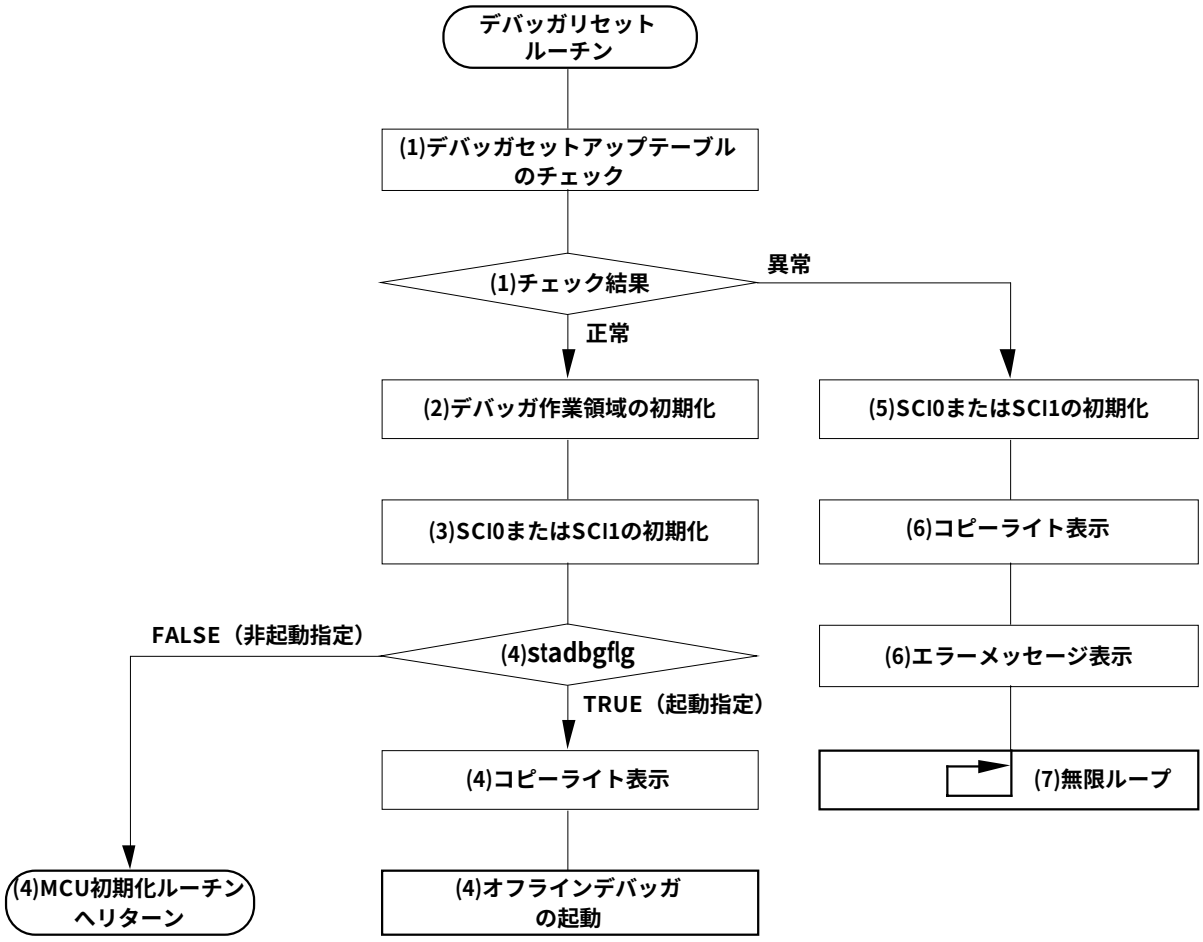


図２－７　デバッガリセットルーチンの内容

(1)　デバッガセットアップテーブルのチェック

デバッガセットアップテーブルの定義内容をチェックします。定義内容が正しければ、(2)～(4)の処理を行ないます。誤っていた場合には、(5)～(7)の処理を行ないます。

(2)　デバッガ作業領域の初期化

デバッガセットアップテーブルの定義内容にもとづいて、デバッガが使用する作業領域を初期化します。

(3)　SCIの初期化（チェック結果が正常の場合）

デバッガセットアップテーブルの定義内容にもとづいて、コマンド入出力を行なうためのSCI（SCI0またはSCI1）を初期化します。

(4)　オフラインデバッガの起動／MCU初期化ルーチンへのリターン

デバッガセットアップテーブルに定義するオフラインデバッガ起動フラグ（stadbgflg）の内容によって、次の処理を行ないます。

・TRUE（起動指定）　　：　本製品のコピーライトを表示し、オフラインデバッガを起動します。
・FALSE（非起動指定）　：　MCU初期化ルーチンへリターンします。

(5)　SCIの初期化（チェック結果が異常の場合）

デバッガセットアップテーブルの定義内容にもとづいて、(6)でメッセージ表示するためのSCI（SCI0またはSCI1）を初期化します。ただし、コンソールドライバに関する定義内容が誤っていた場合は、次の条件でSCIを初期化します。

・SCIチャンネル　：　SCIチャンネルの定義（SCI0またはSCI1）が誤っていた場合には、SCI1を初期化します。

・シリアル転送速度　：　シリアル転送速度の定義が誤っていた場合には、9600bpsでSCIを初期化します。また、CPUクロックの定義が誤っていた場合には、20MHzとしてSCIを初期化します。（SCIのシリアル転送速度は、CPUクロックによって変わります）

(6)　コピーライト表示とエラーメッセージ表示

本製品のコピーライトと、デバッガセットアップテーブルのエラーメッセージを表示します。エラーメッセージについての詳細は、**「付録Ｃ．２　デバッガセットアップテーブルチェックのエラーメッセージ」**を参照してください。

(7)　無限ループ

割込みをすべてマスク（RESET/NMI割込みを除く）した状態で無限ループします。RESET割込みを発生させると、(5)～(7)の処理が再度行なわれます。NMI割込みを発生させた場合、その後の動作は保証されません。

なお、デバッガセットアップテーブルの定義内容についての詳細は、**「４．３　デバッガセットアップテーブルの作成」**を参照してください。

２．７　オフラインデバッガの動作

図２－８にオフラインデバッガの動作を示します。

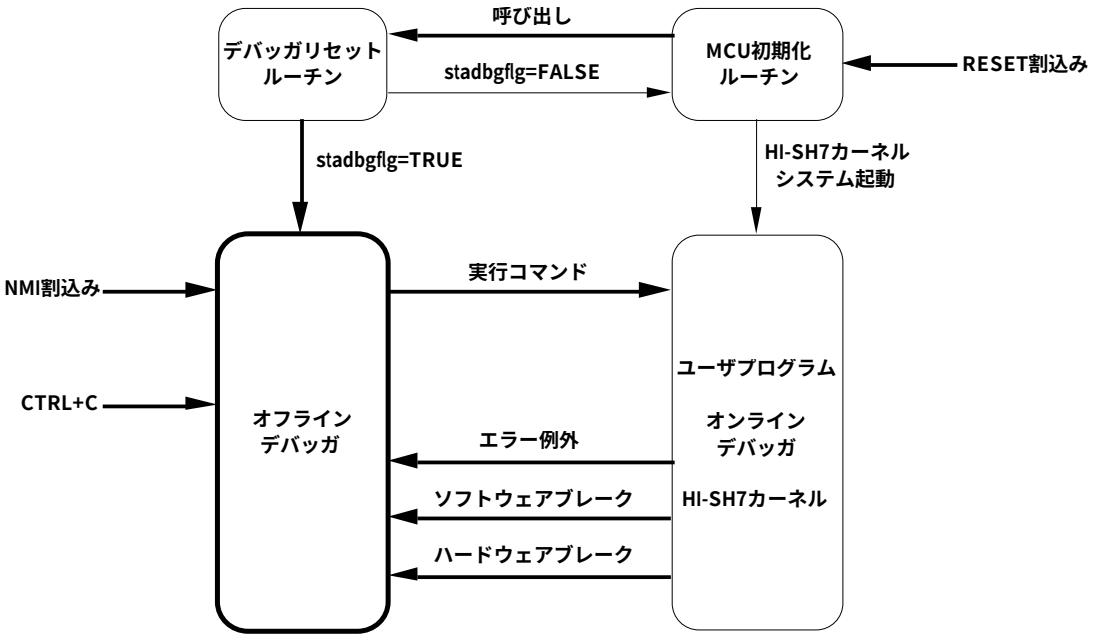


図２－８　オフラインデバッガの動作

(1)　オフラインデバッガの起動

オフラインデバッガは、次のとき起動します。

・RESET割込み

　パワーオンリセット、またはマニュアルリセットが発生すると、デバッガセットアップテーブルのオフラインデバッガ起動フラグ（stadbgflg）が起動指定（TRUE）であれば、MCU初期化ルーチン、デバッガリセットルーチンを実行後、オフラインデバッガが起動します。

・NMI割込み

　NMI割込みが発生すると、オフラインデバッガが起動します。なお、ユーザプログラム実行中にNMI割込みが発生したときのみ、オフラインデバッガが起動します。

・エラー例外（不当命令／スロット不当命令／CPUバスエラー／DMAバスエラー）

　不当命令、スロット不当命令、CPUバスエラー、およびDMAバスエラーが発生すると、オフラインデバッガが起動します。

・ソフトウェアブレーク／ハードウェアブレーク

　ハードウェアブレーク、およびソフトウェアブレークの条件が成立すると、オフラインデバッガが起動します。

・CTRL+C（コンソール割込み）

　ユーザプログラム実行中（オンライン状態）に、端末からCTRL+Cを入力すると、オフラインデバッガが起動します。ただし、オンラインデバッガタスクが終了しているときのみ、オフラインデバッガが起動します。

(2)　オフラインデバッガの終了

オフラインデバッガは、次のとき終了します。

・実行コマンド（GO/STEP/STEP_OVER/STEP_UBC）

　実行コマンドを入力すると、オフラインデバッガは終了し、ユーザプログラム、オンラインデバッガ、またはHI-SH7カーネルを実行します（オンライン状態になります）。

２．８　オンラインデバッガの動作

２．８．１　オンラインデバッガのプログラム構成

図２－９にオンラインデバッガのプログラム構成を示します。
オンラインデバッガは、オンラインデバッガタスク、コンソールタスク、およびコンソール割込みハンドラから構成されます。

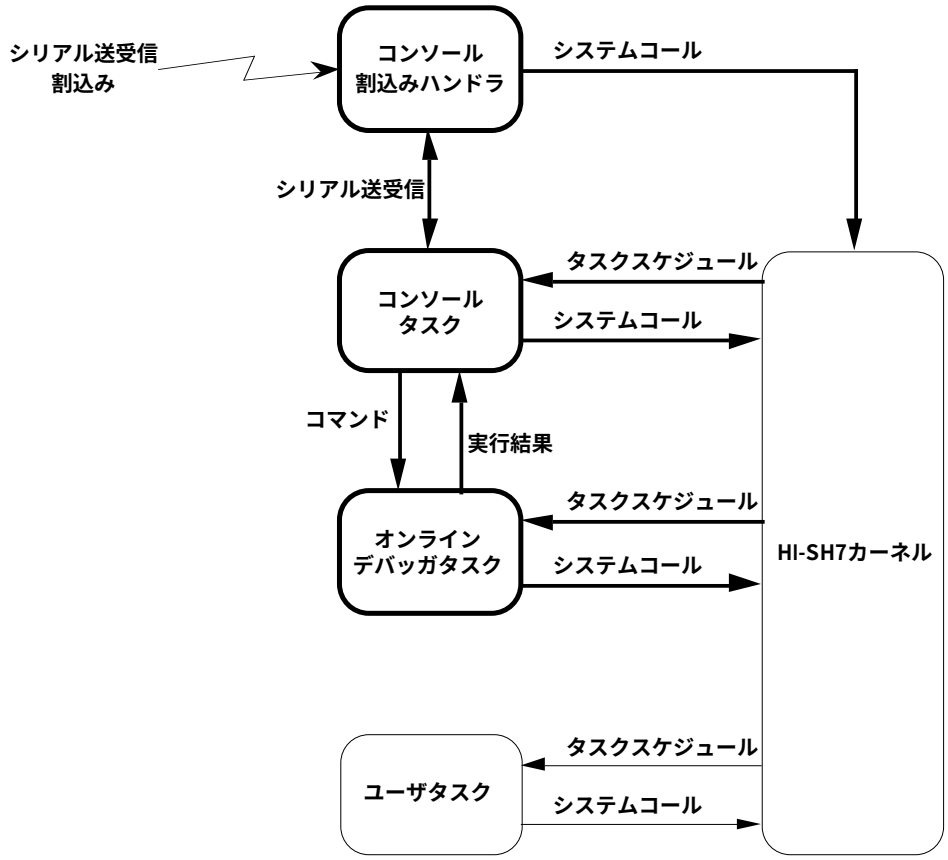


図２－９　オンラインデバッガのプログラム構成

(1)　オンラインデバッガタスクとコンソールタスク

オンラインデバッガタスクとコンソールタスクは、メイルボックスを利用してコマンドと実行結果のメッセージ送受信を行なうため、以下のシステムコールを発行[*1]します。

・オンラインデバッガタスク　：　snd_msgシステムコール、rcv_msgシステムコール
・コンソールタスク　　：　rcv_msgシステムコール、snd_msgシステムコール

(2)　コンソールタスクとコンソール割込みハンドラ

コンソールタスクとコンソール割込みハンドラは、割込みを利用してシリアル送受信を行なうため、以下のシステムコールを発行[*1]します。

・コンソールタスク　　：　slp_taskシステムコール
・コンソール割込みハンドラ　：　iwup_taskシステムコール、ret_intシステムコール

以上のように、オンラインデバッガは、タスクと割込みハンドラを利用しているため、ユーザタスクと並列に動作します。

【注】*1　発行したシステムコールは、HI-SH7のトレースバッファに履歴として格納されます。

２．８．２　オンラインデバッガタスクとコンソールタスクの生成

図２－１０に示すように、オンラインデバッガタスクとコンソールタスクは、HI-SH7セットアップテーブルに初期登録して、HI-SH7カーネルのシステム起動時に生成します。詳細は、**「４．４　HI-SH7セットアップテーブルの修正」**を参照してください。

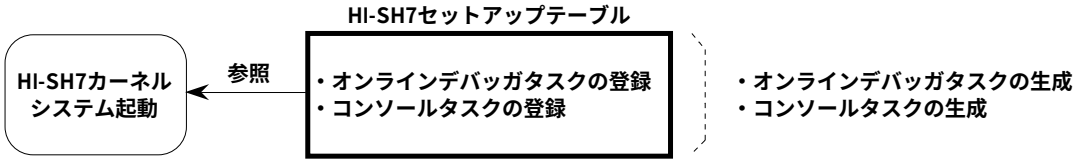


図２－１０　オンラインデバッガタスクとコンソールタスクの生成

２．８．３　オンラインデバッガタスクとコンソールタスクの起動

図２－１１に示すように、オンラインデバッガタスクとコンソールタスクは、システム初期化ハンドラからista_tskシステムコールを発行して起動します。詳細は、**「４．５．３　システム初期化ハンドラの修正」**を参照してください。

また、端末からCTRL+Dを入力すると、コンソール割込みハンドラからista_tskシステムコールが発行され、オンラインデバッガタスクが起動します（オンラインデバッガタスクを終了した後、再起動する場合に利用します）。

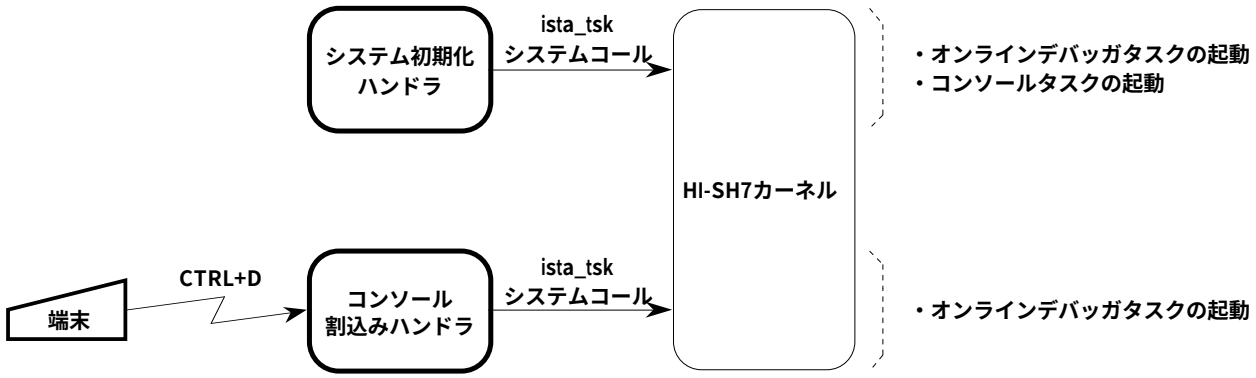


図２－１１　オンラインデバッガタスクとコンソールタスクの起動

２．８．４　オンラインデバッガタスクの終了

図２－１２に示すように、オンラインデバッガタスクは、CTRL+Z(RET)を入力すると、ext_tskシテムコールを発行して終了します。

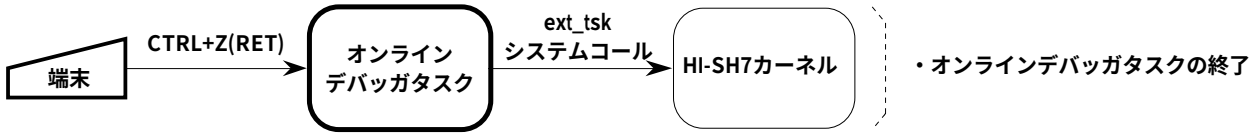


図２－１２　オンラインデバッガタスクの終了

２．８．５　オンラインデバッガが使用するOS資源

表２－２にオンラインデバッガが使用するOS資源を示します。

表２－２　オンラインデバッガが使用するOS資源

項番	OS資源	使用数	用　途
1	タスク	２ケ	オンラインデバッガタスクとコンソールタスク
2	メイルボックス	２ケ	コマンドと実行結果のメッセージ送受信

なお、使用するタスクID、メイルボックスIDは、共通ヘッダファイルで定義します。詳細は、**「４．２．５　デバッガが使用するタスクID、メイルボックスIDの定義」**を参照してください。

２．９　オフラインデバッガとオンラインデバッガの動作

２．９．１　オフラインデバッガとオンラインデバッガの操作例

次にオフラインデバッガとオンラインデバッガの操作例を示します。

```
A>INTFC(RET)                                    ←ホストコンピュータのHシリーズインタフェースソフトを起動します。
 H-SERIES INTERFACE (型名) Ver n.m
 Copyright(c) Hitachi,LTD. 19xx
 Licensed Material of Hitachi,LTD.
                                  ←ユーザシステムの電源投入により、コピーライト表示後、オフラインデバッガが起動します。
 SH7000 HI-SH7 DEBUGGER (HS0700IRCN1SMB) Ver n.m
 Copyright (C) Hitachi, Ltd. 1994
 Licensed Material of Hitachi, Ltd.
:(RET)                                          ←オフラインコマンドを入力することができます。
:LOAD TEST.MOT;S(RET)                           ←ユーザプログラムをロードしています。
 TOP     ADDRESS = 02000000
 BOTTOM ADDRESS = 020003DF
 LOAD NORMAL END
:BREAK 02000000;1(RET)                          ←ブレークポイントを設定しています。
:BREAK(RET)
 <ADDRESS>     <TASK> <CONT> <PASS>
 02000000        0001    0001    0000
:GO(RET)                                        ←実行コマンドを入力すると、オンラインデバッガが起動します。
#(RET)                                          ←オンラインコマンドを入力することができます。
#CRE_TSK 1 02000000 2(RET)                      ←ユーザタスクを生成しています。
0001=E_OK
#STA_TSK 1(RET)                                 ←ユーザタスクを起動しています。
0001=E_OK
#
TSKID=0001                                      ←ブレークが発生すると、オフラインデバッガに戻ります。
 PC=02000000 SR=00000000:----------
 PR=00000000 GBR=00000000 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 BREAK POINT=02000000
:
```

２．９．２　オフラインモードとオンラインモードの状態遷移

図２－１３にオフラインモードとオンラインモードの状態遷移を示します。

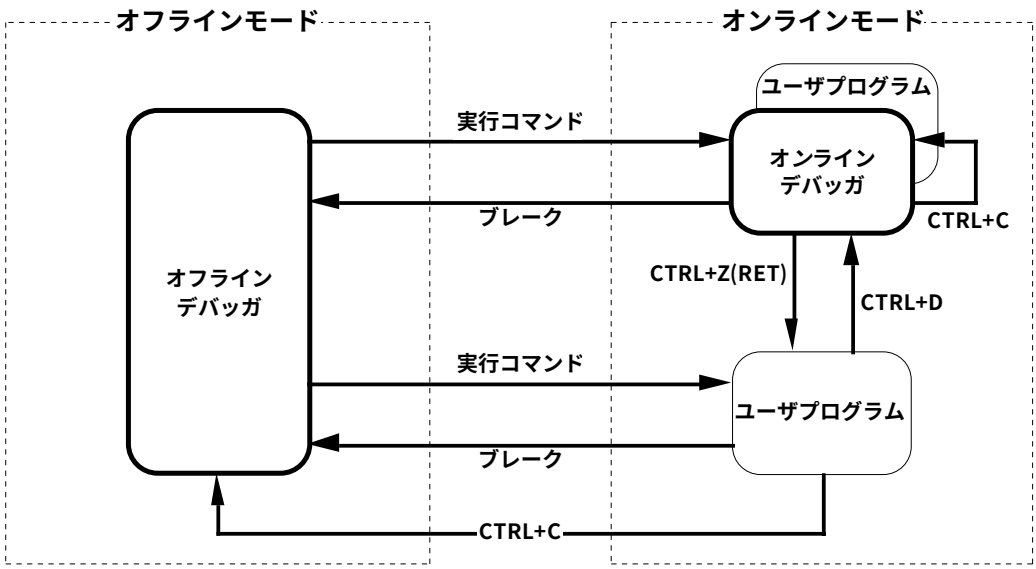


図２－１３　オフラインモードとオンラインモードの状態遷移

(1)　オフラインモードからオンラインモードへの状態遷移

　オフラインモードのとき、実行コマンド（GO/STEP/STEP_OVER/STEP_UBC）を入力すると、オンラインモードに状態が遷移します。

(2)　オンラインモードからオフラインモードへの状態遷移

　オンラインモードのとき、ブレーク（RESET/NMI/エラー例外/ソフトウェアブレーク/ハードウェアブレーク）が発生すると、オフラインモードに状態が遷移します。また、ユーザプログラムのみ実行している状態のとき、CTRL+Cを入力[*1]するとオフラインモードに状態が遷移します。

　なお、オンラインデバッガは、ユーザプログラム実行状態でCTRL+Dの入力により起動し、CTRL+Z(RET)の入力により終了します。

　また、オンラインデバッガ起動後に、CTRL+Cを入力すると、オンラインデバッガは一端終了した後、再起動し直します。

【注】*1　シリアル受信割込み（割込みレベル=14）がマスクされていると、CTRL+Cを入力することはできません。この場合は、NMI割込みを使用してください。

２．９．３　特殊キーの使用方法

表２－３に特殊キーの使用方法を示します。

表２－３　特殊キーの使用方法

項番	特殊キー	状　態	動　　作
1	CTRL+C	コマンド入力中	入力中のコマンドラインをキャンセルします
		コマンド実行結果表示中	実行中のコマンドを中断します
		オンライン状態	ユーザプログラムのみ実行している状態のとき、オフラインデバッガを起動します
2	CTRL+D	オンライン状態	オンラインデバッガを起動します
3	CTRL+S	コマンド実行結果表示中	コマンド実行結果の表示を一時、停止します
4	CTRL+Q	コマンド実行結果一時停止中	コマンド実行結果の表示を再開します
5	CTRL+X	コマンド入力中	入力中のコマンドラインをキャンセルします
6	BS, DEL	コマンド入力中	入力中のコマンドラインを1文字戻します
7	CTRL+Z	オンライン状態	CTRL+Z(RET)を入力すると、オンラインデバッガを終了します

３．　デバッグコマンド

３．１　デバッグ機能の概要

図３－１にデバッグ機能の構成を示します。

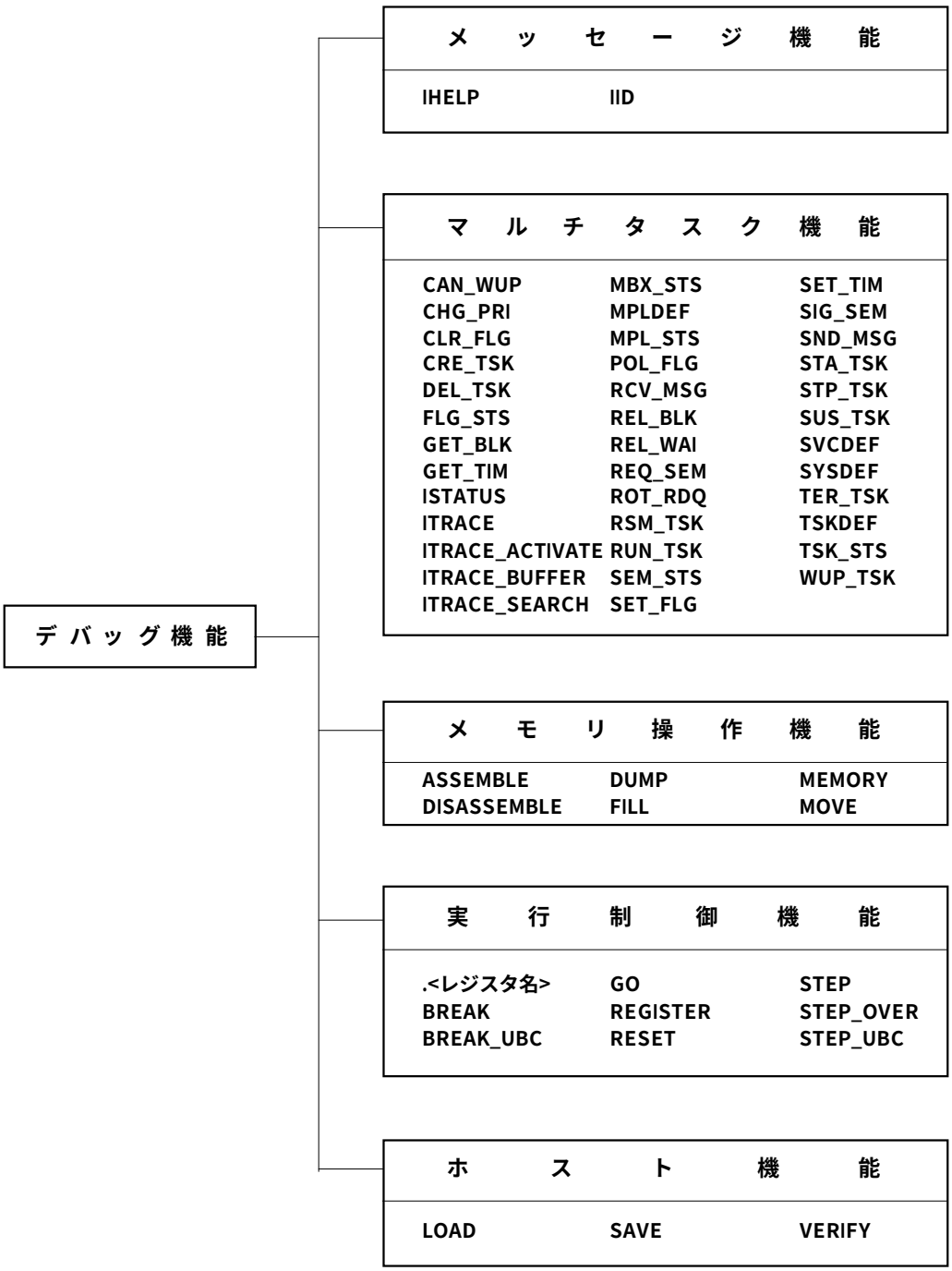


図３－１　デバッグ機能の構成

表３－１にデバッグ機能の概要を示します。

表３－１　デバッグ機能の概要

項番	機　　能	概　　要
1	メッセージ機能	・コマンドヘルプ、バージョンを表示します
2	マルチタスク機能	・HI-SH7の構築定義情報を表示します ・HI-SH7のタスク、イベントフラグ等のオブジェクトを操作します ・HI-SH7のトレースバッファを操作します
3	メモリ操作機能	・メモリの表示／変更／初期化／転送を行ないます ・１行アセンブル／逆アセンブルを行ないます
4	実行制御機能	・レジスタの表示／変更を行ないます（タスク指定可） ・ブレークポイントの設定／表示／解除を行います（タスク指定可） ・プログラムの実行／シングルステップ実行を行ないます
5	ホスト機能	・ファイルのロード／セーブ／メモリとのベリファイを行ないます

３．２　デバッグコマンド一覧

表３－２にデバッグコマンド一覧を示します。

表３－２　デバッグコマンド一覧（１／２）

	コマンド	省略形	機　　能	使用可能モード	記載頁
メッセージ機能					
1.	IHELP	IHE	ﾃﾞﾊﾞｯｶﾞのｺﾏﾝﾄﾞﾍﾙﾌﾟﾒｯｾｰｼﾞの表示	ON/OFF	3-40
2.	IID	IID	デバッガのバージョン表示	ON/OFF	3-42
マルチタスク機能					
3.	CAN_WUP	CAN	タスクの起床要求の無効	ON	3-17
4.	CHG_PRI	CHG	タスク優先度の変更	ON	3-19
5.	CLR_FLG	CLR	イベントフラグのクリア	ON	3-21
6.	CRE_TSK	CRE	タスクの生成	ON	3-23
7.	DEL_TSK	DEL	タスクの削除	ON	3-25
8.	FLG_STS	FLG	イベントフラグ状態の表示	ON/OFF	3-33
9.	GET_BLK	GETB	固定長メモリブロックの獲得	ON	3-35
10.	GET_TIM	GETT	HI-SH7システムクロックの表示	ON/OFF	3-37
11.	ISTATUS	IST	指定状態タスクの表示	ON/OFF	3-43
12.	ITRACE	ITR	HI-SH7トレース情報の表示	ON/OFF	3-45
13.	ITRACE_ACTIVATE	ITA	HI-SH7トレース取得の開始／停止／表示	ON/OFF	3-49
14.	ITRACE_BUFFER	ITB	HI-SH7トレースバッファの設定／表示	ON/OFF	3-51
15.	ITRACE_SEARCH	ITS	HI-SH7トレース情報の検索／表示	ON/OFF	3-53
16.	MBX_STS	MBX	メイルボックス状態の表示	ON/OFF	3-59
17.	MPLDEF	MPLD	メモリプール定義情報の表示	ON/OFF	3-65
18.	MPL_STS	MPL	メモリプール状態の表示	ON/OFF	3-67
19.	POL_FLG	POL	イベントフラグを得る	ON	3-69
20.	RCV_MSG	RCV	メイルボックスからの受信	ON	3-71
21.	REL_BLK	RELB	固定長メモリブロックの返却	ON	3-75
22.	REL_WAI	RELW	タスクの待ち状態強制解除	ON	3-77
23.	REQ_SEM	REQ	セマフォ資源を得る	ON	3-79
24.	ROT_RDQ	ROT	タスクのレディーキューを回転する	ON	3-82
25.	RSM_TSK	RSM	強制待ち状態タスクの再開	ON	3-83
26.	RUN_TSK	RUN	タスクのデバッグ待ち状態の解除	ON	3-85
27.	SEM_STS	SEM	セマフォ状態の表示	ON/OFF	3-89
28.	SET_FLG	SETF	イベントフラグのセット	ON	3-91
29.	SET_TIM	SETT	HI-SH7システムクロックの設定	ON	3-93
30.	SIG_SEM	SIG	セマフォに対する信号操作（V命令）	ON	3-94
31.	SND_MSG	SND	メイルボックスへの送信	ON	3-96
32.	STA_TSK	STA	タスクの起動	ON	3-98
33.	STP_TSK	STP	タスクのデバッグ待ち状態への遷移	ON	3-107
34.	SUS_TSK	SUS	タスクの強制待ち状態への遷移	ON	3-110
35.	SVCDEF	SVCD	HI-SH7システムコール定義情報の表示	ON/OFF	3-112
36.	SYSDEF	SYSD	HI-SH7システム定義情報の表示	ON/OFF	3-115
37.	TER_TSK	TER	タスクを強制的に異常終了させる	ON	3-116
38.	TSKDEF	TSKD	タスク定義情報の表示	ON/OFF	3-118
39.	TSK_STS	TSK	タスク状態の表示	ON/OFF	3-120
40.	WUP_TSK	WUP	タスクの起床	ON	3-126

表３－２　デバッグコマンド一覧（２／２）

	コマンド	省略形	機　　能	使用可能モード	記載頁
メモリ操作機能					
41.	ASSEMBLE	A	１行アセンブル	OFF	3-9
42.	DISASSEMBLE	DA	メモリ内容の逆アセンブル表示	ON/OFF	3-27
43.	DUMP	D	メモリのダンプ表示	ON/OFF	3-29
44.	FILL	F	メモリへのデータ書き込み	ON/OFF	3-31
45.	MEMORY	M	メモリ内容の表示、変更	ON/OFF	3-61
46.	MOVE	MV	メモリ転送	ON/OFF	3-63
実行制御機能					
47.	.<レジスタ名>		レジスタ内容の表示、変更	ON/OFF	3-6
48.	BREAK	B	ｿﾌﾄｳｪｱﾌﾞﾚｰｸﾎﾟｲﾝﾄの設定／表示／解除	ON/OFF	3-11
49.	BREAK_UBC	BU	ﾊｰﾄﾞｳｪｱﾌﾞﾚｰｸﾎﾟｲﾝﾄの設定／表示／解除	ON/OFF	3-14
50.	GO	G	プログラムの実行	OFF	3-38
51.	REGISTER	R	全レジスタ内容の表示	ON/OFF	3-73
52.	RESET	RS	CPUのリセット	OFF	3-81
53.	STEP	S	シングルステップ実行	OFF	3-100
54.	STEP_OVER	SO	サブルーチンステップ実行	OFF	3-102
55.	STEP_UBC	SU	ﾊｰﾄﾞｳｪｱﾌﾞﾚｰｸによるｼﾝｸﾞﾙｽﾃｯﾌﾟ実行	OFF	3-104
ホスト機能					
56.	LOAD	L	ファイルからのロード	OFF	3-57
57.	SAVE	SV	ファイルへのセーブ	OFF	3-87
58.	VERIFY	V	ファイルとメモリのベリファイ	OFF	3-124

【注】　OFF　：　オフラインモード
　　ON　：　オンラインモード

３．３　デバッグコマンドの機能

本節は、図３－２に示す形式でデバッグコマンドの機能を説明しています。

		コマンド名
項番号	**コマンド名**	**コマンド内容**
	コマンド省略形	
		使用可能モード

【コマンドフォーマット】

【説　　明】

【応答メッセージ】

【注意事項】

【関連コマンド】

【使　用　例】

・コマンド名　　：　コマンドの名称です。
・コマンド省略形：　コマンド名の省略形です。
・コマンド内容　：　当該コマンドの内容です。
・使用可能モード：
　　コマンドが使用可能なモードを示します。

・コマンドフォーマット：
　　コマンドの入力形式を示します。

・説明：　コマンドの機能や使用方法を示します。

・応答メッセージ：　コマンドの処理結果を表示するメッセージを示します。

・注意事項：　当該コマンドを使用する上での注意事項です。特に注意事項のないコマンドの場合は省略してあります。

・関連コマンド：　関連のあるコマンドを示します。

・使用例：　当該コマンドの使用例を載せています。

図３－２　デバッグコマンドの説明形式

コマンドフォーマットの説明に使用している記号は、次のような意味を持っています。

[　]	：	[　]で囲まれた部分のパラメータは省略可能を示します。
{｜}	：	{｜}で囲まれた部分の複数のパラメータのうち、１つを選択して指定することを示します。
<　>	：	<　>内に示した内容を指定または表示することを示します。
. . .	：	直前の指定を繰り返し入力できることを示します。
△	：	スペースキーの入力を示します。コマンドフォーマットのみ使用します。
(RET)	：	リターンキーの入力を示します。
____	：	____の下線は入力部分を示します。コマンドフォーマットでは使用しません。

なお、パラメータの指定方法として、次のような指定が可能です。

<パラメータ１>,<パラメータ２>...	：	パラメータを複数指定することを意味します。
<パラメータ１>:<パラメータ２>	：	パラメータ１からパラメータ２までの範囲を指定することを意味します。

３．３．１	．<レジスタ名>	
	．<レジスタ名>	レジスタ内容の表示、変更

オフライン／オンライン

【コマンドフォーマット】

.<レジスタ名>[△<データ>](RET)　　：　CPU指定（オフラインのみ）
.<レジスタ名>[△<データ>];<タスクID>(RET)　　：　タスク指定（オンラインのみ）

<レジスタ名>　：　変更する制御レジスタ名または汎用レジスタ名
（制御レジスタ）　PC, SR[*1], PR, GBR, VBR[*1], MACH, MACL
（汎用レジスタ）　R0, R1, R2, R3, R4, R5, R6, R7,
R8, R9, R10, R11, R12, R13, R14, R15[*1]
<データ>　：　変更するレジスタ値
<タスクID>　：　対象タスクID

【注】*1　CPU指定のときのみ指定できます

【説　　明】

レジスタ内容を変更します。本コマンドは、CPU指定する場合にはオフラインモード、タスク指定する場合にはオンラインモードで、それぞれ実行することができます。

(1)　直接

・CPU指定（オフラインのみ）

指定した値でCPUのレジスタ内容を変更します。

:.<レジスタ名>△<データ>(RET)

・タスク指定（オンラインのみ）

指定した値でタスクのレジスタ内容を変更します。

#.<レジスタ名>△<データ>;<タスクID>(RET)

(2)　対話

・CPU指定（オフラインのみ）

現在のCPUのレジスタ値を表示し、変更値を要求します。この処理をPC, SR, PR, GBR, VBR, MACH, MACL, R0～R15のレジスタ順序で行ないます。

:.<レジスタ名>(RET)
<レジスタ名> =aaaaaaaa ? <サブコマンド>(RET)
<レジスタ名> =aaaaaaaa ? <サブコマンド>(RET)

・タスク指定（オンラインのみ）

現在の対象タスクのレジスタ値を表示し、変更値を要求します。この処理をPC, PR, GBR, MACH, MACL, R0～R14のレジスタ順序で行ないます。なお、SR, VBR, R15レジスタは指定できません。

```
#.<レジスタ名>;<タスクID>(RET)
TSKID=bbbb
 <レジスタ名> =aaaaaaaa ? <サブコマンド>(RET)
 <レジスタ名> =aaaaaaaa ? <サブコマンド>(RET)
```

aaaaaaaa　：　現在のレジスタ値
bbbb　：　対象タスクID
<サブコマンド>　：　サブコマンド（表３－３に示す内容を入力します）

表３－３　レジスタ内容の変更、表示のサブコマンド

項番	サブコマンド	意　味
1	<データ>	表示レジスタを指定したデータに変更します
2	(RET)のみ	レジスタ値を変更せずに、次のレジスタ内容を表示します
3	^	レジスタ値を変更せずに、直前のレジスタ内容を表示します
4	.	コマンドを終了します

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー ・オフラインモードのときタスク指定した ・オンラインモードのときCPU指定した
2	*** I08:INVALID REGISTER	レジスタ不正 ・タスク指定のときSR, VBRまたはR15を指定した
3	*** I32:DEBUGGER USE ID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	*** I37:TASK IS NON EXISTENT	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
5	*** I38:TASK IS DORMANT	指定したタスクは休止（DORMANT）状態である
6	*** I39:TASK IS STACK WAIT	指定したタスクは共有スタック待ち状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【注意事項】

(1)　VBRレジスタに正しい値（ベクタベースアドレス）を設定していない場合、GO/STEP/STEP_OVER/STEP_UBCコマンド実行時のシステムの正常な動作は保証されません。

【関連コマンド】

REGISTER

【使　用　例】

(1)　直接

・CPU指定

PCにH'1000を設定します。

```
:.PC 1000(RET)
:
```

・タスク指定

タスク(ID=1)のR0レジスタにH'1234を設定します。

```
#.R0 1234;1(RET)
#
```

(2)　対話

・CPU指定

PCにH'1000、SRにH'F0を設定します。

```
:.PC(RET)
 PC    =00000400 ? 1000(RET)
 SR    =00000000 ? F0(RET)
 PR    =00000000 ? .(RET)
:
```

・タスク指定

タスク(ID=1)のR0レジスタにH'1234、R1レジスタにH'FFFFFFFFを設定します。

```
#.R0 ;1(RET)
TSKID=0001
 R0    =00000000 ? 1234(RET)
 R1    =11111111 ? FFFFFFFF(RET)
 R2    =02020202 ? .(RET)
#
```

3．3．2	ASSEMBLE	1 line Assemble
	A	１行アセンブル

オフライン

【コマンドフォーマット】

ASSEMBLE△<アドレス>[△<アセンブル文>](RET)

<アドレス>　：　機械語を書き込むアドレス
<アセンブル文>　：　SHマイコンのアセンブリ言語命令

【説　明】

(1) 直接

アセンブル文を機械語に変換し、指定したアドレスに書き込みます。

:ASSEMBLE△<アドレス>△<アセンブル文>(RET)

(2) 対話

指定したアドレスの逆アセンブルを表示し、サブコマンドモードになります。サブコマンドモードでアセンブル文を入力すると機械語に変換し、メモリに書き込みます。この動作を終了のサブコマンド入力まで繰り返します。

:ASSEMBLE△<アドレス>(RET)

```
  ADDR     CODE    MNEMONIC   OPERAND
aaaaaaaa   bbbb    逆アセンブル表示
aaaaaaaa         ? <サブコマンド>(RET)
```

aaaaaaaa　：　アドレス
bbbb　：　機械語コード
<サブコマンド>　：　サブコマンド（表３－４に示す内容を入力します）

表３－４　１行アセンブルのサブコマンド

項番	サブコマンド	意　味
1	<アセンブル文>	指定されたアセンブル文を機械語に変換し、表示アドレスのメモリに書き込みます
2	(RET)のみ	アドレスを＋２して、再度サブコマンド入力待ちになります
3	^	アドレスを－２して、再度サブコマンド入力待ちになります
4	.	ASSEMBLEコマンドを終了します

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I04:INVALID ADDRESS	アドレスエラー（アドレスが0または奇数である）
2	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
3	*** I13:NOT RAM AREA	アドレスエラー（アドレスがRAM領域でない）
4	*** I16:INVALID ASM MNEMONIC	アセンブル文の命令ニーモニック不正
5	*** I17:INVALID ASM OPERAND	アセンブル文の命令オペランド不正
6	*** I31:RESERVED ADDRESS AREA	アドレスエラー（アドレスがHI-SH7または本デバッガの領域である）

【関連コマンド】

DISASSEMBLE

【使　用　例】

(1)　直接

H'1000番地に"MOV R1,R2"の機械語を書き込みます。

```
:A 1000 MOV R1,R2(RET)
:
```

(2)　対話

H'1000番地から１行アセンブルを行ないます。

```
:A 1000(RET)
   ADDR     CODE    MNEMONIC   OPERAND
 00001000   0001    MOV        R0,R1
 00001000        ?  MOV        R1,R2(RET)
 00001002   1203    ADD        R2,R3
 00001002        ?  ADD        R3,R4(RET)
 00001004   2004    JMP        @R4
 00001004        ?  JMP        @R5(RET)
 00001006   0009    NOP
 00001006        ?  .(RET)
:
```

3．3．3	BREAK	Break point set, display, reset
	B	ソフトウェアブレークポイントの設定、表示、解除

オフライン／オンライン

【コマンドフォーマット】

(1) 設定

BREAK△<アドレス>[△<回数>](RET)　　：　CPU指定

BREAK△<アドレス>[△<回数>];<タスクID>(RET)　　：　タスク指定

(2) 表示

BREAK(RET)

(3) 解除

BREAK[△]－[<アドレス>](RET)

<アドレス>　　：　ブレークポイントのアドレス（RAM領域）

<回数>　　：　ブレークポイントの通過回数（1～H'FFFF）（省略時：1）

<タスクID>　　：　ブレークポイントの対象タスクID（H'0001～H'03FF）

【説　　明】

(1) 設定

指定したアドレス（RAM領域）にソフトウェアブレークポイントを設定します。ブレークポイントは1～32ケ所（セットアップテーブルのhd_breaknumに定義した値）まで設定することができます。

回数には1～H'FFFFの値を指定することができます。省略した場合は1が設定されます。

既に設定されているアドレスと同じアドレスに設定すると、現在の通過回数がクリアされます。

CPU指定の場合、既に設定されているアドレスと同じアドレスに設定すると、回数が更新され、タスクIDの指定は解除されます。

タスク指定の場合、既に設定されているアドレスと同じアドレスに設定すると、回数およびタスクIDが更新されます。

・CPU指定

:BREAK△<アドレス>[△<回数>](RET)

指定した回数だけブレークポイントを通過すると、プログラムの実行を停止します（ブレークポイントの指す命令を実行する直前）。その後、システムはオフラインのコマンド待ちになります。

・タスク指定

:BREAK△<アドレス>[△<回数>];<タスクID>(RET)

対象タスクは、指定した回数だけブレークポイントを通過すると、プログラムの実行を停止します（ブレークポイントの指す命令を実行する直前）。その後、システムはオフラインのコマンド待ちになります。

(2) 表示

設定したソフトウェアブレークポイントの内容を表示します。

```
:BREAK(RET)
<ADDRESS>     <TASK> <CONT> <PASS>
aaaaaaaa        bbbb    cccc    dddd
```

aaaaaaaa	：	アドレス
bbbb	：	タスクID（CPU指定のときは"----"を表示）
cccc	：	指定回数
dddd	：	通過回数（実際に通過した回数）

(3) 解除

・指定したアドレスのソフトウェアブレークポイントを解除します。

:BREAK-<アドレス>(RET)

・すべてのソフトウェアブレークポイントを解除します。

:BREAK-(RET)

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I04:INVALID ADDRESS	アドレスエラー ・アドレスが0または奇数である
2	*** I10:NOT FOUND	解除するブレークポイントがない
3	*** I11:COUNT ERROR	回数エラー（回数=0）
4	*** I13:NOT RAM AREA	アドレスエラー（アドレスがRAM領域でない）
5	*** I15:TOO MANY BREAK POINT	ブレークポイントの設定数が1～32[*1]を超えた
6	*** I31:RESERVED ADDRESS AREA	アドレスエラー（アドレスがHI-SH7または本デバッガの領域である）
7	*** I32:DEBUGGER USE ID	タスクIDエラー（指定したタスクは本デバッガで使用している）
8	*** I35:RESERVED ID	タスクIDエラー ・予約ID（タスクID=0）
9	*** I36:ID OVER	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*2]）
10	*** I50:MACHINE CODE IS DELAY BRANCH INSTRUCTION	アドレスエラー ・アドレスが遅延分岐命令の次命令アドレスである

【注】*1　ブレークポイントの最大設定数は、セットアップテーブルのhd_breaknumに定義した値です。

*2　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【注意事項】

(1) 本コマンドは、指定されたアドレスにソフトウェア例外命令を埋め込んで実現しているため、ROM領域を指定することはできません。ROM領域にブレークポイントを設定する場合は、BREAK_UBCコマンドを使用してください。

(2) 回数指定またはタスク指定を行なうと、ブレークポイント通過時にプログラムの実行を一時、停止するため、システムのリアルタイム性はなくなります。

【関連コマンド】

BREAK_UBC

【使　用　例】

(1)　設定

・CPU指定

ユーザプログラムがH'1000番地を通過した時に停止するように、ソフトウェアブレークポイントを設定します。

```
:B 1000(RET)
:
```

ユーザプログラムがH'2000番地を3回通過した時に停止するように、ソフトウェアブレークポイントを設定します。

```
:B 2000 3(RET)
:
```

・タスク指定

タスク(ID=1)がH'3000番地を通過した時に停止するように、ソフトウェアブレークポイントを設定します。

```
:B 3000;1(RET)
:
```

タスク(ID=2)がH'4000番地を5回通過した時に停止するように、ソフトウェアブレークポイントを設定します。

```
:B 4000 5;2(RET)
:
```

(2)　表示

設定したソフトウェアブレークポイントの内容を表示します。

```
:B(RET)
 <ADDRESS>     <TASK> <CONT> <PASS>
 00001000       ----   0001   0000
 00002000       ----   0003   0000
 00003000       0001   0001   0000
 00004000       0002   0005   0000
:
```

(3)　解除

・設定したH'1000番地のソフトウェアブレークポイントを解除します。

```
:B-1000(RET)
:
```

・設定したすべてのソフトウェアブレークポイントを解除します。

```
:B-(RET)
:
```

３．３．４	BREAK_UBC	Break by Ubc point set, display, reset
	BU	ハードウェアブレークポイントの設定、表示、解除

オフライン／オンライン

【コマンドフォーマット】

(1) 設定

BREAK_UBC△<アドレス>[△<回数>](RET)　　：　CPU指定

BREAK_UBC△<アドレス>[△<回数>];<タスクID>(RET)　　：　タスク指定

(2) 表示

BREAK_UBC(RET)

(3) 解除

BREAK_UBC[△]－(RET)

<アドレス>　　：　ブレークポイントのアドレス（ROM/RAM領域）

<回数>　　：　ブレークポイントの通過回数（1～H'FFFF）（省略時：1）

<タスクID>　　：　ブレークポイントの対象タスクID（H'0001～H'03FF）

【説　　明】

(1) 設定

指定したアドレス（ROM/RAM領域）にハードウェアブレークポイントを設定します。ブレークポイントは1ケ所だけ設定することができます。ブレークポイントを再設定すると、新しい設定値に更新します。

回数には1～H'FFFFの値を指定することができます。省略した場合は1が設定されます。

・CPU指定

:BREAK_UBC△<アドレス>[△<回数>](RET)

指定した回数だけブレークポイントを通過すると、プログラムの実行を停止します（ブレークポイントの指す命令をフェッチした直後）。その後、システムはオフラインのコマンド待ちになります。

・タスク指定

:BREAK_UBC△<アドレス>[△<回数>];<タスクID>(RET)

対象タスクは、指定した回数だけブレークポイントを通過すると、プログラムの実行を停止します（ブレークポイントの指す命令をフェッチした直後）。その後、システムはオフラインのコマンド待ちになります。

(2) 表示

設定したハードウェアブレークポイントの内容を表示します。

```
:BREAK_UBC(RET)
 <ADDRESS>      <TASK>  <CONT>  <PASS>
 aaaaaaaa         bbbb     cccc     dddd
```

aaaaaaaa	：	アドレス
bbbb	：	タスクID（CPU指定のときは"----"を表示）
cccc	：	指定回数
dddd	：	通過回数（実際に通過した回数）

(3) 解除

設定したハードウェアブレークポイントを解除します。

```
:BREAK_UBC-(RET)
```

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I04:INVALID ADDRESS	アドレスエラー ・アドレスが0または奇数である
2	*** I11:COUNT ERROR	回数エラー（回数=0）
3	*** I31:RESERVED ADDRESS AREA	アドレスエラー（アドレスがHI-SH7または本デバッガの領域である）
4	*** I32:DEBUGGER USE ID	タスクIDエラー（指定したタスクは本デバッガで使用している）
5	*** I35:RESERVED ID	タスクIDエラー ・予約ID（タスクID=0）
6	*** I36:ID OVER	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]）
6	*** I51:MACHINE CODE IS INTERRUPT MASK INSTRUCTION	アドレスエラー ・アドレスが割込み禁止命令の次命令アドレスです

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【注意事項】

(1) 本コマンドは、SHマイコン内蔵のUBC（ユーザブレークコントローラ）を使用して実現しています。UBCは、指定したブレークポイントの命令をCPUがフェッチすると、割込みレベル=15の割込みをCPUに対して要求します。したがって、以下の制限があります。

- 割込みレベル=15の割込み処理プログラム（割込みハンドラ）には、ブレークポイントを設定しないでください。
- 割込みを一時的に禁止する命令（LDC命令など）の次の命令にブレークポイントを設定することはできません。
- ブレークポイントの命令をCPUがフェッチした直後に、タイマなどの割込みが同時に発生すると、CPUの停止位置（PCレジスタ）は、その割込みの処理プログラム内を指していることがあります。この場合、ユーザプログラム実行前に、必要に応じて割込みをマスクしてください（.<レジスタ>コマンドでSRレジスタのIビットを1～14に変更）。

(2) 回数指定またはタスク指定を行なうと、ブレークポイント通過時にプログラムの実行を一時的に停止するため、システムのリアルタイム性はなくなります。

【関連コマンド】

BREAK

【使　用　例】

(1)　設定

・CPU指定

ユーザプログラムがH'1000番地を通過した時に停止するように、ハードウェアブレークポイントを設定します。

```
:BU 1000(RET)
:
```

ユーザプログラムがH'2000番地を3回通過した時に停止するように、ハードウェアブレークポイントを設定します。

```
:BU 2000 3(RET)
:
```

・タスク指定

タスク(ID=1)がH'3000番地を通過した時に停止するように、ハードウェアブレークポイントを設定します。

```
:BU 3000;1(RET)
:
```

タスク(ID=2)がH'4000番地を5回通過した時に停止するように、ハードウェアブレークポイントを設定します。

```
:BU 4000 5;2(RET)
:
```

(2)　表示

設定したハードウェアブレークポイントの内容を表示します。

```
:BU(RET)
 <ADDRESS>      <TASK> <CONT> <PASS>
 00004000         0002    0005    0000
:
```

(3)　解除

設定したハードウェアブレークポイントを解除します。

```
:BU-(RET)
:
```

３．３．５	CAN_WUP	CANcel wakeup task
	CAN	タスクの起床要求を無効にする

オンライン

【コマンドフォーマット】

CAN_WUP△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクにキューイングされていた起床要求回数を表示し、同時にその起床要求をすべて解除します。タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

ID=aaaa WUP=bb

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID
bb　　：　キューイングされていた起床要求回数

本コマンドはcan_wupシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_DMT	タスクが休止（DORMANT）状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

TSK_STS, WUP_TSK

【使　用　例】

(1)　タスクID=1のタスクの起床要求をすべて解除します。

```
#CAN_WUP 1(RET)
ID=0001 WUP=02
#
```

(2)　タスクID=1, 2, 5, 7, 9のタスクの起床要求をすべて解除します。

```
#CAN_WUP 1,2,5,7,9(RET)
ID=0001 WUP=00, ID=0002 WUP=03, ID=0005 WUP=00, ID=0007 WUP=0F,
ID=0009 WUP=00
#
```

3．3．6	CHG_PRI	CHanGe task priority
	CHG	タスク優先度を変更する

オンライン

【コマンドフォーマット】

CHG_PRI△<タスクID>△<タスク優先度>[,<タスクID>△<タスク優先度>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）
<タスク優先度>　：　変更タスク優先度（H'00～H'FF）

【説　　明】

指定したタスクの現在のタスク優先度を変更します。タスク優先度に0を指定すると、初期タスク優先度に戻します。パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドはchg_priシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID*1） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_TPRI	タスク優先度エラー（タスク優先度<0, タスク優先度>最大タスク優先度*2）
5	ED_DBGTPRI	タスク優先度エラー （タスク優先度=1：本デバッガで予約しています）
6	E_DMT	タスクが休止（DORMANT）状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。
*2　最大タスク優先度は、HI-SH7カーネルセットアップテーブルのhi_maxtskpriに定義した値です。

【関連コマンド】

TSK_STS, CRE_TSK

【使　用　例】

(1)　タスクID=1のタスク優先度を5に変更します。

```
#CHG_PRI 1 5(RET)
0001=E_OK
#
```

(2)　タスクID=1のタスク優先度を初期タスク優先度に戻し、タスクID=5のタスク優先度を7に変更します。

```
#CHG_PRI 1 0,5 7(RET)
0001=E_OK, 0005=E_OK
#
```

３．３．７	CLR_FLG	CLeaR eventflag
	CLR	イベントフラグをクリアする

オンライン

【コマンドフォーマット】

CLR_FLG△<イベントフラグID>△<ビットパターン>[,<イベントフラグID>△<ビットパターン>...](RET)

<イベントフラグID> ： 対象イベントフラグID（H'0001～H'03FF）
<ビットパターン> ： クリアするビットパターン（H'00000000～H'FFFFFFFF）

【説　明】

指定したイベントフラグに対して、ビットパターンの0になっているビットをクリアします。パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合
aaaa=E_OK

(2) 正常に処理されなかった場合
aaaa=<応答メッセージ>

aaaa ： イベントフラグID

本コマンドはclr_flgシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	*** I34:UNDEFINED OBJECT	イベントフラグを「未使用」でHI-SH7を構築している
3	E_NOEXS	イベントフラグIDエラー ・ID範囲外（イベントフラグID<0, イベントフラグID>最大イベントフラグID[*1]） ・予約ID（イベントフラグID=0）

【注】*1 最大イベントフラグIDは、HI-SH7カーネルセットアップテーブルのhi_maxflgidに定義した値です。

【関連コマンド】

FLG_STS, SET_FLG

【使　用　例】

(1)　イベントフラグID=H'000Aに対して、H'F0F0F0F0のビットパターンでイベントフラグをクリアします。

```
#CLR_FLG A F0F0F0F0(RET)
000A=E_OK
#
```

(2)　イベントフラグID=1, 2, 3に対して、それぞれH'F0, H'1111, H'FFFFFFFEのビットパターンでイベントフラグをクリアします。

```
#CLR_FLG 1 F0,2 1111,3 FFFFFFFE(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

３．３．８	CRE_TSK	CREate task
	CRE	タスクを生成する

オンライン

【コマンドフォーマット】

CRE_TSK△<タスクID>△<アドレス>△<タスク優先度>[,<タスクID>△<アドレス>△<タスク優先度>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）
<アドレス>　　：　生成するタスクの開始アドレス
<タスク優先度>　：　タスク優先度（H'02～H'FF）

【説　　明】

指定したタスクを生成します。パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合
　　aaaa=E_OK

(2)　正常に処理されなかった場合
　　aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドはcre_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_RSID	タスクIDエラー ・予約ID（タスクID=0）
3	E_IDOVR	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]）
4	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
5	E_ILADR	アドレスエラー（アドレスが0または奇数である）
6	ED_DBGADR	アドレスエラー（アドレスがHI-SH7または本デバッガの領域である）
7	E_TPRI	タスク優先度エラー（タスク優先度<0, タスク優先度>最大タスク優先度[*2]）
8	ED_DBGTPRI	タスク優先度エラー （タスク優先度=1：本デバッガで予約しています）
9	E_EXS	タスクがすでに存在している

【注】　*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。
　　*2　最大タスク優先度は、HI-SH7カーネルセットアップテーブルのhi_maxtskpriに定義した値です。

【関連コマンド】

DEL_TSK, TSKDEF, TSK_STS

【使　用　例】

(1)　タスクID=H'Bのタスクを、タスク開始アドレスをH'5000、初期タスク優先度を3として生成します。

```
#CRE_TSK B 5000 3(RET)
000B=E_OK
#
```

(2)　タスクID=H'10とH'20のタスクを、それぞれ次の条件で生成します。

・タスクID=H'10　　：　タスク開始アドレス=H'6000、初期タスク優先度=H'A
・タスクID=H'20　　：　タスク開始アドレス=H'6500、初期タスク優先度=H'F

```
#CRE_TSK 10 6000 A,20 6500 F(RET)
00010=E_OK, 0020=E_OK
#
```

3．3．9	DEL_TSK	DELete task
	DEL	タスクを削除する

オンライン

【コマンドフォーマット】

DEL_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクを削除します。タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドはdel_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_NODMT	タスクが休止（DORMANT）状態でない

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

CRE_TSK, TSKDEF, TSK_STS, TER_TSK

【使　用　例】

(1)　タスクID=H'Aのタスクを削除します。

```
#DEL_TSK A(RET)
000A=E_OK
#
```

(2)　タスクID=H'10, H'20, H'30, H'40, H'50のタスクを削除します。

```
#DEL_TSK 10,20,30,40,50(RET)
0010=E_OK, 0020=E_OK, 0030=E_OK, 0040=E_OK,
0050=E_OK
#
```

３．３．１０	DISASSEMBLE	DisAssemble
	DA	メモリ内容の逆アセンブル表示

オフライン／オンライン

【コマンドフォーマット】

DISASSEMBL E△<開始アドレス>[{△<終了アドレス> ¦ △@<命令数>}](RET)

<開始アドレス> ： 逆アセンブル開始アドレス
<終了アドレス> ： 逆アセンブル終了アドレス
<命令数> ： 逆アセンブルする命令数

【説　明】

指定したメモリ領域の逆アセンブルを行ない、アドレス、機械語、ニモニック、およびオペランドの表示を次のフォーマットで行ないます。

```
   ADDR    CODE   MNEMONIC   OPERAND
aaaaaaaa  bbbb   ccc...c    ddd...d
```

aaaaaaaa ： アドレス
bbbb ： 機械語コード
ccc...c ： ニモニック
ddd...d ： オペランド

(1) 終了アドレスまたは命令数を省略すると、16行分の逆アセンブルを行ないます。
(2) 該当する命令がない場合は次のように表示します。

```
   ADDR    CODE   MNEMONIC   OPERAND
aaaaaaaa  bbbb   .DATA.W    eeee
```

eeee ： bbbbと同じ値です

(3) 本コマンド終了後、他のコマンドを入力しなければ、(RET)キーの入力のみで直前に行なった逆アセンブル表示の次から16行分の逆アセンブル表示を行ないます。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I04:INVALID ADDRESS	アドレスエラー（アドレスが奇数である）

【関連コマンド】

ASSEMBLE

【使　用　例】

(1)　H'1000番地から16命令分を逆アセンブル表示します。また、(RET)のみでさらに16命令分を逆アセンブル表示します。

```
:DA 1000(RET)
   ADDR    CODE   MNEMONIC   OPERAND
 00001000  1F01   MOV.L      R0,@(4,R15)
 00001002  6673   MOV        R7,R6
 00001004  E001   MOV        #1,R0
 00001006  3708   SUB        R0,R7
 00001008  1F52   MOV.L      R5,@(8,R15)
 0000100A  1F43   MOV.L      R4,@(C,R15)
 0000100C  E00A   MOV        #0A,R0
 0000100E  6053   MOV        R5,R0
 00001010  1658   MOV.L      R5,@(20,R6)
 00001012  5568   MOV.L      @(20,R6),R5
 00001014  6053   MOV        R5,R0
 00001016  880A   CMP/EQ     #0A,R0
 00001018  8902   BT         00001020
 0000101A  E001   MOV        #01,R0
 0000101C  380C   ADD        R0,R8
 0000101E  0009   NOP
:(RET)
   ADDR    CODE   MNEMONIC   OPERAND
 00001020  2100   MOV.B      R0,@(4,R15)
 00001022  2201   MOV.W      R0,@R2
 00001024  2302   MOV.L      R0,@R3
                :
                :
:
```

(2)　H'1000番地からH'1004番地までを逆アセンブル表示します。

```
:DA 1000 1004(RET)
   ADDR    CODE   MNEMONIC   OPERAND
 00001000  1F01   MOV.L      R0,@(4,R15)
 00001002  6673   MOV        R7,R6
 00001004  E001   MOV        #1,R0
:
```

(3)　H'1000番地から3命令分を逆アセンブル表示します。

```
:DA 1000 @3(RET)
   ADDR    CODE   MNEMONIC   OPERAND
 00001000  1F01   MOV.L      R0,@(4,R15)
 00001002  6673   MOV        R7,R6
 00001004  E001   MOV        #1,R0
:
```

３．３．１１	DUMP	memory Dump
	D	メモリのダンプ表示

オフライン／オンライン

【コマンドフォーマット】

DUMP△<開始アドレス>[{△<終了アドレス> ¦ △@<バイト数>}] [;<表示単位>](RET)

<開始アドレス> ： 表示する開始アドレス
<終了アドレス> ： 表示する終了アドレス
<バイト数> ： 表示するバイト数
（省略時：256バイトの指定となります）
<表示単位> ： 表示するデータサイズ
B ： 1バイト単位
W ： 2バイト単位
L ： 4バイト単位
省略時 ： 1バイト単位

【説　明】

指定したメモリ領域のダンプ表示を行ないます。以下に表示フォーマットを示します。

(a) 1バイト単位

```
 <ADDR>                  <  D  A  T  A  >                  <ASCII CODE>
aaaaaaaa  bb bb bb bb bb bb bb bb  bb bb bb bb bb bb bb bb  cccccccccccccccc
```

(b) 2バイト単位

```
 <ADDR>                  <  D  A  T  A  >                  <ASCII CODE>
aaaaaaaa  bbbb  bbbb  bbbb  bbbb  bbbb  bbbb  bbbb  bbbb    cccccccccccccccc
```

(c) 4バイト単位

```
 <ADDR>                  <  D  A  T  A  >                  <ASCII CODE>
aaaaaaaa  bbbbbbbb    bbbbbbbb    bbbbbbbb    bbbbbbbb      cccccccccccccccc
```

aaaaaaaa ： メモリアドレスを表示します。
bbb...b ： メモリの内容を表示します。
ccc...c ： メモリの内容に該当するASCIIコードを表示します。
（該当するASCIIコードがない場合ピリオド'.'を表示）

(1) 開始アドレスは、表示単位が2バイトのとき2の倍数に、4バイトのとき4の倍数にそれぞれ切り上げます。

(2) 終了アドレスとバイト数は、表示単位が2バイトのとき2の倍数に、4バイトのとき4の倍数にそれぞれ切り捨てます。

(3) 本コマンド終了後、他のコマンドを入力しなければ、(RET)キーの入力のみで直前に行なったメモリダンプの次のアドレスから256バイト分表示します。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I04:INVALID ADDRESS	アドレスエラー ・開始アドレス＞終了アドレスである

【関連コマンド】

MEMORY

【使　用　例】

(1)　H'1000番地から256バイト分を1バイト単位で表示します。また、(RET)のみでさらに256バイト分を表示します。

```
:D 1000(RET)
  <ADDR>                    <  D  A  T  A  >                    <ASCII CODE>
 00001000   30 31 32 33 34 35 36 37  38 39 41 42 43 44 45 46  0123456789ABCDEF
 00001010   00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F  ................
 00001020   00 11 22 33 44 55 66 77  88 99 AA BB CC DD EE FF  ..3DUfw........
                                     :
                                     :
 000010F0   48 49 54 41 43 48 49 20  44 45 42 55 47 47 45 52  HITACHI DEBUGGER
:(RET)
  <ADDR>                    <  D  A  T  A  >                    <ASCII CODE>
 00001100   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
 00001110   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
 00001120   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
                                     :
                                     :
 000011F0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
:
```

(2)　H'1000番地からH'101Fまで2バイト単位で表示します。

```
:D 1000 101F;W(RET)
  <ADDR>                    <  D  A  T  A  >                    <ASCII CODE>
 00001000   3031  3233  3435  3637  3839  4142 4344  4546     0123456789ABCDEF
 00001010   0001  0203  0405  0607  0809  0A0B 0C0D  0E0F     ................
:
```

(3)　H'1000番地からH'20バイト分を4バイト単位で表示します。

```
:D 1000 @20;L(RET)
  <ADDR>                    <  D  A  T  A  >                    <ASCII CODE>
 00001000   30313233     34353637     38394142     43444546     0123456789ABCDEF
 00001010   00010203     04050607     08090A0B     0C0D0E0F     ................
:
```

３．３．１２	FILL	memory Fill
	F	メモリへのデータ書き込み

オフライン／オンライン

【コマンドフォーマット】

FILL△<開始アドレス>{△<終了アドレス> ｜ △@<バイト数>} [△<データ>] [;[<書き込み単位>] [△N]](RET)

<開始アドレス> ： 書き込み開始アドレス
<終了アドレス> ： 書き込み終了アドレス
<バイト数> ： 書き込みデータバイト数
<データ> ： 書き込みデータ（省略時：0）
<書き込み単位> ： 書き込むデータサイズ
B ： 1バイト単位
W ： 2バイト単位
L ： 4バイト単位
省略時 ： 1バイト単位
N ： ベリファイチェックなし

【説　　明】

指定したメモリ領域に指定データを書き込みます。

Nパラメータを指定しなかった場合は、データ書き込み後、正しくメモリに書き込まれたかをチェックするため、ベリファイを行ないます。エラー発生時は次のメッセージを表示して、コマンドを中断します。

```
FAILED AT  aaaaaaaa  WRITE=bb...'b...'  READ=cc...'c...'
```

aaaaaaaa ： エラーが発生したメモリアドレス
bb...'b...' ： 書き込みデータ（16進数とASCII表示）
cc...'c...' ： 読み込みデータ（16進数とASCII表示）

(1) 開始アドレスは、書き込み単位が2バイトのとき2の倍数に、4バイトのとき4の倍数にそれぞれ切り下げます。
(2) 終了アドレスとバイト数は、書き込み単位が2バイトのとき2の倍数に、4バイトのとき4の倍数にそれぞれ切り捨てます。
(3) 書き込むデータを省略すると0を書き込みます。
(4) Nパラメータを指定すると、メモリ書き込み後のベリファイを行ないません。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I31:RESERVED ADDRESS AREA	アドレスエラー（指定したメモリ領域がHI-SH7または本デバッガの領域である）

【関連コマンド】

MOVE

【使　用　例】

(1)　H'1000番地からH'1FFF番地までのメモリ領域に、H'00を書き込みます。

```
:F 1000 1FFF(RET)
:
```

(2)　H'1000番地からH'1FFF番地までのメモリ領域に、H'1234を書き込みます。

```
:F 1000 1FFF 1234;W(RET)
:
```

(3)　H'1000番地からH'1FFF番地までのメモリ領域に、H'12345678を書き込みます。

```
:F 1000 1FFF 12345678;L(RET)
:
```

(4)　H'1000番地からH'100バイト分のメモリ領域（H'1000～H'10FF）に、H'12を書き込みます（ベリファイなし）。

```
:F 1000 @100 12;B N(RET)
:
```

３．３．１３	FLG_STS	eventFLaG status
	FLG	イベントフラグの状態を表示する

オフライン／オンライン

【コマンドフォーマット】

FLG_STS[△<イベントフラグID>[{,<イベントフラグID>... ¦ :<イベントフラグID>}]](RET)

<イベントフラグID> ： 対象イベントフラグID（H'0001～H'03FF）

【説　　明】

指定したイベントフラグの状態を参照して、対象イベントフラグの現在のイベントフラグ値と、待ちタスクのIDを表示します。

イベントフラグIDは8個まで繰り返し指定することができます。イベントフラグIDを省略した場合、定義されているすべてのイベントフラグの状態を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

ID=aaaa FLGPTN=bbbbbbbb WTSK=cccc

(2) 正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa	：	イベントフラグID
bbbbbbbb	：	イベントフラグのビットパターン
cccc	：	待ちタスクID（待ちタスクが無い場合は"0000"を表示します）

本コマンドはflg_stsシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始イベントフラグID>終了イベントフラグID（範囲指定時）
2	*** I34:UNDEFINED OBJECT	イベントフラグを「未使用」でHI-SH7を構築している
3	E_NOEXS	イベントフラグIDエラー ・ID範囲外（イベントフラグID<0, イベントフラグID>最大イベントフラグID*1） ・予約ID（イベントフラグID=0）

【注】*1 最大イベントフラグIDは、HI-SH7カーネルセットアップテーブルのhi_maxflgidに定義した値です。

【注意事項】

(1) 本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

CLR_FLG, POL_FLG, SET_FLG

【使　用　例】

(1) イベントフラグID=1の状態を表示します。

```
#FLG_STS 1(RET)
ID=0001 FLGPTN=00000001 WTSK=0001
#
```

(2) イベントフラグID=1, 2, 3の状態を表示します。

```
#FLG_STS 1,2,3(RET)
ID=0001 FLGPTN=00000001 WTSK=0001, ID=0002 FLGPTN=000000FF WTSK=0002,
ID=0003 FLGPTN=FFFFFFFF WTSK=0000
#
```

(3) イベントフラグID=1から4の状態を表示します。

```
#FLG_STS 1:4(RET)
ID=0001 FLGPTN=00000001 WTSK=0001, ID=0002 FLGPTN=000000FF WTSK=0002,
ID=0003 FLGPTN=FFFFFFFF WTSK=0000, ID=0004 FLGPTN=0000FF00 WTSK=0000
#
```

(4) すべてのイベントフラグの状態を表示します。

```
#FLG_STS(RET)
ID=0001 FLGPTN=00000001 WTSK=0001, ID=0002 FLGPTN=000000FF WTSK=0002,
ID=0003 FLGPTN=FFFFFFFF WTSK=0000, ID=0004 FLGPTN=0000FF00 WTSK=0000
ID=0005 FLGPTN=FFFF0000 WTSK=0000
#
```

３．３．１４	GET_BLK	GET memory Block
	GETB	固定長メモリブロックを獲得する

オンライン

【コマンドフォーマット】

GET_BLK△<メモリプールID>[,<メモリプールID>...](RET)

<メモリプールID>　：　対象メモリプールID（H'0001～H'03FF）

【説　　明】

指定したメモリプールから1つのメモリブロックを獲得し、獲得したメモリブロックの先頭アドレスを表示します。メモリプールIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

ID=aaaa BLK=bbbbbbbb

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　：　メモリプールID
bbbbbbbb　：　メモリブロックの先頭アドレス

本コマンドはpget_blkシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	*** I34:UNDEFINED OBJECT	メモリプールを「未使用」でHI-SH7を構築している
3	E_NOEXS	メモリプールIDエラー ・ID範囲外（メモリプールID<0, メモリプールID>最大メモリプールID[*1]） ・予約ID（メモリプールID=0）
4	E_PLFAIL	ポーリング失敗（メモリプールにメモリブロックがない）

【注】*1　最大メモリプールIDは、HI-SH7カーネルセットアップテーブルのhi_maxmplidに定義した値です。

【関連コマンド】

REL_BLK, MPLDEF, MPL_STS

【注意事項】

(1)　本コマンドで獲得したメモリブロックは、本デバッガでは管理しません。したがって、獲得したメモリブロックは、ユーザが管理してください。（REL_BLKコマンドでメモリブロックを返却する場合には、メモリブロックの先頭アドレスを指定する必要があります。本デバッガには、獲得したメモリブロックを自動的に返却する機能はありません。）

【使　用　例】

(1)　メモリプールID=1のメモリプールから1つのメモリブロックを獲得します。

```
#GET_BLK 1(RET)
ID=0001 BLK=0FFFE100
#
```

(2)　メモリプールID=1,2,3のメモリプールから、それぞれ1つのメモリブロックを獲得します。

```
#GET_BLK 1,2,3(RET)
ID=0001 BLK=0FFFE110, ID=0002 BLK=0FFFE200
ID=0003 BLK=0FFFE400
#
```

３．３．１５	GET_TIM	GET Time
	GETT	HI-SH7システムクロックを表示する

オフライン／オンライン

【コマンドフォーマット】

GET_TIM(RET)

【説　　明】

HI-SH7システムが保持しているシステムクロックの現在値を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

TIME=aaaa bbbbbbbb

(2) 正常に処理されなかった場合

<応答メッセージ>

aaaa　：　現在の年月日時刻データの上位
bbbbbbbb　：　現在の年月日時刻データの下位

本コマンドはget_timシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	E_TNOSPT	タイマがサポートされていない

【注意事項】

(1) 本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

SET_TIM

【使　用　例】

(1) システムクロックの現在値を表示します。

```
#GET_TIM(RET)
TIME=0000 00037725
#
```

３．３．１６	GO	Go program
	G	プログラムの実行

オフライン

【コマンドフォーマット】

GO[△<開始アドレス>(RET)

<開始アドレス>　　：　プログラムの開始アドレス

【説　　明】

指定した開始アドレスからプログラムを実行します。開始アドレスを省略すると、現在のPCの値から実行します。

本コマンドを実行すると、デバッグモードは、オフラインモードからオンラインモードになります。

実行が停止した場合、実行タスクID（または割込みレベル）、全レジスタ、アドレスと命令ニーモニック、停止理由を表示します。

(1)ブレークポイントで停止した場合

(a)　<実行タスクID（または割込みレベル）>

(b)　PC=00003102 SR=00000000:----------
PR=00003080 GBR=05FFFEC0 VBR=00000000
MACH=00000000 MACL=00000000
R0-7　00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394

(c)　<アドレス> : <命令ニモニック>

(d)　<停止理由>

(a)　：　タスク部で実行を停止した場合、タスクID(TSKID=0001)を表示します。非タスク部で実行を停止した場合、割込みレベル(INTRRUPT LEVEL=5)を表示します。

(b)　：　停止時の各レジスタの内容です。

(c)　：　命令のアドレスとニモニックを表示します。

(d)　：　表３－５に示す停止理由を表示します。

(2)ブレークポイント以外で停止した場合

(a)　<停止理由>

(b)　PC=00003102 SR=00000000:----------
PR=00003080 GBR=05FFFEC0 VBR=00000000
MACH=00000000 MACL=00000000
R0-7　00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394

(a)　：　表３－５に示す停止理由を表示します。

(b)　：　停止時の各レジスタの内容です。

表３－５　停止理由

項番	表　示	停　止　理　由
1	BREAK POINT=xxxxxxxx	BREAKコマンドによるブレークポイントxxxxxxxxで停止
2	BREAK_UBC POINT=xxxxxxxx	BREAK_UBCコマンドによるブレークポイントxxxxxxxxで停止
3	CPU BUS ERROR !!!	CPUバスエラーが発生した
4	DMA BUS ERROR !!!	DMAバスエラーが発生した
5	ILLEGAL INSTRUCTION !!!	不当命令を実行
6	BREAK(NMI) !!!	NMI割込みが発生した
7	SLOT ILLEGAL INSTRUCTION !!!	スロット不当命令を実行
8	BREAK(CTRL+C) !!!	(CTRL)+Cキーにより強制終了

【応答メッセージ】

項番	**応答メッセージ**	内　　容
1	*** I04:INVALID ADDRESS	アドレスエラー（アドレスが0または奇数である）
2	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
3	*** I31:RESERVED ADDRESS AREA	アドレスエラー（アドレスがHI-SH7または本デバッガの領域である）

【注意事項】

(1)　HI-SH7カーネルを一度も起動していない状態のオフラインモードからオンラインモードへ移行（HI-SH7を起動）する場合、必ずオフラインデバッガが起動した状態のPCの値から実行してください。

【関連コマンド】

.<レジスタ名>, REGISTER, BREAK, BREAK_UBC

【使　用　例】

(1)　現在のPCの値からプログラムを実行します。

```
:GO(RET)
#
```

(2)　H'1000番地からプログラムを実行します。

```
:GO 1000(RET)
#
```

３．３．１７	IHELP	Itron debugger HElp
	IHE	デバッガのコマンドヘルプメッセージの表示

オフライン／オンライン

【コマンドフォーマット】

IHELP[△<コマンド名>](RET)

<コマンド名>　　：　本デバッガのコマンド名（省略形での指定可）

【説　　明】

指定したデバッガのコマンドフォーマットを表示します。デバッガのコマンド名を省略した場合、デバッガがサポートしているコマンドを表示します。

(1)　全コマンドの実行可能モード、省略形、および正式名称を表示します。

```
:IHELP(RET)
  .<REGISTER>          -A    : ASSEMBLE          B     : BREAK
  BU    : BREAK_UBC    +CAN  : CAN_WUP           +CHG  : CHG_PRI
 +CLR   : CLR_FLG      +CRE  : CRE_TSK           +DEL  : DEL_TSK
  DA    : DISASSEMBLE   D    : DUMP              F     : FILL
  FLG   : FLG_STS      +GETB : GET_BLK           GETT  : GET_TIM
 -G     : GO           IHE   : IHELP             IID   : IID
  IST   : ISTATUS      ITR   : ITRACE            ITA   : ITRACE_ACTIVATE
  ITB   : ITRACE_BUFFER ITS  : ITRACE_SEARCH     -L    : LOAD
  MBX   : MBX_STS      M     : MEMORY            MV    : MOVE
  MPLD  : MPLDEF       MPL   : MPL_STS           +POL  : POL_FLG
 +RCV   : RCV_MSG      R     : REGISTER          +RELB : REL_BLK
 +RELW  : REL_WAI      +REQ  : REQ_SEM           -RS   : RESET
 +ROT   : ROT_RDQ      +RSM  : RSM_TSK           +RUN  : RUN_TSK
 -SV    : SAVE         SEM   : SEM_STS           +SETF : SET_FLG
 +SETT  : SET_TIM      +SIG  : SIG_SEM           +SND  : SND_MSG
 +STA   : STA_TSK      -S    : STEP              -SO   : STEP_OVER
 -SU    : STEP_UBC     +STP  : STP_TSK           +SUS  : SUS_TSK
  SVCD  : SVCDEF       SYSD  : SYSDEF            +TER  : TER_TSK
  TSKD  : TSKDEF       TSK   : TSK_STS           -V    : VERIFY
 +WUP   : WUP_TSK
:
```

全コマンドの表示フォーマットを次に示します。

```
:IHELP(RET)
 abbbb : ccc......c     abbbb : ccc......c     abbbb : ccc......c
```

a　　：　コマンドの実行可能モードを表わします
　　なし　：　オフライン、オンライン両モード
　　+　：　オンラインモードのみ
　　-　：　オフラインモードのみ

bbbb　　：　コマンドの省略形

ccc......c　　：　コマンドの正式名称

(2) 各コマンドのフォーマットを表示します。

```
:IHELP STA_TSK(RET)
 (Start task)
  STA_TSK <tskid>[,<tskid>...](RET)
:
```

各コマンドの表示フォーマットを次に表示します。

```
:IHELP <コマンド名>(RET)
 (ddd......d)
  eee......e
```

ddd……d　：　コマンドの内容

eee……e　：　コマンドのフォーマット

- <　>　：　<　>内に示した内容を指定または表示することを示します。
- [　]　：　[　]で囲まれた部分のパラメータは省略可能を示します。
- {｜}　：　{｜}で囲まれた部分の複数のパラメータのうち、１つを選択して指定することを示します。
- <ﾊﾟﾗﾒｰﾀ1>,<ﾊﾟﾗﾒｰﾀ2>,...　：　繰り返し指定可能であることを示します。（パラメータは最大８つまで指定可能）
- <ﾊﾟﾗﾒｰﾀ1>:<ﾊﾟﾗﾒｰﾀ2>　：　範囲指定が可能であることを示します。（<ﾊﾟﾗﾒｰﾀ1>から<ﾊﾟﾗﾒｰﾀ2>まで）
- ...　：　直前の指定を繰り返し入力できることを示します。
- (RET)　：　リターンキーの入力を示します。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I01:SYNTAX ERROR	シンタックスエラー（指定したコマンド名は存在しない）

３．３．１８	IID	Itron debugger ID
	IID	デバッガのバージョンの表示

オフライン／オンライン

【コマンドフォーマット】

IID(RET)

【説　　明】

本デバッガのバージョンおよびリビジョンなどを表示します。

```
:IID(RET)
 SH7000 HI-SH7 DEBUGGER (HS0700IRCN1SMB) Ver n.m
 Copyright (C) Hitachi, Ltd. 1994
 Licensed Material of Hitachi, Ltd.
:
```

３．３．１９	ISTATUS	Itron STatus
	IST	指定状態のタスクを表示する

オフライン／オンライン

【コマンドフォーマット】

ISTATUS△<タスク状態>(RET)

<タスク状態>　：　タスク状態の名称

RUN　：　実行状態
RDY　：　実行可能状態
WAI　：　wai_tskシステムコールによる待ち状態
SLP　：　slp_tskシステムコールによる待ち状態
　　wai_tskシステムコールでtmout=-1指定による待ち状態
SUS　：　強制待ち状態
DMT　：　休止状態
STK　：　共有スタックによる起動待ち状態
FLG　：　wai_flgシステムコールによる待ち状態
SEM　：　wai_semシステムコールによる待ち状態
MBX　：　rcv_msgシステムコールによる待ち状態
MPL　：　get_blkシステムコールによる待ち状態
DBG　：　デバッグ待ち状態

【説　　明】

指定した状態のタスクIDを表示します。指定した状態のタスクが複数存在する場合は、タスクIDの小さい順に表示します。

タスク状態の名称には、複合した状態（二重待ち状態など）を指定することはできません。

なお、複合した状態のタスクは、要素となる状態を指定すると表示対象となり、重複して表示されます。例えば、メッセージ待ちでかつ強制待ち状態のタスクは、MBXとSUSのどちらを指定しても表示されます。

【注意事項】

(1) 本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。
- HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。
- カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

TSK_STS

【使　用　例】

(1)　実行可能（RDY）状態のタスクを表示します。

```
#ISTATUS RDY(RET)
0001, 0003, 0004
#
```

(2)　休止（DORMANT）状態のタスクを表示します。

```
#ISTATUS DMT(RET)
0005, 0006, 0007, 0008, 0009, 000A, 000B, 000C,
0010
#
```

３．３．２０	ITRACE	Itron TRace
	ITR	HI-SH7トレース情報の表示

オフライン／オンライン

【コマンドフォーマット】

ITRACE[△<開始イベント番号>[:<終了イベント番号>]] [△N](RET)

<開始イベント番号> ： トレース表示の開始イベント番号
<終了イベント番号> ： トレース表示の終了イベント番号
N ： デバッガ情報の未表示の指定

【説　明】

開始イベント番号で指定したイベントから、終了イベント番号で指定したイベントまでを表示します。

開始イベント番号だけ指定した場合には、指定したイベントから最終イベントまでを表示します。また、開始／終了イベント番号を省略した場合にはすべてのイベントを表示します。イベント番号は、本コマンド入力以前のイベントであるため、"－"を付けて指定します。

Nを指定すると、デバッガによるイベントを表示しません。

オンラインモードで本コマンドを使用した場合には、無条件にイベント取得を停止します。イベント取得を開始する場合には、ITRACE_ACTIVATEコマンドにより行ないます。なお、オフラインモードで本コマンドを使用した場合には、イベント取得停止は行ないません。

イベントは、以下のフォーマットで表示します。

(1) イベント属性がSVC属性の場合

```
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=SVC    EVENT=dddddddd
-D'aaaaaaaa bbbbbbbb cccc eeeeeeee ffffffff gggggggg hhhhhhhh iiiiiiii jjjjjjjj
```

aaaaaaaa ： イベント番号です。トレースバッファに格納されている情報の中で、最新のイベントに対する番号は0です。
SVC ： イベント属性です。
bbbbbbbb ： システムコール発行時のシステムクロックの下位4バイトの値です。
cccc ： システムコールを発行したタスクIDです。非タスク部の場合は"NTSK"を表示します。
dddddddd ： 発行したシステムコール名です。HI-SH7がサポートしているシステムコールを発行した場合にはその名称を発行します。またデバッグハンドラが発行するシステムコールのうち、ユーザに公開されていないシステムコールについてはそのシステムコールに対するコマンド名を表示します。それ以外（拡張SVC含む）のシステムコールを発行した場合には機能コード（R0の値）を表示します。
eeeeeeee ： システムコール発行アドレス+2を示します。
ffffffff ： システムコール発行時のSRの値です。
gggggggg ： システムコール発行時のR4レジスタ（システムコールの第1パラメータ）の値です。
hhhhhhhh ： システムコール発行時のR5レジスタ（システムコールの第2パラメータ）の値です。
iiiiiiii ： システムコール発行時のR6レジスタ（システムコールの第3パラメータ）の値です。
jjjjjjjj ： システムコール発行時のR7レジスタ（システムコールの第4パラメータ）の値です。

(2) イベント属性がRTN属性の場合（システムコール発行元に戻る場合の表示フォーマット）

```
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=RTN    EVENT=dddddddd
-D'aaaaaaaa bbbbbbbb cccc eeeeeeee ffffffff gggggggg hhhhhhhh iiiiiiii jjjjjjjj
```

aaaaaaaa	：	イベント番号です。トレースバッファに格納されている情報の中で、最新のイベントに対する番号は0です。
RTN	：	イベント属性です。
bbbbbbbb	：	システムコール発行元に戻る時のシステムクロックの下位4バイトの値です。
cccc	：	システムコールを発行したタスクIDです。非タスク部の場合は"NTSK"を表示します。
dddddddd	：	システムコールのエラーコードです。HI-SH7がサポートしているシステムコールからのエラーコードである場合にはその名称を表示します。それ以外の値である場合にはエラーコード（R0の値）を表示します。
eeeeeeee	：	ユーザプログラムへの戻り先（システムコール発行アドレス+2）アドレスを示します。
ffffffff	：	ユーザプログラムへ戻った時のSRの値です。
gggggggg	：	システムコールからリターン時のR4レジスタの値です。
hhhhhhhh	：	システムコールからリターン時のR5レジスタの値です。
iiiiiiii	：	システムコールからリターン時のR6レジスタの値です。
jjjjjjjj	：	システムコールからリターン時のR7レジスタの値です。

(3) イベント属性がRTN属性の場合（タスク起動時の表示フォーマット）

```
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=RTN    EVENT=INITIATE
-D'aaaaaaaa bbbbbbbb cccc eeeeeeee ffffffff
```

aaaaaaaa	：	イベント番号です。トレースバッファに格納されている情報の中で、最新のイベントに対する番号は0です。
RTN	：	イベント属性です。
bbbbbbbb	：	タスクが起動される時のシステムクロックの下位4バイトの値です。
cccc	：	起動されるタスクIDです。
INITIATE	：	タスクが起動される（タスク起動のイベント）ことを示します。
eeeeeeee	：	起動されるタスクのPCの値（タスク開始アドレス）です。
ffffffff	：	起動されるタスクのSRの値です。

(4) イベント属性がCONT属性の場合

```
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=CONT   EVENT=********
-D'aaaaaaaa bbbbbbbb  cccc
```

aaaaaaaa　：　イベント番号です。トレースバッファに格納されている情報の中で、最新のイベントに対する番号は0です。
CONT　：　イベント属性です。
bbbbbbbb　：　タスクが実行再開するときのシステムクロックの下位4バイトの値です。
cccc　：　割込み発生時点から実行再開するタスクIDです。

(5) イベント属性がIDLE属性の場合

```
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=IDLE   EVENT=********
-D'aaaaaaaa bbbbbbbb
```

aaaaaaaa　：　イベント番号です。トレースバッファに格納されている情報の中で、最新のイベントに対する番号は0です。
IDLE　：　イベント属性です。
bbbbbbbb　：　システムアイドリングに入る時のシステムクロックの下位4バイトの値です。

(6) デバッガによるイベントの場合

```
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=kkkk   EVENT=DEBUGGER
-D'aaaaaaaa bbbbbbbb
```

aaaaaaaa　：　イベント番号です。トレースバッファに格納されている情報の中で、最新のイベントに対する番号は0です。
bbbbbbbb　：　デバッガによるイベント発生時のシステムクロックの下位4バイトの値です。
kkkk　：　イベント属性（SVC/RTN/CONT）です。
DEBUGGER　：　デバッガによるイベントを表わします。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I30:UNDEFINED TRACE BUFFER	トレースバッファ領域が定義されていません

【注意事項】

(1) 本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

(2) トレース機能の詳細については、『**HI-SH7ユーザーズマニュアル**』の**「２．１６　トレース機能」**を参照してください。

【関連コマンド】

ITRACE_ACTIVATE, ITRACE_BUFFER, ITRACE_SEARCH

【使　用　例】

(1) 最新のイベントから4イベント分を表示します。

```
#ITRACE -3(RET)
 NO           LTIME    TSKID    PC        SR        R4        R5        R6        R7
      ATR=SVC    EVENT=ter_tsk
-D'00000003 00000A32 NTSK 00003DAC 000000D0 00000002 00000000 00000000 00000000
      ATR=RTN    EVENT=E=OK
-D'00000002 00000A32 NTSK 00003DAC 000000D0 00000002 00000000 00000000 00000000
      ATR=CONT   EVENT=********
-D'00000001 00000A32 0003
      ATR=SVC    EVENT=DEBUGGER
-D'00000000 00000AC8
#
```

(2) イベント番号-H'E（-D'14）から-H'9（-D'9）までのイベントを表示します。

```
#ITRACE -E:-9(RET)
 NO           LTIME    TSKID    PC        SR        R4        R5        R6        R7
      ATR=SVC    EVENT=set_flg
-D'00000014 00000510 0002 00004100 00000000 00000005 0000000A 00000000 00000000
      ATR=RTN    EVENT=E=OK
-D'00000013 00000510 0002 00004100 00000000 00000005 0000000A 00000000 00000000
      ATR=SVC    EVENT=sig_sem
-D'00000012 0000051C 0002 0000415C 00000000 0000000A 0000FFFF 00000000 00000000
      ATR=RTN    EVENT=E=OK
-D'00000011 00000612 0002 0000415C 00000000 0000000A 0000FFFF 00000000 00000000
      ATR=SVC    EVENT=ext_tsk
-D'00000010 0000063A 0002 000042D4 00000000 00000001 00000000 00000000 00000000
      ATR=IDLE   EVENT=********
-D'00000009 0000063A
#
```

３．３．２１	ITRACE_ACTIVATE	Itron Trace Activate
	ITA	HI-SH7トレース取得の開始、停止、表示

オフライン／オンライン

【コマンドフォーマット】

ITRACE_ACTIVATE[△<活性情報>](RET)

<活性情報>　：　トレース情報取得の開始／停止情報
ON　：　トレース情報取得を開始
OFF　：　トレース情報取得を停止

【説　明】

HI-SH7トレース情報の取得を開始または停止します。
活性情報を省略した場合には、現在のトレース情報取得の活性状態を表示します。

現在のトレース情報取得の活性状態は、以下のフォーマットで表示します。

ITRACE_ACTIVATE=aaa

aaa　：　トレース情報取得の活性状態を示します。
ON　：　トレース情報取得中であることを示します。
OFF　：　トレース情報取得は停止状態であることを示します。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I30:UNDEFINED TRACE BUFFER	トレースバッファ領域が定義されていません

【注意事項】

(1)　トレース情報取得の活性状態表示だけは、オフラインモードでも実行することができますが、以下の点に注意してください。
・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。
・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

(2)　トレース情報取得の開始または停止は、オンラインモードのみ実行可能です。

(3)　トレース情報取得の開始を指定した場合、それまでに取得されたトレース情報はすべてクリアされます。

(4)　トレース機能の詳細については、『**HI-SH7ユーザーズマニュアル**』の「**２．１６　トレース機能**」を参照してください。

【関連コマンド】

ITRACE, ITRACE_BUFFER, ITRACE_SEARCH

【使　用　例】

(1)　トレース情報取得を開始します。

```
#ITRACE_ACTIVATE ON(RET)
#
```

(2)　トレース情報取得を停止します。

```
#ITRACE_ACTIVATE OFF(RET)
#
```

(3)　現在のトレース情報取得の活性状態を表示します。

```
#ITRACE_ACTIVATE(RET)
ITRACE_ACTIVATE=OFF
#
```

３．３．２２	ITRACE_BUFFER	Itron Trace Buffer
	ITB	HI-SH7トレースバッファの設定、表示

オフライン／オンライン

【コマンドフォーマット】

ITRACE_BUFFER[△<先頭アドレス>{△<最終アドレス+1>|△@<バイト数>}](RET)

<先頭アドレス>　：　トレースバッファの先頭アドレス
<最終アドレス>　：　トレースバッファの最終アドレス+1
<バイト数>　：　トレースバッファのサイズ

【説　明】

(1)　トレースバッファの設定

トレースバッファは、先頭アドレスと、最終アドレスまたはバイト数で領域を指定します。
トレースバッファ領域のサイズは、次の計算式で求めます。

トレースバッファ領域サイズ ＝ H'10 ＋ H'24 × n　　n：トレースエントリ数（>1）

したがって、H'34（52）バイト以上の領域を指定する必要があります。これより小さい領域を指定した場合は、エラー"*** I04:INVALID ADDRESS"を表示します。また、指定した領域が計算式と一致しない場合には、下方（計算式の単位）に丸めて設定します。

なお、トレースバッファを設定すると、無条件にイベントの取得を停止します。トレースバッファ設定後に、イベントの取得を開始するには、ITRACE_ACTIVATEコマンドを実行する必要があります。

(2)　トレースバッファの表示

トレースバッファ領域の指定を省略した場合には、現在設定されているトレースバッファ領域のアドレスを表示します。現在設定されているトレースバッファ領域のアドレスは、以下のフォーマットで表示します。

ITRACE_BUFFER=aaaaaaaa bbbbbbbb

aaaaaaaa　：　現在設定されているトレースバッファの先頭アドレスです。
bbbbbbbb　：　現在設定されているトレースバッファの最終アドレス+1です。

なお、トレースバッファが未定義の時は、"*** I30:UNDEFINED TRACE BUFFER"を表示します。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I04:INVALID ADDRESS	アドレスエラー ・アドレスが0または奇数である ・トレースバッファのサイズが52バイト未満である
2	*** I13:NOT RAM AREA	アドレスエラー（指定したバッファ領域はRAM領域でない）
3	*** I31:RESERVED ADDRESS AREA	アドレスエラー（指定したバッファ領域はHI-SH7または本デバッガの領域である）
4	*** I30:UNDEFINED TRACE BUFFER	トレースバッファ領域が定義されていません

【注意事項】

(1) 本コマンドのトレースバッファアドレスの表示は、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

(2) 本コマンドのトレースバッファアドレスの設定は、オンラインモードのみ実行可能です。

(3) 本コマンドでトレースバッファを設定した場合、それまでに取得されたトレース情報はすべてクリアされます。

(4) トレース機能の詳細については、『**HI-SH7ユーザーズマニュアル**』の**「２．１６　トレース機能」**を参照してください。

【関連コマンド】

ITRACE, ITRACE_ACTIVATE, ITRACE_SEARCH

【使　用　例】

(1) H'0FFFF000番地からH'0FFFF640番地の領域をトレースバッファとして設定します。

```
#ITRACE_BUFFER FFFF000 FFFF640(RET)
#
```

(2) H'0FFFF000からH'640バイトの領域をトレースバッファとして設定します。

```
#ITRACE_BUFFER FFFF000 @640(RET)
#
```

(3) 現在設定されているトレースバッファのアドレスを表示します。

```
#ITRACE_BUFFER(RET)
ITRACE_BUFFER=0FFFF000 0FFFF640
#
```

３．３．２３	ITRACE_SEARCH	Itron Trace Search
	ITS	HI-SH7トレース情報の検索、表示

オフライン／オンライン

【コマンドフォーマット】

ITRACE_SEARCH[△<条件>[△<条件>...]][△<開始イベント番号>[:<最終イベント番号>]][△N](RET)

<条件>	：	検索、表示するトレース情報を指定 省略時、取得してあるトレース情報のイベント数とシステムクロックの範囲を表示します。
<開始イベント番号>	：	検索を開始するイベント番号
<最終イベント番号>	：	検索を終了するイベント番号
N	：	デバッガ情報の未表示の指定

【説　　明】

指定した条件でトレースバッファ内のイベントを検索し、該当するイベントをすべて表示します。オンラインモードで本コマンドを使用した場合、無条件にトレース情報の取得を停止します。

条件の指定を省略すると、トレースバッファ内の全イベント数とシステムクロックの範囲を表示します。

条件を指定せずに開始および最終イベント番号を指定した場合は、エラー"*** I01:SYNTAX ERROR"を表示します。

条件として、表３－６に示す項目を指定できます。各項目は組み合わせて指定できます。各条件での組み合わせは、AND条件となります。ただし、同一条件内での組み合わせ（同一条件に複数のパラメータを指定）はOR条件となります。

Nを指定すると、デバッガによるイベントを表示しません（検索対象から外します）。

表３－６　トレース情報検索の条件

項番	項目と入力フォーマット	内　　容
1	<システムクロック条件> TIM=<開始システムクロック値> [:<終了システムクロック値>]	検索する範囲を開始システムクロック値から終了システムクロック値で指定したイベント内とします。終了システムクロック値を省略した場合には最終イベントまでのイベントを検索する範囲とします。 指定したシステムクロック値はトレースエントリに取得される値（下位4バイト）として検索範囲を決定します。
2	<タスクID条件> TSKID=<tskid>[,<tskid>...] [,NTSK]*1	指定したタスクIDに関するイベントを検索または、非タスク部に関する情報を検索します。本条件を省略した場合にはタスクおよび非タスク部の情報を検索の対象とします。指定可能なタスクIDは"NTSK"も含めて最大8個です。 検索するイベント属性に、"IDLE"を指定した場合には、本条件は無視されます。ただしイベント属性を指定せずに本条件を指定した場合には、イベント属性"IDLE"は検索されません。
3	<イベント属性条件> ATR=SVC[,RTN[,CONT[,IDLE]]]	指定した属性のイベントを検索します。本条件を省略した場合にはすべての属性のイベントを検索の対象とします。*2
4	<イベント情報条件> EVNT=<event値>[,<event値>...]	指定したevent値（イベント情報）に関するイベントを検索します。event値にはシステムコールおよびエラーコードの名称、値（数値定数）を指定できます。本条件を省略した場合には、すべてのevent値を検索の対象とします。指定可能なevent値は最大8個です。 検索するイベント属性に、"CONT", "IDLE"を指定した場合には、本条件は無視されます。ただし、イベント属性を指定せずに本条件を指定した場合には、イベント属性"IDLE", "CONT"は検索されません。

【注】*1　"NTSK"は非タスク部を意味します。したがって非タスク部のイベントを検索する場合には"NTSK"と指定してください。また、"NTSK"ではなく0を指定しても非タスク部のイベントを検索します。

*2　イベント属性条件を指定せずに、タスクID条件およびイベント情報条件を指定した場合には、その条件が有効となるイベントだけを検索します。例えばイベント情報条件だけを指定した場合には、検索の対象となるイベントは"SVC"と"RTN"だけとなります。トレースエントリ内容については、**『HI-SH7ユーザーズマニュアル』**の**「２．１６　トレース機能」**を参照してください。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I10:NOT FOUND	指定した条件のイベントはない
2	*** I30:UNDEFINED TRACE BUFFER	トレースバッファ領域が定義されていません

(1)　トレース情報の検索

　トレースバッファに格納されている内容を、指定した条件で検索します。検索したイベントは、ITRACEコマンドで表示するイベントのフォーマットと同一フ ォーマットで表示されます。

　なお、指定した条件に合致するイベントが無い場合には"*** I10:NOT FOUND"を表示します。

(2)　トレースバッファ内の全イベント数の表示

　検索の条件、および検索開始／終了イベント番号を省略すると、トレースバッファ内に取得されている全イベント数と、そのイベントを取得したシステムクロックの範囲が表示されます。

　全イベント数は以下のフォーマットで表示されます。

```
NUMBER OF EVENTS=aaaaaaaa  LTIME=bbbbbbbb - cccccccc
```

aaaaaaaa	：	トレースバッファに取得されている全イベント数です。
bbbbbbbb	：	トレースバッファに取得されている最も古いイベントに対する、システムクロック値を示します。
cccccccc	：	トレースバッファに取得されている最も新しいイベント（イベント番号0）に対する、システムクロック値を示します。

【注意事項】

(1)　本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

(2)　検索の範囲（イベント番号、システムクロック値）が、トレースバッファ内のイベントの内容を越えている（検索範囲オーバ）場合には、無条件に"*** I10:NOT FOUND"を表示します。

(3)　指定したタスクID条件については、最大登録可能IDである1023(H'3FF)を越えているかだけをチェックします。すなわち、指定したタスクが存在するかどうかのチェックは行ないません。なお、タスクIDとして0を指定した場合には、"NTSK"を指定したものとして検索します。

(4)　タスクや割込みハンドラでset_timシステムコールを発行、またはSET_TIMコマンドを実行した場合には、全イベントのシステムクロックの範囲が逆に表示されることがあります。この場合には、検索条件としてシステムクロック条件を指定しないでください。指定した場合には、表示される検索結果は保証されません。

　set_timシステムコールを発行、またはSET_TIMコマンドでシステムクロックを設定する場合には、検索を行なう前に全イベント数の表示を行なってシステムクロック値の範囲が逆になっていないことを確認してください。

(5)　トレース機能の詳細については、**『HI-SH7ユーザーズマニュアル』**の**「２．１６　トレース機能」**を参照してください。

【関連コマンド】

ITRACE, ITRACE_ACTIVATE, ITRACE_BUFFER

【使　用　例】

(1)　トレースバッファ内の全イベント数を表示します。

```
#ITRACE_SEARCH(RET)
NUMBER OF EVENTS=000001CB  LTIME=00000123 - 00000138
```

(2)　tskid=3, 5で、sta_tskシステムコール発行およびエラーE_NOEXSが返されるトレース情報を検索します。

```
#ITRACE_SEARCH TSKID=3,5 ATR=SVC,RTN EVNT=STA_TSK,E_NOEXS(RET)
 NO          LTIME    TSKID    PC        SR        R4        R5        R6        R7
     ATR=SVC    EVENT=sta_tsk
-D'00000032 00000010 0003 000031EC 00000000 000000FF 00000000 00000000 00000000
     ATR=RTN    EVENT=E_NOEXS
-D'00000031 00000010 0003 000031EC 00000000 000000FF 00000000 00000000 00000000
     ATR=SVC    EVENT=sta_tsk
-D'00000005 00000023 0005 0000351A 00000000 0000000A 00000000 00000000 00000000
#
```

３．３．２４	LOAD	Load
	L	ファイルのロード

オフライン

【コマンドフォーマット】

LOAD△<ファイル名>[;S](RET)

<ファイル名>　：　ロードするファイル名
S　：　Sタイプロードモジュールの指定
（省略時：SYSROFタイプロードモジュール）

【説　明】

ホストコンピュータから、指定したロードモジュールをロードします。

Sパラメータを指定すると、Sタイプロードモジュール（標準ファイル拡張子：.mot）をロードします。省略すると、SYSROFタイプロードモジュール（標準ファイル拡張子：.abs）をロードします。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常にロードした場合

```
TOP     ADDRESS = aaaaaaaa
BOTTOM ADDRESS = bbbbbbbb
LOAD NORMAL END
```

aaaaaaaa　：　ロードしたメモリ先頭アドレス
bbbbbbbb　：　ロードしたメモリ最終アドレス

(2) ロード中に異常が発生した場合

<応答メッセージ>

本コマンドを使用するには、ホストコンピュータとHシリーズインタフェースソフトが必要となります。ホストコンピュータとは、Hシリーズインタフェースソフトを実行して、本デバッガの端末として動作させているコンピュータのことです。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
2	*** I20:TIME OUT ERROR	タイムアウトエラー（シリアル回線が正しく接続されていません）
3	*** I21:CHECK SUM ERROR	ロード中にチェックサムエラーが発生しました
4	*** I22:NOT SAME FILE	指定したロードモジュールの形式が違います
5	*** I31:RESERVED ADDRESS AREA	ロードする領域がHI-SH7または本デバッガの領域である

【注意事項】

(1) 本デバッガでは、SYSROFタイプロードモジュールに含まれるシンボル情報は無視します。

(2) 本コマンドは、メモリとのベリファイは行ないません。ベリファイが必要な場合は、VERIFYコマンドを使用してください。

(3) ホストコンピュータとの転送速度は9600BPSにしてください。

【関連コマンド】

SAVE, VERIFY

【使　用　例】

(1) SYSROFタイプのロードモジュール"TEST.ABS"をロードします。

```
:L TEST.ABS(RET)
 TOP     ADDRESS = 00010000
 BOTTOM  ADDRESS = 0001063F
 LOAD  NORMAL  END
:
```

(2) Sタイプロードモジュール"TEST.MOT"をロードします。

```
:L TEST.MOT;S(RET)
 TOP     ADDRESS = 00010000
 BOTTOM  ADDRESS = 0001063F
 LOAD  NORMAL  END
:
```

３．３．２５	MBX_STS	MailBoX status
	MBX	メイルボックスの状態を表示する

オフライン／オンライン

【コマンドフォーマット】

MBX_STS[△<メイルボックスID>[{,<メイルボックスID>... ¦ :<メイルボックスID>}]](RET)

<メイルボックスID> ： 対象メイルボックスID（H'0001～H'03FF）

【説　　明】

指定したメイルボックスの状態を参照して、次に受信されるメッセージの先頭アドレスまたは待ちタスクIDを表示します。

メイルボックスIDは8個まで繰り返し指定することができます。メイルボックスIDを省略した場合、定義されているすべてのメイルボックスの状態を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

ID=aaaa MSG=bbbbbbbb WTSK=cccc

(2) 正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa	：	メイルボックスID
bbbbbbbb	：	次に受信されるメッセージの先頭アドレス （メッセージが無い場合は"FFFFFFFF"を表示します）
cccc	：	待ちタスクID （待ちタスクが無い場合は"0000"を表示します）

本コマンドはmbx_stsシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始メイルボックスID>終了メイルボックスID（範囲指定時）
2	E_NOEXS	メイルボックスIDエラー ・ID範囲外（メイルボックスID<0, メイルボックスID>最大メイルボックスID[*1]） ・予約ID（メイルボックスID=0）
3	ED_DBGID	メイルボックスIDエラー（指定したメイルボックスは本デバッガで使用している）

【注】*1 最大メイルボックスIDは、HI-SH7カーネルセットアップテーブルのhi_maxmbxidに定義した値です。

【注意事項】

(1) 本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

RCV_MSG, SND_MSG

【使　用　例】

(1) メイルボックスID=1の状態を表示します。

```
#MBX_STS 1(RET)
ID=0001 MSG=00001000 WTSK=0000
#
```

(2) メイルボックスID=1, 2, 3の状態を表示します。

```
#MBX_STS 1,2,3(RET)
ID=0001 MSG=00001000 WTSK=0000, ID=0002 MSG=00002000 WTSK=0000,
ID=0003 MSG=FFFFFFFF WTSK=0001
#
```

(3) メイルボックスID=1から4の状態を表示します。

```
#MBX_STS 1:4(RET)
ID=0001 MSG=00001000 WTSK=0000, ID=0002 MSG=00002000 WTSK=0000,
ID=0003 MSG=FFFFFFFF WTSK=0001, ID=0004 MSG=00004000 WTSK=0000
#
```

(4) すべてのメイルボックスの状態を表示します。

```
#MBX_STS(RET)
ID=0001 MSG=00001000 WTSK=0000, ID=0002 MSG=00002000 WTSK=0000,
ID=0003 MSG=FFFFFFFF WTSK=0001, ID=0004 MSG=00004000 WTSK=0000,
ID=0005 MSG=FFFFFFFF WTSK=0003
#
```

３．３．２６	MEMORY	Memory
	M	メモリ内容の表示、変更

オフライン／オンライン

【コマンドフォーマット】

MEMORY△<アドレス>[△<データ>] [;[<アクセス単位>] [△N]](RET)

<アドレス> ： 表示、変更するメモリのアドレス
<データ> ： 変更データ
<アクセス単位> ： メモリのアクセス単位
B ： 1バイト単位
W ： 2バイト単位
L ： 4バイト単位
省略時 ： 1バイト単位
N ： ベリファイなしの指定

【説　明】

(1) 直接

指定したデータを、指定したアドレスのメモリに書き込みます。

:MEMORY△<アドレス>△<データ>[;[<アクセス単位>][△N]](RET)

(2) 対話

指定したアドレスのメモリ内容を表示し、サブコマンドモードになります。サブコマンドモードで変更データを入力すると、メモリに書き込みます。この動作を終了のサブコマンド入力まで繰り返します。

:MEMORY△<アドレス>[;[<アクセス単位>][△N]](RET)
aaaaaaaa　bbb...b ? <サブコマンド>(RET)

aaaaaaaa ： メモリのアドレス
bbb...b ： メモリの内容
<サブコマンド> ： サブコマンド（表３－７に示す内容を入力します）

表３－７　MEMORYコマンドのサブコマンド

項番	サブコマンド	意　味
1	<データ>	指定したデータを表示アドレスのメモリに書き込みます。
2	(RET)のみ	次のアドレスを表示して、再度サブコマンド入力待ちになります。
3	^	直前のアドレスを表示して、再度サブコマンド入力待ちになります。
4	.	MEMORYコマンドを終了します。

なお、Nパラメータを指定しなかった場合は、データ書き込み後、正しくメモリに書き込まれたかをチェックするため、ベリファイを行ないます。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I04:INVALID ADDRESS	アドレスエラー ・アクセス単位が2バイトのとき、アドレスが奇数である ・アクセス単位が4バイトのとき、アドレスが4の倍数でない
2	*** I23:VERIFY ERROR	ベリファイエラー
3	*** I31:RESERVED ADDRESS AREA	アドレスエラー（アドレスがHI-SH7または本デバッガの領域である）*1

【注】*1　メモリを更新しなければ、エラーとなりません。

【注意事項】

(1)　I/O領域を更新する場合は、Nパラメータを指定（ベリファイなし）してください。

【関連コマンド】

DUMP

【使　用　例】

(1)　直接

・H'1000番地にH'12を書き込みます。

```
:M 1000 12(RET)
:
```

・H'1000番地にH'1234を書き込みます。

```
:M 1000 1234;W(RET)
:
```

・H'1000番地にH'12345678を書き込みます。

```
:M 1000 12345678;L(RET)
:
```

・H'05FFFEC0番地（I/O領域）にH'00AAを書き込みます（ベリファイなし）。

```
:M 5FFFEC0 AA;W N(RET)
:
```

(2)　対話

H'1000番地からメモリ内容の参照と変更を行ないます。

```
:M 1000(RET)
 00001000   00 ? 12(RET)
 00001001   01 ? 34(RET)
 00001002   02 ? 56(RET)
 00001003   03 ? 78(RET)
 00001004   41 ? .(RET)
:
```

３．３．２７	MOVE	MoVe
	MV	メモリ転送

オフライン／オンライン

【コマンドフォーマット】

MOVE△<先頭アドレス>{△<最終アドレス>｜△@<バイト数>}△<転送先アドレス>[;N](RET)

<先頭アドレス>	：	転送元の先頭アドレス
<最終アドレス>	：	転送元の最終アドレス
<バイト数>	：	転送元のバイト数
<転送先アドレス>	：	転送先の先頭アドレス
N	：	ベリファイなしの指定

【説　　明】

指定した転送元のメモリ領域の内容を、指定した転送先のメモリ領域へ転送します。

メモリの転送は、先頭アドレスから順に行ないます。ただし、転送元の領域内に、転送先の先頭アドレスがある場合は、最終アドレスから逆の順序で行ないます。

Nパラメータを指定しなかった場合は、データ転送後、正しくメモリに書き込まれたかをチェックするため、ベリファイを行ないます。エラー発生時は次のメッセージを表示します。

```
FAILED AT  aaaaaaaa  WRITE=bb'b'  READ=cc'c'
```

aaaaaaaa	：	エラーが発生したメモリアドレス
bb'b'	：	書き込みデータ（16進数とASCII表示）
cc'c'	：	読み込みデータ（16進数とASCII表示）

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I31:RESERVED ADDRESS AREA	アドレスエラー（転送先のメモリ領域がHI-SH7または本デバッガの領域である）

【関連コマンド】

DUMP

【使　用　例】

(1)　H'1000番地からH'1FFF番地のメモリ内容を、H'2000番地（H'2000～H'2FFF）へ転送します。

```
:MV 1000 1FFF 2000(RET)
:
```

(2)　H'1000番地からH'FFFバイト分（H'1000～H'1FFF）のメモリ内容を、H'2000番地（H'2000～H'2FFF）へ転送します（ベリファイなし）。

```
:MV 1000 @FFF 2000;N(RET)
:
```

(3)　H'2000番地からH'2FFF番地のメモリ内容を、H'1000番地（H'1000～H'1FFF）へ転送します。

```
:MV 2000 2FFF 1000(RET)
:
```

(4)　H'2000番地からH'2FFF番地のメモリ内容を、H'2800番地（H'2800～H'37FF）へ転送します。この場合のメモリ転送は、最終アドレスから逆の順序で行ないます。

```
:MV 2000 2FFF 2800(RET)
:
```

３．３．２８	MPLDEF	Memory PooL Define information
	MPLD	メモリプール定義情報を表示する

オフライン／オンライン

【コマンドフォーマット】

MPLDEF[△<メモリプールID>[{,<メモリプールID>... ¦ :<メモリプールID>}]](RET)

<メモリプールID>　：　対象メモリプールID（H'0001～H'03FF）

【説　　明】

指定したメモリプールの定義情報を表示します。メモリプール定義情報は、HI-SH7セットアップテーブルで定義した内容です。

メモリプールIDは8個まで繰り返し指定することができます。メモリプールIDを省略した場合、定義されているすべてのメモリプールの定義情報を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

ID=aaaa MPLADR=bbbbbbbb BLKSZ=cccccccc BLKCNT=dddddddd

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa	：	メモリプールID
bbbbbbbb	：	メモリプールの先頭アドレス
cccccccc	：	メモリブロック長
dddddddd	：	メモリブロック数

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始メモリプールID>終了メモリプールID（範囲指定時）
2	*** I34:UNDEFINED OBJECT	メモリプールを「未使用」でHI-SH7を構築している
3	E_NOEXS	メモリプールIDエラー ・ID範囲外（メモリプールID<0, メモリプールID>最大メモリプールID*1） ・予約ID（メモリプールID=0）

【注】*1　最大メモリプールIDは、HI-SH7カーネルセットアップテーブルのhi_maxmplidに定義した値です。

【関連コマンド】

MPL_STS, GET_BLK, REL_BLK, SYSDEF, SVCDEF, TSKDEF

【使　用　例】

(1)　メモリプールID=1の定義情報を表示します。

```
#MPLDEF 1(RET)
ID=0001 MPLADR=0FFFE000 BLKSZ=00000020 BLKCNT=00000004
#
```

(2)　メモリプールID=1, 2, 3の定義情報を表示します。

```
#MPLDEF 1,2,3(RET)
ID=0001 MPLADR=0FFFE000 BLKSZ=00000020 BLKCNT=00000004
ID=0002 MPLADR=0FFFE090 BLKSZ=00000010 BLKCNT=00000008
ID=0003 MPLADR=0FFFE130 BLKSZ=00000064 BLKCNT=0000000A
#
```

(3)　メモリプールID=1から4の定義情報を表示します。

```
#MPLDEF 1:4(RET)
ID=0001 MPLADR=0FFFE000 BLKSZ=00000020 BLKCNT=00000004
ID=0002 MPLADR=0FFFE090 BLKSZ=00000010 BLKCNT=00000008
ID=0003 MPLADR=0FFFE130 BLKSZ=00000064 BLKCNT=0000000A
ID=0004 MPLADR=0FFFE540 BLKSZ=00000030 BLKCNT=00000010
#
```

(4)　すべてのメモリプールの定義情報を表示します。

```
#MPLDEF(RET)
ID=0001 MPLADR=0FFFE000 BLKSZ=00000020 BLKCNT=00000004
ID=0002 MPLADR=0FFFE090 BLKSZ=00000010 BLKCNT=00000008
ID=0003 MPLADR=0FFFE130 BLKSZ=00000064 BLKCNT=0000000A
ID=0004 MPLADR=0FFFE540 BLKSZ=00000030 BLKCNT=00000010
ID=0005 MPLADR=0FFFE880 BLKSZ=00000040 BLKCNT=00000001
#
```

３．３．２９	MPL_STS	Memory PooL status
	MPL	メモリプールの状態を表示する

オフライン／オンライン

【コマンドフォーマット】

MPL_STS[△<メモリプールID>[{,<メモリプールID>... ¦ :<メモリプールID>}]](RET)

<メモリプールID>　：　対象メモリプールID（H'0001～H'03FF）

【説　　明】

指定したメモリプールの状態を参照して、空き領域のメモリブロック数と、待ち行列の先頭のタスクIDを表示します。

メモリプールIDは8個まで繰り返し指定することができます。メモリプールIDを省略した場合、定義されているすべてのメモリプールの状態を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

ID=aaaa FRBCNT=bbbbbbbb WTSK=cccc

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa	：	メモリプールID
bbbbbbbb	：	空き領域のメモリブロック数
cccc	：	待ちタスクID（待ちタスクが無い場合は"0000"を表示します）

本コマンドはmpl_stsシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始メモリプールID>終了メモリプールID（範囲指定時）
2	*** I34:UNDEFINED OBJECT	メモリプールを「未使用」でHI-SH7を構築している
3	E_NOEXS	メモリプールIDエラー ・ID範囲外（メモリプールID<0, メモリプールID>最大メモリプールID*1） ・予約ID（メモリプールID=0）

【注】*1　最大メモリプールIDは、HI-SH7カーネルセットアップテーブルのhi_maxmplidに定義した値です。

【注意事項】

(1)　本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

MPLDEF, GET_BLK, REL_BLK

【使　用　例】

(1) メモリプールID=1の状態を表示します。

```
#MPL_STS 1(RET)
ID=0001 FRBCNT=00000020 WTSK=0000
#
```

(2) メモリプールID=1, 2, 3の定義情報を表示します。

```
#MPL_STS 1,2,3(RET)
ID=0001 FRBCNT=00000020 WTSK=0000, ID=0002 FRBCNT=0000000A WTSK=0000,
ID=0003 FRBCNT=00000000 WTSK=0002
#
```

(3) メモリプールID=1から4の定義情報を表示します。

```
#MPL_STS 1:4(RET)
ID=0001 FRBCNT=00000020 WTSK=0000, ID=0002 FRBCNT=0000000A WTSK=0000,
ID=0003 FRBCNT=00000000 WTSK=0002, ID=0004 FRBCNT=0000002D WTSK=0000
#
```

(4) すべてのメモリプールの定義情報を表示します。

```
#MPL_STS(RET)
ID=0001 FRBCNT=00000020 WTSK=0000, ID=0002 FRBCNT=0000000A WTSK=0000,
ID=0003 FRBCNT=00000000 WTSK=0003, ID=0004 FRBCNT=0000002D WTSK=0000,
ID=0005 FRBCNT=00000000 WTSK=0002
#
```

３．３．３０	POL_FLG	POLl eventflag
	POL	イベントフラグを得る

オンライン

【コマンドフォーマット】

POL_FLG△<イベントフラグID>△<ビットパターン>△<待ちモード>
[,<イベントフラグID>△<ビットパターン>△<待ちモード>...](RET)

<イベントフラグID> ： 対象イベントフラグID（H'0001～H'03FF）
<ビットパターン> ： 待ちビットパターン
<待ちモード> ： {AND待ち ｜ OR待ち} ｜ クリア指定
AND待ち ： H'00000000
OR待ち ： H'00000002
クリア指定 ： H'00000001

【説　明】

指定したイベントフラグが、待ちビットパターンと待ちモードで指定した条件にセットされていれば、そのときのイベントフラグの値を表示します。この場合、クリア指定があると、イベントフラグの値はクリアされます。

一方、指定した条件にセットされていなければ、応答メッセージ"E_PLFAIL"を表示します。この場合、クリア指定があっても、イベントフラグの値はクリアされません。

パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合
ID=aaaa FLGPTN=bbbbbbbb

(2) 正常に処理されなかった場合
aaaa=<応答メッセージ>

aaaa ： イベントフラグID
bbbbbbbb ： イベントフラグのビットパターン
（クリア指定がある場合はクリアする前の値）

本コマンドはpol_flgシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I34:UNDEFINED OBJECT	イベントフラグを「未使用」でHI-SH7を構築している
2	E_NOEXS	イベントフラグIDエラー ・ID範囲外（イベントフラグID<0, イベントフラグID>最大イベントフラグID[*1]） ・予約ID（イベントフラグID=0）
3	E_PAR	パラメータエラー ・待ちパターン、待ちモードが不正である
4	E_PLFAIL	ポーリング失敗 （指定した条件でイベントフラグがセットされていません）

【注】*1　最大イベントフラグIDは、HI-SH7カーネルセットアップテーブルのhi_maxflgidに定義した値です。

【関連コマンド】

FLG_STS, CLR_FLG, SET_FLG

【使　用　例】

(1)　イベントフラグID=1のイベントを以下の条件で得ます。
待ちビットパターン：H'F0F0F0F0, 待ちモード：AND待ち（クリア指定あり）

```
#POL_FLG 1 F0F0F0F0 1(RET)
ID=0001 FLGPTN=F2F0FFF5
#
```

(2)　イベントフラグID=1, 2, 3のイベントを以下の条件で得ます。

・イベントフラグID=1
待ちビットパターン：H'11111111,　　待ちモード：AND待ち（クリア指定あり）

・イベントフラグID=2
待ちビットパターン：H'FFFFFFFF,　　待ちモード：OR待ち（クリア指定なし）

・イベントフラグID=3
待ちビットパターン：H'00000007,　　待ちモード：AND待ち（クリア指定なし）

```
#POL_FLG 1 11111111 1,2 FFFFFFFF 2,3 7 0(RET)
0001=E_PLFAIL, ID=0002 FLGPTN=00000001,
ID=0003 FLGPTN=04001087
#
```

３．３．３１	RCV_MSG	ReCeiVe message
	RCV	メイルボックスから受信する

オンライン

【コマンドフォーマット】

RCV_MSG△<メイルボックスID>[,<メイルボックスID>...](RET)

<メイルボックスID> ： 対象メイルボックスID（H'0001～H'03FF）

【説　　明】

指定したメイルボックスからメッセージを受信し、受信したメッセージの先頭アドレスを表示します。メイルボックスにメッセージが無い場合には、応答メッセージ"E_PLFAIL"を表示します。

メイルボックスIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

ID=aaaa MSG=bbbbbbbb

(2) 正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa ： メイルボックスID
bbbbbbbb ： メッセージの先頭アドレス

本コマンドはprcv_msgシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	E_NOEXS	メイルボックスIDエラー ・ID範囲外（メイルボックスID<0, メイルボックスID>最大メイルボックスID[*1]） ・予約ID（メイルボックスID=0）
2	ED_DBGID	メイルボックスIDエラー （指定したメイルボックスは本デバッガで使用している）
3	E_PLFAIL	ポーリング失敗（メイルボックスにメッセージがありません）

【注】*1 最大メイルボックスIDは、HI-SH7カーネルセットアップテーブルのhi_maxmbxidに定義した値です。

【関連コマンド】

MBX_STS, SND_MSG

【使　用　例】

(1)　メイルボックスID=1からメッセージを受信します。

```
#RCV_MSG 1(RET)
ID=0001 MSG=00001000
#
```

(2)　メイルボックスID=1,2,3からメッセージを受信します。

```
#RCV_MSG 1,2,3(RET)
0001=E_PLFAIL, ID=0002 MSG=00002000,
ID=0003 MSG=00003000
#
```

３．３．３２	REGISTER	Register
	R	全レジスタ内容の表示

オフライン／オンライン

【コマンドフォーマット】

REGISTER(RET) ： CPU指定（オフラインのみ）
REGISTER;<タスクID>(RET) ： タスク指定

【説　　明】

本コマンドは、CPUの全レジスタ、またはタスクが保有している全レジスタを表示します。
CPU指定は、オフラインモードのみ実行することができます。

(1) CPU指定（オフラインモードのみ）
CPUの全レジスタ内容を表示します。
:REGISTER(RET)

(2) タスク指定
指定したタスクの全レジスタ内容（VBRレジスタは含まれません）を表示します。
#REGISTER;<タスクID>(RET)

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー ・オンラインモードのときCPU指定した
2	*** I37:TASK IS NO EXIST	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	*** I38:TASK IS DORMANT	指定したタスクは休止（DORMANT）状態である
4	*** I39:TASK IS STACK WAIT	指定したタスクは共有スタック待ち状態である
5	*** I40:TASK IS RUN	指定したタスクは実行（RUN）状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【注意事項】

(1) 本コマンドのタスク指定は、オフラインモードでも実行することができますが、以下の点に注意してください。
・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。
・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

.<レジスタ名>

【使　用　例】

(1)　CPUの全レジスタ内容を表示します。

```
:R(RET)
 PC=00003100 SR=000000F0:--IIII----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7   00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
:
```

(2)　タスク指定

タスクID=1の全レジスタ内容を表示します。

```
#R;1(RET)
TSKID=0001
 PC=00005000 SR=00000000:----------
 PR=00004300 GBR=05FFFEC0
 MACH=00000000 MACL=00000000
 R0-7   00000069 00000000 00005D32 00005932 0FFFE3E8 00000301 00000000 00000000
 R8-15 00000000 00003000 00000000 00000000 00000000 00000000 00000000 0FFFE1F0
#
```

３．３．３３	REL_BLK	RELease memory Block
	RELB	固定長メモリブロックを返却する

オンライン

【コマンドフォーマット】

REL_BLK△<メモリプールID>△<先頭アドレス>[,<メモリプールID>△<先頭アドレス>...](RET)

<メモリプールID>　：　対象メモリプールID（H'0001～H'03FF）
<先頭アドレス>　　：　返却するメモリブロックの先頭アドレス

【説　　明】

指定したメモリプールへメモリブロックを返却します。

メモリブロックの先頭アドレスには、GET_BLKコマンドやget_blk, pget_blkシステムコールで獲得したメモリブロックと同じアドレスを指定してください。異なるアドレスを指定すると、"E_ILBLK"を表示します。また、すでに返却したメモリブロックを指定した場合にも、"E_ILBLK"を表示します。

パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　メモリプールID

本コマンドはrel_blkシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	*** I34:UNDEFINED OBJECT	メモリプールを「未使用」でHI-SH7を構築している
3	E_NOEXS	メモリプールIDエラー ・ID範囲外（メモリプールID<0, メモリプールID>最大メモリプールID[*1]） ・予約ID（メモリプールID=0）
4	E_ILADR	アドレスエラー（アドレスが0または4の倍数でない）
5	E_ILBLK	不正メモリブロックの返却

【注】*1　最大メモリプールIDは、HI-SH7カーネルセットアップテーブルのhi_maxmplidに定義した値です。

【関連コマンド】

GET_BLK, MPLDEF, MPL_STS

【使　用　例】

(1)　メモリプールID=1のメモリプールへメモリブロックを返却します。

```
#REL_BLK 1 FFFE100(RET)
0001=E_OK
#
```

(2)　メモリプールID=1, 2, 3のメモリプールへ、メモリブロックを返却します。

```
#REL_BLK 10FFFE110,2 FFFE200,3 FFFE400(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

３．３．３４	REL_WAI	RELease Wait
	RELW	タスクの待ち状態を強制解除する

オンライン

【コマンドフォーマット】

REL_WAI△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクが何らかの待ち（WAIT）状態（強制待ち（SUSPEND）状態は含まれません）の場合、それを強制的に解除します。二重待ち（WAIT-SUSPEND）状態のタスクは、強制待ち（SUSPEND）状態へ移行します。待ち状態を解除したタスクに対しては、エラーコードとしてE_RLWAIが返されます。

指定したタスクが待ち状態でない、または共有スタック待ち状態である場合は、"E_NOWAI"を表示します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドはrel_waiシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_NOWAI	・タスクが待ち（WAIT）状態でない ・タスクが共有スタック待ち状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

ISTATUS, TSK_STS

【使　用　例】

(1)　タスクID=1のタスクの待ち状態を強制的に解除します。

```
#REL_WAI 1(RET)
0001=E_OK
#
```

(2)　タスクID=1, 2, 3のタスクの待ち状態を強制的に解除します。

```
#REL_WAI 1,2,3(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

３．３．３５	REQ_SEM	REQuest semaphore
	REQ	セマフォ資源を得る

オンライン

【コマンドフォーマット】

REQ_SEM△<セマフォID>[,<セマフォID>...](RET)

<セマフォID>　　：　対象セマフォID（H'0001～H'03FF）

【説　　明】

指定したセマフォのカウント値を1減らします。セマフォのカウント値が0の場合は、セマフォのカウント値を変更せずに、"E_PLFAIL"を表示します。

セマフォIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　メモリプールID

本コマンドはpreq_semシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	*** I34:UNDEFINED OBJECT	セマフォを「未使用」でHI-SH7を構築している
3	E_NOEXS	セマフォIDエラー ・ID範囲外（セマフォID<0, セマフォID>最大セマフォID[*1]） ・予約ID（セマフォID=0）
4	E_PLFAIL	ポーリング失敗 （セマフォのカウント値が0である）

【注】*1　最大セマフォIDは、HI-SH7カーネルセットアップテーブルのhi_maxsemidに定義した値です。

【関連コマンド】

SIG_SEM, SEM_STS

【使　用　例】

(1)　セマフォID=1のセマフォのカウント値を1減らします。

```
#REQ_SEM 1(RET)
0001=E_OK
#
```

(2)　セマフォID=1,2,3のセマフォのカウント値を1減らします。

```
#REQ_SEM 1,2,3(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

RESET

３．３．３６	RESET	ReSet
	RS	CPUをリセットする

オフライン

【コマンドフォーマット】

RESET(RET)

【説　　明】

CPUをリセットします。リセットにより、CPUの各レジスタの値は次のようになります。
なお、リセット後、MCU初期化ルーチンを実行します。

R0-14	：	リセット前の値を保持	VBR	：	H'00000000
R15	：	マニュアルリセットベクタの値	GBR	：	リセット前の値を保持
PC	：	マニュアルリセットベクタの値	MACH	：	リセット前の値を保持
SR	：	H'000000F0（IビットのみB'1111）	MACL	：	リセット前の値を保持
PR	：	リセット前の値を保持			

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）

【注意事項】

(1)　本コマンドは、パワーオンリセットベクタの値ではなく、マニュアルリセットベクタの値を参照します。

(2)　本コマンドは、ソフトウェアによって、レジスタを初期化しているだけです。したがって、内蔵I/Oや周辺回路のリセットは行ないません。HI-SH7・MCU初期化ルーチンで内蔵I/Oや周辺回路の初期化を行なってください。

【使　用　例】

(1)　CPUをリセットします。

```
:RS(RET)
 RESET IN BY HI-SH7 DEBUGGER

 SH7000 HI-SH7 DEBUGGER (HS0700IRCN1SMB) Ver n.m
 Copyright (C) Hitachi, Ltd. 1994
 Licensed Material of Hitachi, Ltd.
:
```

３．３．３７	ROT_RDQ	ROTate ready queue
	ROT	タスクのレディーキューを回転する

オンライン

【コマンドフォーマット】

ROT_RDQ△<タスク優先度>(RET)

<タスク優先度>　　：　対象タスク優先度（H'02～H'FF）

【説　　明】

指定したタスク優先度のレディーキューを回転します。すなわち、そのタスク優先度のレディーキューの先頭につながれているタスクをレディーキューの最後尾につなぎかえ、同一タスク優先度のタスクに実行を切り換えます。指定したタスク優先度のレディーキューにタスクが無い場合は何も行ないません。

なお、タスク優先度に0または1を指定することはできません。これは、本デバッガのタスク（オンラインデバッガタスク、デバッガ用コンソールタスク）を最高優先度（タスク優先度=1）で動作させているためです。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

```
E_OK
```

(2)　正常に処理されなかった場合

<応答メッセージ>

本コマンドはrot_rdqシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_TPRI	タスク優先度エラー（タスク優先度<0, タスク優先度>最大タスク優先度*1）
3	ED_DBGTPRI	タスク優先度エラー（タスク優先度=0, 1：本デバッガで予約しています）

【注】*1　最大タスク優先度は、HI-SH7カーネルセットアップテーブルのhi_maxtskpriに定義した値です。

【関連コマンド】

CHG_PRI, CRE_TSK

【使　用　例】

(1)　タスク優先度=3のレディーキューを回転します。

```
#ROT_RDQ 3(RET)
E_OK
#
```

３．３．３８	RSM_TSK	ReSuMe task
	RSM	強制待ち状態のタスクを再開する

オンライン

【コマンドフォーマット】

RSM_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクが強制待ち（SUSPEND）状態の場合、強制待ち状態を解除し実行可能（READY）状態へ移行します。指定したタスクが二重待ち（WAIT-SUSPEND）状態の場合は、待ち（WAIT）状態へ移行します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドはrsm_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_NOSUS	タスクが強制待ち（SUSPEND）状態でない

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

SUS_TSK, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1の強制待ち状態を解除します。

```
#RSM_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1,2,3の強制待ち状態を解除します。

```
#RSM_TSK 1,2,3(RET)
0001=E_NOSUS, 0002=E_OK, 0003=E_OK
#
```

３．３．３９	RUN_TSK	RUN task
	RUN	タスクのデバッグ待ち状態を解除する

オンライン

【コマンドフォーマット】

RUN_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクのデバッグ待ち状態を解除します。タスクのデバッグ待ち状態へは、STP_TSKコマンドにより遷移します。

待ち（WAIT）状態のタスクに本コマンドを実行しても、その待ち状態は解除されません。

デバッグ待ち状態でないタスクに対して、本コマンドを実行した場合は"E_OBJ"を表示します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

表３－８に各状態のタスクに対して本コマンドを実行した場合のタスク遷移を示します。なお、デバッグ待ち状態について詳細は、**「３．３．５０　STP_TSK」**を参照してください。

表３－８　RUN_TSKコマンド実行によるタスク遷移

項番	タスクの状態	RUN_TSKコマンド実行によるタスク遷移
1	休止状態かつデバッグ待ち状態	休止状態に遷移する
2	デバッグ待ち状態	実行状態、または実行可能状態に遷移する
3	待ち状態かつデバッグ待ち状態	待ち状態に遷移する
4	強制待ち状態かつデバッグ待ち状態	強制待ち状態に遷移する
5	二重待ち状態かつデバッグ待ち状態	二重待ち状態に遷移する
6	共有スタック待ち状態かつデバッグ待ち状態	共有スタック待ち状態に遷移する

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_OBJ	タスクがデバッグ待ち状態でない

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

STP_TSK, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1のデバッグ待ち状態を解除します。

```
#RUN_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1, 2, 3のデバッグ待ち状態を解除します。

```
#RUN_TSK 1,2,3(RET)
0001=E_OBJ, 0002=E_OK, 0003=E_OK
#
```

３．３．４０	SAVE	SaVe
	SV	ファイルへのセーブ

オフライン

【コマンドフォーマット】

SAVE△<ファイル名>△<先頭アドレス>{△<最終アドレス>｜△@<バイト数>}(RET)

<ファイル名>　：　セーブするファイル名
<先頭アドレス>　：　セーブするメモリの先頭アドレス
<最終アドレス>　：　セーブするメモリの最終アドレス
<バイト数>　：　セーブするメモリのバイト数

【説　　明】

指定したメモリ領域の内容を、指定したファイル名でホストコンピュータへセーブします。

セーブするロードモジュールのタイプは、Sタイプロードモジュールです。SYSROFタイプでセーブすることはできません。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常にセーブした場合

```
TOP     ADDRESS = aaaaaaaa
BOTTOM  ADDRESS = bbbbbbbb
SAVE  NORMAL  END
```

aaaaaaaa　：　セーブしたメモリ先頭アドレス
bbbbbbbb　：　セーブしたメモリ最終アドレス

(2)　セーブ中に異常が発生した場合

<応答メッセージ>

本コマンドを使用するには、ホストコンピュータとHシリーズインタフェースソフトが必要となります。ホストコンピュータとは、Hシリーズインタフェースソフトを実行して、本デバッガの端末として動作させているコンピュータのことです。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）

【注意事項】

(1)　本コマンドは、セーブ後にベリファイしません。必要に応じて、VERIFYコマンドを使用して、ベリファイしてください。

(2)　ホストコンピュータとの転送速度は9600BPSにしてください。

【関連コマンド】

LOAD, VERIFY

【使　用　例】

(1)　H'1000番地からH'1FFF番地のメモリ内容を、"SAMPLE.MOT"のファイル名でセーブします。

```
:SV SAMPLE.MOT 1000 1FFF(RET)
 TOP     ADDRESS = 00001000
 BOTTOM ADDRESS = 00001FFF
 SAVE NORMAL END
:
```

(2)　H'1000番地からH'FFFバイト分のメモリ内容（H'1000～H'1FFF）を、"SAMPLE.MOT"のファイル名でセーブします。

```
:SV SAMPLE.MOT 1000 @FFF(RET)
 TOP     ADDRESS = 00001000
 BOTTOM ADDRESS = 00001FFF
 SAVE NORMAL END
:
```

３．３．４１	SEM_STS	SEMaphore status
	SEM	セマフォの状態を表示する

オフライン／オンライン

【コマンドフォーマット】

SEM_STS[△<セマフォID>[{,<セマフォID>... ¦ :<セマフォID>}]](RET)

<セマフォID>　　：　対象セマフォID（H'0001～H'03FF）

【説　　明】

指定したセマフォの状態を参照して、対象セマフォの現在のセマフォカウント値と、待ち行列の先頭のタスクIDを表示します。

セマフォIDは8個まで繰り返し指定することができます。セマフォIDを省略した場合、定義されているすべてのセマフォの状態を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

ID=aaaa SEMCNT=bbbb WTSK=cccc

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　：　セマフォID
bbbb　：　セマフォカウント値
cccc　：　待ちタスクID
（待ちタスクが無い場合は"0000"を表示します）

本コマンドはsem_stsシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始セマフォID>終了セマフォID（範囲指定時）
2	*** I34:UNDEFINED OBJECT	セマフォを「未使用」でHI-SH7を構築している
3	E_NOEXS	セマフォIDエラー ・ID範囲外（セマフォID<0, セマフォID>最大セマフォID[*1]） ・予約ID（セマフォID=0）

【注】*1　最大セマフォIDは、HI-SH7カーネルセットアップテーブルのhi_maxsemidに定義した値です。

【注意事項】

(1) 本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。

・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。

・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

REQ_SEM, SIG_SEM

【使　用　例】

(1) セマフォID=1の状態を表示します。

```
#SEM_STS 1(RET)
ID=0001 SEMCNT=0001 WTSK=0000
#
```

(2) セマフォID=1, 2, 3の状態を表示します。

```
#SEM_STS 1,2,3(RET)
ID=0001 SEMCNT=0001 WTSK=0000, ID=0002 SEMCNT=0000 WTSK=0002,
ID=0003 SEMCNT=0005 WTSK=0000
#
```

(3) セマフォID=1から4の状態を表示します。

```
#SEM_STS 1:4(RET)
ID=0001 SEMCNT=0001 WTSK=0000, ID=0002 SEMCNT=0000 WTSK=0002,
ID=0003 SEMCNT=0005 WTSK=0000, ID=0004 SEMCNT=0000 WTSK=000A
#
```

(4) すべてのセマフォの状態を表示します。

```
#SEM_STS(RET)
ID=0001 SEMCNT=0001 WTSK=0000, ID=0002 SEMCNT=0000 WTSK=0002,
ID=0003 SEMCNT=0005 WTSK=0000, ID=0004 SEMCNT=0000 WTSK=000A,
ID=0005 SEMCNT=FFFF WTSK=0000
#
```

３．３．４２	SET_FLG	SET eventFlag
	SETF	イベントフラグをセットする

オンライン

【コマンドフォーマット】

SET_FLG△<イベントフラグID>△<ビットパターン>[,<イベントフラグID>△<ビットパターン>...](RET)

<イベントフラグID> ： 対象イベントフラグID（H'0001～H'03FF）
<ビットパターン> ： セットするビットパターン（H'00000000～H'FFFFFFFF）

【説　明】

指定したイベントフラグに対して、セットするビットパターンの値で論理和（OR）をとってセットします。イベントフラグ値の変更の結果、そのイベントフラグを待っていたタスクの待ち解除条件を満たすようになれば、そのタスクは実行可能（READY）状態へ遷移します。

パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

aaaa=E_OK

(2) 正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa ： イベントフラグID

本コマンドはset_flgシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	*** I34:UNDEFINED OBJECT	イベントフラグを「未使用」でHI-SH7を構築している
3	E_NOEXS	イベントフラグIDエラー ・ID範囲外（イベントフラグID<0, イベントフラグID>最大イベントフラグID[*1]） ・予約ID（イベントフラグID=0）

【注】*1 最大イベントフラグIDは、HI-SH7カーネルセットアップテーブルのhi_maxflgidに定義した値です。

【関連コマンド】

CLR_FLG, FLG_STS

【使　用　例】

(1)　イベントフラグID=H'000Aに対して、H'F0F0F0F0のビットパターンでイベントフラグをセットします。

```
#SET_FLG A F0F0F0F0(RET)
000A=E_OK
#
```

(2)　イベントフラグID=1, 2, 3に対して、それぞれH'F0, H'1111, H'FFFFFFFEのビットパターンでイベントフラグをセットします。

```
#SET_FLG 1 F0,2 1111,3 FFFFFFFE(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

３．３．４３	SET_TIM	SET Time
	SETT	HI-SH7システムクロックを設定する

オンライン

【コマンドフォーマット】

SET_TIM[△<年月日時刻データの上位>]△<年月日時刻データの下位>(RET)

【説　　明】

指定した年月日時刻データを、HI-SH7カーネルが保持しているシステムクロックに設定します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

E_OK

(2)　正常に処理されなかった場合

<応答メッセージ>

本コマンドはset_timシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_TNOSPT	タイマサポートなし
3	E_ILTIME	年月日時刻データ不正（年月日時刻データに負の値を指定した）

【関連コマンド】

GET_TIM

【使　用　例】

(1)　現在の年月日時刻データにH'5を設定します。（年月日時刻データの上位にはH'0、下位にはH'5を設定します）

```
#SET_TIM 5(RET)
E_OK
#
```

(2)　現在の年月日時刻データにH'7FFFFFFFFFを設定します。（年月日時刻データの上位にはH'7F、下位にはH'FFFFFFFFを設定します）

```
#SET_TIM 7F FFFFFFFF(RET)
E_OK
#
```

３．３．４４	SIG_SEM	SIGnal semaphore
	SIG	セマフォに対する信号操作（Ｖ命令）

オンライン

【コマンドフォーマット】

SIG_SEM△<セマフォID>[,<セマフォID>...](RET)

<セマフォID>　　：　対象セマフォID（H'0001～H'03FF）

【説　　明】

指定したセマフォに待ち行列がなければ、セマフォのカウント値を1増やします。そのセマフォの待ち行列にタスクがつながれているならば、そのタスクをセマフォの待ち行列からはずし、実行可能（READY）状態へ遷移します。セマフォのカウントの最大値はH'FFFFで、これを越えた場合は"E_QOVR"を表示します。

セマフォIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　セマフォID

本コマンドはsig_semシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	*** I34:UNDEFINED OBJECT	セマフォを「未使用」でHI-SH7を構築している
3	E_NOEXS	セマフォIDエラー ・ID範囲外（セマフォID<0, セマフォID>最大セマフォID[*1]） ・予約ID（セマフォID=0）
4	E_QOVR	セマフォのカウント値のオーバフロー

【注】*1　最大セマフォIDは、HI-SH7カーネルセットアップテーブルのhi_maxsemidに定義した値です。

【関連コマンド】

REQ_SEM, SEM_STS

【使　用　例】

(1)　セマフォID=1ののセマフォに対して、信号操作（V命令）を発行します。

```
#SIG_SEM 1(RET)
0001=E_OK
#
```

(2)　セマフォID=1,2,3のセマフォに対して、信号操作（V命令）を発行します。

```
#SIG_SEM 1,2,3(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

３．３．４５	SND_MSG	SeND message
	SND	メイルボックスへ送信する

オンライン

【コマンドフォーマット】

SND_MSG△<メイルボックスID>△<先頭アドレス>[<メイルボックスID>△<先頭アドレス>...](RET)

<メイルボックスID> ： 対象メイルボックスID（H'0001～H'03FF）
<先頭アドレス> ： 送信するメッセージの先頭アドレス

【説　明】

指定したメイルボックスに、指定した先頭アドレスで示されたメッセージを送信します。

メイルボックスに待ちタスクが無い場合、メッセージをメイルボックスに入れメッセージの行列につなぎます。

メイルボックスに待ちタスクがある場合、待ち行列の先頭タスクにメッセージの先頭アドレスを渡し、そのタスクを実行可能（READY）状態へ遷移します。

パラメータは4組まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

aaaa=E_OK

(2) 正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa ： メイルボックスID

本コマンドはsnd_msgシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	メイルボックスIDエラー ・ID範囲外（メイルボックスID<0, メイルボックスID>最大メイルボックスID[*1]） ・予約ID（メイルボックスID=0）
3	ED_DBGID	メイルボックスIDエラー（指定したメイルボックスは本デバッガで使用している）
4	E_ILADR	アドレスエラー（メッセージの先頭アドレスが0または4の倍数でない）
5	ED_DBGADR	アドレスエラー（メッセージの先頭アドレスがHI-SH7または本デバッガの領域である）
6	E_ILMSG	メッセージの先頭4バイトの内容が0でない

【注】*1 最大メイルボックスIDは、HI-SH7カーネルセットアップテーブルのhi_maxmbxidに定義した値です。

【関連コマンド】

MBX_STS, RCV_MSG

【使　用　例】

(1)　メイルボックスID=1のメイルボックスに、H'4000番地のメッセージを送信します。

```
#SND_MSG 1 4000(RET)
0001=E_OK
#
```

(2)　メイルボックスID=1, 2, 3のメイルボックスに、それぞれH'5000, H'5100, H'5200番地のメッセージを送信します。

```
#SND_MSG 1 5000,2 5100,3 5200(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

３．３．４６	STA_TSK	STArt task
	STA	タスクを起動する

オンライン

【コマンドフォーマット】

STA_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクを起動します。起動したタスクは、休止（DORMANT）状態から実行可能（READY）状態に移行します。

指定したタスクが休止状態でない場合、"E_NODMT"を表示します。ただし、共有スタック待ち状態のときは、"E_OK"を表示します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドはsta_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_NODMT	タスクが休止（DORMANT）状態でない

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

CRE_TSK, TER_TSK, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1のタスクを起動します。

```
#STA_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1, 2, 3のタスクを起動します。

```
#STA_TSK 1,2,3(RET)
0001=E_NODMT, 0002=E_OK, 0003=E_OK
#
```

３．３．４７	STEP	Step
	S	シングルステップ実行

オフライン

【コマンドフォーマット】

STEP[△<ステップ数>](RET)

<ステップ数>　　：　実行するステップ数（1～H'FFFF）（省略時：1）

【説　　明】

RAM領域に配置されたユーザプログラムを指定したステップ数だけ実行します。

実行する命令がTRAPA命令のときは、例外処理を1ステップで実行します。これにより、ユーザプログラムからHI-SH7カーネルへのシステムコール要求を1ステップで実行することができます。

ステップ数には1～H'FFFFの値を指定することができます。省略した場合は1ステップだけ実行します。

:STEP[△<ステップ数>](RET)

ステップ実行後、実行タスクID（または割込みレベル）、全レジスタ、アドレスと命令ニーモニック、ステップ実行終了メッセージを表示します。

(a)　<実行タスクID（または割込みレベル）>
(b)　PC=00003102 SR=00000000:----------
　　PR=00003080 GBR=05FFFEC0 VBR=00000000
　　MACH=00000000 MACL=00000000
　　R0-7　00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
　　R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
(c)　<アドレス> : <命令ニモニック>
(d)　STEP NORMAL END

(a)　：　タスク部を実行した場合、タスクID(TSKID=0001)を表示します。非タスク部を実行した場合、割込みレベル(INTRRUPT LEVEL=5)を表示します。
(b)　：　各レジスタの内容です。
(c)　：　実行した命令のアドレスとニモニックを表示します。
(d)　：　ステップ実行終了メッセージを表示します。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
2	*** I11:COUNT ERROR	ステップ数エラー（ステップ数=0）
3	*** I13:NOT RAM AREA	次命令が格納されているアドレスがRAM領域でない
4	*** I31:RESERVED ADDRESS AREA	次命令が格納されているアドレスがHI-SH7カーネルの領域である

【注意事項】

(1) 本コマンドは、次に実行すべき命令が格納されているアドレスにソフトウェア例外命令を埋め込んで実現している為、ROM領域に配置されたユーザプログラムを実行することはできません。
(2) 遅延分岐命令は、直後の命令も実行してシングルステップ終了となります。
(3) 実行する命令がTRAPA命令のときは、STEP_OVERコマンドと同じ結果になります。

【関連コマンド】

STEP_OVER, STEP_UBC, BREAK, BREAK_UBC

【使　用　例】

(1) ユーザタスクを1ステップ実行します。

```
:S(RET)
TSKID=0001
 PC=01003102 SR=00000000:----------
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003100  MOV       R0,R1
 STEP NORMAL END
:
```

(2) ユーザ割込みハンドラ（割込みレベル=5）を3ステップ実行します。

```
:S 3(RET)
INTERRUPT LEVEL=5
 PC=01003102 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003100  MOV       R0,R1
 STEP COUNT=0001
INTERRUPT LEVEL=5
 PC=01003104 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000001 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003102  MOV       #01,R0
 STEP COUNT=0002
INTERRUPT LEVEL=5
 PC=01003106 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00000002 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003104  MOV       #02,R2
 STEP NORMAL END
:
```

３．３．４８	STEP_OVER	Step Over
	SO	サブルーチンステップ実行

オフライン

【コマンドフォーマット】

STEP_OVER[△<ステップ数>](RET)

<ステップ数>　　：　実行するステップ数（1～H'FFFF）（省略時：1）

【説　　明】

RAM領域に配置されたユーザプログラムを指定したステップ数だけ実行します。

実行する命令がサブルーチン命令（JSR, BSR, TRAPA）のときは、サブルーチンを1ステップで実行します。

ステップ数には1～H'FFFFの値を指定することができます。省略した場合は1ステップだけ実行します。

:STEP_OVER[△<ステップ数>](RET)

ステップ実行後、実行タスクID（または割込みレベル）、全レジスタ、アドレスと命令ニーモニック、ステップ実行終了メッセージを表示します。

(a)　<実行タスクID（または割込みレベル）>
(b)　PC=00003102 SR=00000000:----------
　　PR=00003080 GBR=05FFFEC0 VBR=00000000
　　MACH=00000000 MACL=00000000
　　R0-7　00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
　　R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
(c)　<アドレス> : <命令ニモニック>
(d)　STEP_OVER NORMAL END

(a)　：　タスク部を実行した場合、タスクID(TSKID=0001)を表示します。非タスク部を実行した場合、割込みレベル(INTRRUPT LEVEL=5)を表示します。
(b)　：　各レジスタの内容です。
(c)　：　実行した命令のアドレスとニモニックを表示します。
(d)　：　ステップ実行終了メッセージを表示します。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
2	*** I11:COUNT ERROR	ステップ数エラー（ステップ数=0）
3	*** I13:NOT RAM AREA	次命令が格納されているアドレスがRAM領域でない
4	*** I31:RESERVED ADDRESS AREA	次命令が格納されているアドレスがHI-SH7カーネルの領域である

【注意事項】

(1) 本コマンドは、次に実行すべき命令が格納されているアドレスにソフトウェア例外命令を埋め込んで実現している為、ROM領域に配置されたユーザプログラムを実行することはできません。

(2) 遅延分岐命令は、直後の命令も実行してシングルステップ終了となります。

【関連コマンド】

STEP, STEP_UBC, BREAK, BREAK_UBC

【使　用　例】

(1) ユーザタスクを1ステップ実行します。

```
:SO(RET)
TSKID=0001
 PC=01003102 SR=00000000:----------
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003100  MOV       R0,R1
 STEP_OVER NORMAL END
:
```

(2) ユーザ割込みハンドラ（割込みレベル=5）を3ステップ実行します。

```
:SO 3(RET)
INTERRUPT LEVEL=5
 PC=01003102 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003100  MOV       R0,R1
 STEP_OVER COUNT=0001
INTERRUPT LEVEL=5
 PC=01003104 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000001 00000000 00003DAC 00000003 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003102  JSR       @R2
 00003104  MOV       #03,R3
 STEP_OVER COUNT=0002
INTERRUPT LEVEL=5
 PC=01003108 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00000002 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003106  MOV       #02,R2
 STEP_OVER NORMAL END
:
```

３．３．４９	STEP_UBC	Step by Ubc
	SU	ハードウェアブレークによるシングルステップ実行

オフライン

【コマンドフォーマット】

STEP_UBC[△<ステップ数>](RET)

<ステップ数>　　：　実行するステップ数（1～H'FFFF）（省略時：1）

【説　　明】

ROM/RAM領域に配置されたユーザプログラムを指定したステップ数だけ実行します。

ステップ数には1～H'FFFFの値を指定することができます。省略した場合は1ステップだけ実行します。

:STEP_UBC[△<ステップ数>](RET)

ステップ実行後、実行タスクID（または割込みレベル）、全レジスタ、アドレスと命令ニーモニック、ステップ実行終了メッセージを表示します。

(a)　<実行タスクID（または割込みレベル）>
(b)　PC=00003102 SR=00000000:----------
　　PR=00003080 GBR=05FFFEC0 VBR=00000000
　　MACH=00000000 MACL=00000000
　　R0-7　00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
　　R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
(c)　<アドレス> : <命令ニモニック>
(d)　STEP_UBC NORMAL END

(a)　：　タスク部を実行した場合、タスクID(TSKID=0001)を表示します。非タスク部を実行した場合、割込みレベル(INTRRUPT LEVEL=5)を表示します。
(b)　：　各レジスタの内容です。
(c)　：　実行した命令のアドレスとニモニックを表示します。
(d)　：　ステップ実行終了メッセージを表示します。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
2	*** I07:INVALID MASK LEVEL	現在の割込みマスクレベル（SRレジスタのIビット）が15である
3	*** I11:COUNT ERROR	ステップ数エラー（ステップ数=0）
4	*** I31:RESERVED ADDRESS AREA	次命令が格納されているアドレスがHI-SH7カーネルの領域である

【注意事項】

(1) 本コマンドは、SHマイコン内蔵のUBC（ユーザブレークコントローラ）を使用して実現しています。UBCは、実行する命令の次の命令をCPUがフェッチすると、割込みレベル=15の割込みをCPUに対して要求します。したがって、以下の制限があります。

・割込みレベル=15の割込み処理プログラム（割込みハンドラ）をステップ実行することはできません。

・割込みを一時的に禁止する命令（LDC命令など）を実行すると、次の命令も実行してシングルステップ終了となります。

・ステップ実行中に、タイマなどの割込みが動作していると、ステップ終了後のCPUの停止位置（PCレジスタ）は、その割込みの処理プログラム内を指していることがあります。
例えば、タイマ割込み動作中にユーザタスクをステップ実行すると、CPUは、タイマ割込みを受付け、割込みハンドラを実行します。これを避けたい場合は、必要に応じて割込みをマスクしてください（.<レジスタ>コマンドでSRレジスタのIビットを1～14に変更）。

・遅延分岐命令は、直後の命令も実行してシングルステップ終了となります。

(2) 実行する命令がTRAPA命令のとき、例外処理を1ステップで実行できません。

【関連コマンド】

STEP, STEP_OVER, BREAK, BREAK_UBC

【使　用　例】

(1) ユーザタスクを1ステップ実行します。

```
:SU(RET)
TSKID=0001
 PC=01003102 SR=00000000:----------
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003100  MOV       R0,R1
 STEP_UBC NORMAL END
:
```

(2) ユーザ割込みハンドラ（割込みレベル=5）を3ステップ実行します。

```
:SU 3(RET)
INTERRUPT LEVEL=5
 PC=01003102 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003100  MOV       R0,R1
 STEP_UBC COUNT=0001
INTERRUPT LEVEL=5
 PC=01003104 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000001 00000000 00003DAC 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003102  MOV       #01,R0
 STEP_UBC COUNT=0002
INTERRUPT LEVEL=5
 PC=01003106 SR=00000050:---I-I----
 PR=00003080 GBR=05FFFEC0 VBR=00000000
 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00000002 00003D30 0FFFE3E8 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE394
 00003104  MOV       #02,R2
 STEP_UBC NORMAL END
:
```

３．３．５０	STP_TSK	SToP task
	STP	タスクをデバッグ待ち状態にする

オンライン

【コマンドフォーマット】

STP_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクの実行を中断し、デバッグ待ち状態へ遷移します。デバッグ待ち状態は、RUN_TSKコマンドによって解除されます。指定したタスクがすでにデバッグ待ち状態である場合、"E_QOVR"を表示します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

デバッグ待ち状態は、実行状態以外のすべてのタスク状態と複合します。表３－９に、各状態のタスクに対して本コマンドを実行した場合のタスク遷移を示します。

表３－９　STP_TSKコマンド実行によるタスク遷移

項番	タスクの状態	STP_TSKコマンド実行によるタスク遷移
1	休止状態	休止状態かつデバッグ待ち状態に遷移する （タスクが起動されると、ただちにデバッグ待ち状態に遷移する）
2	実行可能状態	デバッグ待ち状態に遷移する
3	待ち状態	待ち状態かつ、デバッグ待ち状態に遷移する
4	強制待ち状態	強制待ち状態かつ、デバッグ待ち状態に遷移する
5	二重待ち状態	二重待ち状態かつ、デバッグ待ち状態に遷移する
6	共有スタック待ち状態	共有スタック待ち状態かつ、デバッグ待ち状態に遷移する

図３－３にデバッグ待ち状態とSTP_TSK, RUN_TSKコマンドによる状態遷移を示します。

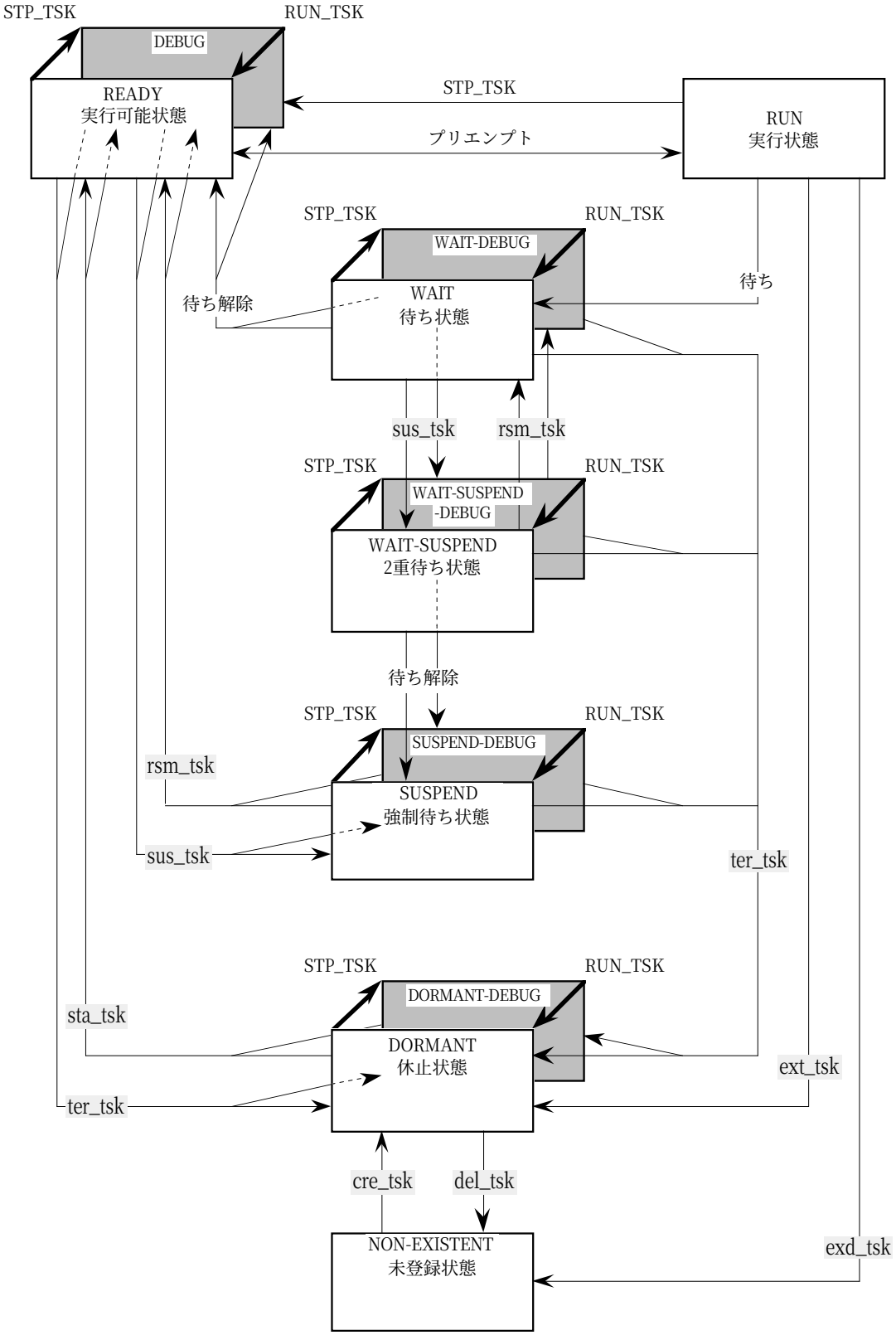


図３－３　STP_TSKおよびRUN_TSKコマンドによる状態遷移

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_QOVR	タスクがすでにデバッグ待ち状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

RUN_TSK, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1のタスクの実行を中断します。

```
#STP_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1, 2, 3のタスクの実行を中断します。

```
#STP_TSK 1,2,3(RET)
0001=E_QOVR, 0002=E_OK, 0003=E_OK
#
```

３．３．５１	SUS_TSK	SUSpend task
	SUS	タスクを強制待ち状態へ移行する

オンライン

【コマンドフォーマット】

SUS_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクを強制待ち（SUSPEND）状態へ移行します。指定したタスクがすでに強制待ち状態である場合、"E_QOVR"を表示します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドは、sus_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_DMT	タスクが休止（DORMANT）状態である
5	E_QOVR	タスクがすでに強制待ち（SUSPEND）状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

RSM_TSK, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1のタスクを強制待ち状態へ移行します。

```
#SUS_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1, 2, 3のタスクを強制待ち状態へ移行します。

```
#SUS_TSK 1,2,3(RET)
0001=E_QOVR, 0002=E_OK, 0003=E_OK
#
```

３．３．５２	SVCDEF	SVC(System Call) Define information
	SVCD	HI-SH7システムコール定義情報の表示

オフライン／オンライン

【コマンドフォーマット】

SVCDEF[△<システムコール種別>](RET)

<システムコール種別> ： システムコールの種別
T ： ﾀｽｸ部から発行可能なｼｽﾃﾑｺｰﾙの定義情報
N ： 非ﾀｽｸ部から発行可能なｼｽﾃﾑｺｰﾙの定義情報
E ： 拡張SVC
省略 ： すべてのシステムコール

【説　明】

指定したシステムコール種別の定義情報を表示します。システムコール種別を省略した場合、定義されているすべてのシステムコール定義情報（タスク部:TASK、非タスク部:NON TASK、拡張SVC:EXTENDED）を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) システムコール種別に'T'を指定した場合

aaaaaaaa bbb　　aaaaaaaa bbb　　aaaaaaaa bbb　　aaaaaaaa bbb

aaaaaaaa ： タスク部から発行可能なシステムコールの名称
bbb ： システムコールの定義状態
USE ： 使用する
NOT ： 使用しない

(2) システムコール種別に'N'を指定した場合

aaaaaaaa bbb　　aaaaaaaa bbb　　aaaaaaaa bbb　　aaaaaaaa bbb

aaaaaaaa ： 非タスク部から発行可能なシステムコールの名称
bbb ： システムコールの定義状態
USE ： 使用する
NOT ： 使用しない

(3) システムコール種別に'E'を指定した場合

FNCD=cc SVCHDR=dddddddd, FNCD=cc SVCHDR=dddddddd

cc ： 拡張SVCの機能コード
bbbbbbbb ： 拡張SVCハンドラの開始アドレス

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I01:SYNTAX ERROR	システムコール種別エラー

【関連コマンド】

SYSDEF, MPLDEF, TSKDEF

【使　用　例】

(1)　タスク部から発行可能なシステムコールの定義情報を表示します。

```
:SVCDEF T(RET)
cre_tsk  USE        sta_tsk  USE        del_tsk  USE        ext_tsk  USE
exd_tsk  USE        ter_tsk  USE        chg_pri  USE        rot_rdq  USE
rel_wai  USE        get_tid  USE        tsk_sts  USE        sus_tsk  NOT
rsm_tsk  USE        slp_tsk  USE        wai_tsk  USE        wup_tsk  USE
can_wup  USE        set_flg  USE        clr_flg  USE        wai_flg  USE
pol_flg  USE        flg_sts  USE        sig_sem  NOT        wai_sem  NOT
preq_sem NOT        sem_sts  NOT        snd_msg  USE        rcv_msg  USE
prcv_msg USE        mbx_sts  USE        chg_ims  USE        ims_sts  USE
get_blk  USE        pget_blk USE        rel_blk  USE        mpl_sts  USE
set_tim  USE        get_tim  USE        get_ver  USE
:
```

(2)　非タスク部から発行可能なシステムコールの定義情報を表示します。

```
:SVCDEF N(RET)
ista_tsk USE        ichg_pri USE        irot_rdq USE        irel_wai USE
get_tid  USE        tsk_sts  USE        isus_tsk USE        irsm_tsk USE
iwup_tsk USE        can_wup  USE        iset_flg USE        clr_flg  USE
pol_flg  USE        flg_sts  USE        isig_sem NOT        preq_sem USE
sem_sts  NOT        isnd_msg USE        prcv_msg USE        mbx_sts  USE
chg_ims  USE        ims_sts  USE        pegt_blk USE        rel_blk  USE
mpl_sts  USE        set_tim  USE        get_tim  USE        get_ver  USE
:
```

(3)　拡張ＳＶＣの定義情報を表示します。

```
:SVCDEF E(RET)
FNCD=01 SVCHDR=00010000, FNCD=02 SVCHDR=00020000,
FNCD=03 SVCHDR=00030000, FNCD=04 SVCHDR=00040000
:
```

(4) すべてのシステムコールの定義情報を表示します。

```
:SVCDEF(RET)
 TASK
cre_tsk  USE        sta_tsk  USE        del_tsk  USE        ext_tsk  USE
exd_tsk  USE        ter_tsk  USE        chg_pri  USE        rot_rdq  USE
rel_wai  USE        get_tid  USE        tsk_sts  USE        sus_tsk  NOT
rsm_tsk  USE        slp_tsk  USE        wai_tsk  USE        wup_tsk  USE
can_wup  USE        set_flg  USE        clr_flg  USE        wai_flg  USE
pol_flg  USE        flg_sts  USE        sig_sem  NOT        wai_sem  NOT
preq_sem NOT        sem_sts  NOT        snd_msg  USE        rcv_msg  USE
prcv_msg USE        mbx_sts  USE        chg_ims  USE        ims_sts  USE
get_blk  USE        pget_blk USE        rel_blk  USE        mpl_sts  USE
set_tim  USE        get_tim  USE        get_ver  USE
 NON TASK
ista_tsk USE        ichg_pri USE        irot_rdq USE        irel_wai USE
get_tid  USE        tsk_sts  USE        isus_tsk USE        irsm_tsk USE
iwup_tsk USE        can_wup  USE        iset_flg USE        clr_flg  USE
pol_flg  USE        flg_sts  USE        isig_sem NOT        preq_sem USE
sem_sts  NOT        isnd_msg USE        prcv_msg USE        mbx_sts  USE
chg_ims  USE        ims_sts  USE        pegt_blk USE        rel_blk  USE
mpl_sts  USE        set_tim  USE        get_tim  USE        get_ver  USE
 EXTENDED
FNCD=01 SVCHDR=00010000, FNCD=02 SVCHDR=00020000,
FNCD=03 SVCHDR=00030000, FNCD=04 SVCHDR=00040000
:
```

３．３．５３	SYSDEF	SYStem Define information
	SYSD	HI-SH7システム定義情報の表示

オフライン／オンライン

【コマンドフォーマット】

SYSDEF(RET)

【説　　明】

HI-SH7セットアップテーブルに定義されたシステム定義情報を表示します。
本コマンドの実行結果は、以下のフォーマットで表示します。

```
KNLMSK=aa UPPINTNST=bb LOWINTNST=c  INIHDR=dddddddd SYSDWN=eeeeeeee
CHKSUT=fff MAXTSKPRI=gg  INITSKNUM=hhhh MAXSVCCD=ii
MAXTSKID=jjjj MAXFLGID=kkkk MAXSEMID=llll MAXMBXID=mmmm MAXMPLID=nnnn
```

aa	：	カーネル割込みマスクレベル
bb	：	上位レベル割込みネスト数
c	：	下位レベル割込みネスト数
dddddddd	：	システム初期化ハンドラの開始アドレス
eeeeeeee	：	システム異常終了処理ルーチンの開始アドレス
fff	：	セットアップテーブルのエラーチェック機能
		USE　：　使用する
		NOT　：　使用しない
gg	：	最大タスク優先度
hhhh	：	初期登録タスク数
ii	：	拡張SVCの最大機能コード
jjjj	：	最大タスクID
kkkk	：	最大イベントフラグID
llll	：	最大セマフォID
mmmm	：	最大メイルボックスID
nnnn	：	最大メモリプールID

【関連コマンド】

SVCDEF, MPLDEF, TSKDEF

【使　用　例】

```
:SYSDEF(RET)
KNLMSK=0E UPPINTNST=00 LOWINTNST=2  INIHDR=000008A4 SYSDWN=00000840
CHKSUT=USE MAXTSKPRI=06  INITSKNUM=0006 MAXSVCCD=00
MAXTSKID=0006 MAXFLGID=0005 MAXSEMID=0003 MAXMBXID=0003 MAXMPLID=0002
:
```

３．３．５４	TER_TSK	TERminate task
	TER	タスクを強制的に異常終了させる

オンライン

【コマンドフォーマット】

TER_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクを強制的に異常終了させ、休止（DORMANT）状態へ移行します。

本コマンドは、タスクがそれ以前に獲得した資源を自動的に解放する機能はありません。したがって、指定したタスクが獲得している資源は、ユーザが管理（SIG_SEMコマンドなどによって解放）してください。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドは、ter_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_DMT	タスクが休止（DORMANT）状態である

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

STA_TSK, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1のタスクを強制的に異常終了させます。

```
#TER_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1,2,3のタスクを強制的に異常終了させます。

```
#TER_TSK 1,2,3(RET)
0001=E_DMT, 0002=E_OK, 0003=E_OK
#
```

３．３．５５	TSKDEF	TaSK Define information
	TSKD	タスク定義情報の表示

オフライン／オンライン

【コマンドフォーマット】

TSKDEF[△<タスクID> [{,<タスクID>... ¦ :<タスクID>}]](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定したタスクの定義情報を表示します。

タスクIDは8個まで繰り返し指定することができます。 タスクIDを省略した場合、定義されているすべてのタスク定義情報を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合（タスクが初期登録されている）

ID=aaaa ITSKPRI=bb STADR=cccccccc ISPADR=ddddddddeeeeee

(2)　正常に処理されなかった場合（タスクが初期登録されていない）

ID=aaaa :NON-EXISTENT　　ISPADR=ddddddddeeeeee

aaaa　　：　タスクID
bb　　：　初期タスク優先度
cccccccc　　：　タスク開始アドレス
dddddddd　　：　初期スタックポインタ
eeeeee　　：　共有タスクスタック機能
　　なし　：　通常タスクスタック
　　-SHARE　：　共有タスクスタック

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始オブジェクトID>終了オブジェクトID（範囲指定時）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0）

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

TSK_STS

【使　用　例】

(1) タスクID=1の定義情報を表示します。

```
:TSKDEF 1(RET)
ID=0001 ITSKPRI=01 STADR=000039EC ISPADR=0FFFE3D4
:
```

(2) タスクID=1, 2, 3の定義情報を表示します。

```
:TSKDEF 1,2,3(RET)
ID=0001 ITSKPRI=01 STADR=000039EC ISPADR=0FFFE3D4
ID=0002 ITSKPRI=02 STADR=00003C00 ISPADR=0FFFE4D4
ID=0003 ITSKPRI=05 STADR=00003E44 ISPADR=0FFFE5D4
:
```

(3) タスクID=1から4の定義情報を表示します。

```
:TSKDEF 1:4(RET)
ID=0001 ITSKPRI=01 STADR=000039EC ISPADR=0FFFE3D4
ID=0002 ITSKPRI=02 STADR=00003C00 ISPADR=0FFFE4D4
ID=0003 ITSKPRI=05 STADR=00003E44 ISPADR=0FFFE5D4
ID=0004 :NON-EXISTENT             ISPADR=0FFFE6D4-SHARE
:
```

(4) すべてのタスクの定義情報を表示します。

```
:TSKDEF(RET)
ID=0001 ITSKPRI=01 STADR=000039EC ISPADR=0FFFE3D4
ID=0002 ITSKPRI=02 STADR=00003C00 ISPADR=0FFFE4D4
ID=0003 ITSKPRI=05 STADR=00003E44 ISPADR=0FFFE5D4
ID=0004 :NON-EXISTENT             ISPADR=0FFFE6D4-SHARE
ID=0005 ITSKPRI=05 STADR=00003EE8 ISPADR=0FFFE6D4-SHARE
:
```

３．３．５６	TSK_STS	TaSK status
	TSK	タスクの状態を表示する

オフライン／オンライン

【コマンドフォーマット】

TSK_STS[△<タスクID> [{,<タスクID>... ¦ :<タスクID>}]](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　明】

指定したタスクの状態に関する情報とレジスタ情報を表示します。実行状態のタスクのレジスタ情報は表示しません。

タスクIDは8個まで繰り返し指定することができます。タスクIDを省略した場合、定義されているすべてのタスクの状態を表示します。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常に処理された場合

```
ID=aaaa PRI=bb ccc-c-c WID=dddd eeeeeeee-ffff WUP=gg TMOUT=hhhhhhhh
 ITSKPRI=ii ISTADR=jjjjjjjj ISPADR=kkkkkkkk
 PC=llllllll SR=llllllll GBR=llllllll PR=llllllll MACH=llllllll MACL=llllllll
 R0-7  llllllll llllllll llllllll llllllll llllllll llllllll llllllll llllllll
 R8-15 llllllll llllllll llllllll llllllll llllllll llllllll llllllll llllllll
```

(2) 正常に処理されなかった場合

aaaa=＜応答ﾒｯｾｰｼﾞ＞

aaaa　：　タスクID
bb　：　現在のタスク優先度
ccc-c-c　：　タスク状態

RUN	：	実行状態
RDY	：	実行可能状態
SUS	：	強制待ち状態
DMT	：	休止状態
STK	：	共有スタックによる起動待ち状態
WAI	：	wai_tskシステムコールによる待ち状態
SLP	：	slp_tskシステムコールによる待ち状態、または wai_tskシステムコールでtmout=-1指定による待ち状態
FLG	：	wai_flgシステムコールによる待ち状態
SEM	：	wai_semシステムコールによる待ち状態
MBX	：	rcv_msgシステムコールによる待ち状態
MPL	：	get_blkシステムコールによる待ち状態
DBG	：	デバッグ待ち状態
***	：	状態遷移中または未登録状態（オフラインのみ表示）
ccc-S	：	二重待ち状態
ccc-D	：	待ち状態かつデバッグ待ち状態
ccc-S-D	：	二重待ち状態かつデバッグ待ち状態

dddd　：　待ち状態での対象ID（待ち状態でない場合は“0000”）
eeeeeeee　：　イベントフラグの待ちビットパターン
（wai_flgシステムコールによる待ち状態でない場合は"********"）
ffff　：　イベントフラグの待ちモード
（wai_flgシステムコールによる待ち状態でない場合は"****"）
AND　：　AND待ち
OR　：　OR待ち
ANDC　：　AND待ち＋クリア指定
ORC　：　OR待ち＋クリア指定
gg　：　タスクの起床要求カウント
hhhhhhhh　：　wai_tskシステムコールのタイムアウトするまでの残り時間
（wai_tskシステムコールによる待ち状態でない場合は"00000000"）
ii　：　初期タスク優先度
jjjjjjjj　：　タスク開始アドレス
kkkkkkkk　：　初期スタックポインタ
llllllll　：　タスクの各レジスタ値

なお、指定したタスクが実行（RUN）状態、休止（DORMANT）状態、共有スタックによる起動待ち状態、またはカーネル実行中である場合には、タスクID、現在のタスク優先度、タスク状態、初期タスク優先度、タスク開始アドレス、初期スタックポインタだけを表示します。

【応答メッセージ】

項番	応答メッセージ	内　容
1	*** I09:PARAMETER ERROR	パラメータエラー ・開始オブジェクトID>終了オブジェクトID（範囲指定時）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【注意事項】

(1)　本コマンドは、オフラインモードでも実行することができますが、以下の点に注意してください。
・HI-SH7システムを一度も起動していない状態のオフラインモードでは、本コマンドによる表示内容は不定です。
・カーネル処理中に停止した状態のオフラインモードでは、本コマンドによる表示内容は正しくない場合があります。

【関連コマンド】

TSKDEF, ISTATUS

【使　用　例】

(1)　タスクID=1の状態を表示します。

```
#TSK_STS 1(RET)
ID=0001 PRI=01 MBX      WID=0001 ********-**** WUP=00 TMOUT=00000000
 ITSKPRI=01 STADR=00003AC8 ISPADR=0FFFE924
 PC=00003ADC SR=00000000 GBR=05FFFEC0 PR=00003ADC MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003E48 00003DB4 0FFFE918 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE8C4
#
```

(2)　タスクID=1,2,3の状態を表示します。

```
#TSK_STS 1,2,3(RET)
ID=0001 PRI=01 MBX      WID=0001 ********-**** WUP=00 TMOUT=00000000
 ITSKPRI=01 STADR=00003AC8 ISPADR=0FFFE924
 PC=00003ADC SR=00000000 GBR=05FFFEC0 PR=00003ADC MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003E48 00003DB4 0FFFE918 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE8C4
ID=0002 PRI=02 DMT
 ITSKPRI=02 STADR=00003FE4 ISPADR=0FFFEA24
ID=0003 PRI=03 DMT
 ITSKPRI=03 STADR=0000424C ISPADR=0FFFEB24
#
```

(3)　タスクID=1から4の状態を表示します。

```
#TSK_STS 1:4(RET)
ID=0001 PRI=01 MBX      WID=0001 ********-**** WUP=00 TMOUT=00000000
 ITSKPRI=01 STADR=00003AC8 ISPADR=0FFFE924
 PC=00003ADC SR=00000000 GBR=05FFFEC0 PR=00003ADC MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003E48 00003DB4 0FFFE918 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE8C4
ID=0002 PRI=02 DMT
 ITSKPRI=02 STADR=00003FE4 ISPADR=0FFFEA24
ID=0003 PRI=03 DMT
 ITSKPRI=03 STADR=0000424C ISPADR=0FFFEB24
ID=0004 PRI=04 SUS      WID=0000 ********-**** WUP=00 TMOUT=00000000
 ITSKPRI=04 STADR=00004320 ISPADR=0FFFEC24
 PC=00004320 SR=80000000 GBR=00000000 PR=00000000 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFEBCC
#
```

(4) すべてのタスクの状態を表示します。

```
#TSK_STS(RET)
ID=0001 PRI=01 MBX      WID=0001 ********-**** WUP=00 TMOUT=00000000
 ITSKPRI=01 STADR=00003AC8 ISPADR=0FFFE924
 PC=00003ADC SR=00000000 GBR=05FFFEC0 PR=00003ADC MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00003E48 00003DB4 0FFFE918 00000001 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFE8C4
ID=0002 PRI=02 DMT
 ITSKPRI=02 STADR=00003FE4 ISPADR=0FFFEA24
ID=0003 PRI=03 DMT
 ITSKPRI=03 STADR=0000424C ISPADR=0FFFEB24
ID=0004 PRI=04 SUS      WID=0000 ********-**** WUP=00 TMOUT=00000000
 ITSKPRI=04 STADR=00004320 ISPADR=0FFFEC24
 PC=00004320 SR=80000000 GBR=00000000 PR=00000000 MACH=00000000 MACL=00000000
 R0-7  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
 R8-15 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0FFFEBCC
#
```

３．３．５７	VERIFY	Verify
	V	ファイルとメモリのベリファイ

オフライン

【コマンドフォーマット】

VERIFY△<ファイル名>[;S](RET)

<ファイル名>　：　ロードするファイル名
S　：　Sタイプロードモジュールの指定
（省略時：SYSROFタイプロードモジュール）

【説　明】

指定したホストコンピュータのロードモジュールとメモリ内容をベリファイします。

Sパラメータを指定すると、Sタイプロードモジュール（標準ファイル拡張子：.mot）とベリファイします。省略すると、SYSROFタイプロードモジュール（標準ファイル拡張子：.abs）とベリファイします。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1) 正常にベリファイした場合

```
TOP     ADDRESS = aaaaaaaa
BOTTOM  ADDRESS = bbbbbbbb
VERIFY NORMAL END
```

(2) ベリファイエラーが発生した場合

```
 <ADDR>    <FILE> <MEM>
cccccccc   dd'd'  ee'e'
              :
TOP     ADDRESS = aaaaaaaa
BOTTOM  ADDRESS = bbbbbbbb
```

aaaaaaaa　：　ベリファイしたメモリ先頭アドレス
bbbbbbbb　：　ベリファイしたメモリ最終アドレス
cccccccc　：　ベリファイエラーが発生したアドレス
dd'd'　：　ロードモジュールのデータ（16進数とASCII）
ee'e'　：　メモリのデータ（16進数とASCII）

(3) ロード中に異常が発生した場合

<応答メッセージ>

本コマンドを使用するには、ホストコンピュータとHシリーズインタフェースソフトが必要となります。ホストコンピュータとは、Hシリーズインタフェースソフトを実行して、本デバッガの端末として動作させているコンピュータのことです。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オンラインモードでは実行できません）
2	*** I20:TIME OUT ERROR	タイムアウトエラー（シリアル回線が正しく接続されていません）
3	*** I21:CHECK SUM ERROR	ベリファイ中にチェックサムエラーが発生しました
4	*** I22:NOT SAME FILE	指定したロードモジュールの形式が違います
5	*** I23:VERIFY ERROR	ベリファイエラー

【注意事項】

(1)　本デバッガでは、SYSROFタイプロードモジュールに含まれるシンボル情報は無視します。
(2)　ベリファイエラーが発生した場合、最大32エントリまでの表示となります。
(3)　ホストコンピュータとの転送速度は9600BPSにしてください。

【関連コマンド】

LOAD, SAVE

【使　用　例】

(1)　SYSROFタイプのロードモジュール"TEST.ABS"とメモリ内容をベリファイします。

```
:V TEST.ABS(RET)
 TOP     ADDRESS = 00010000
 BOTTOM  ADDRESS = 0001063F
 VERIFY  NORMAL  END
:
```

(2)　Sタイプロードモジュール"TEST.MOT"とメモリ内容をベリファイします。

```
:V TEST.MOT;S(RET)
  <ADDR>    <FILE> <MEM>
 00010100   41'A'   42'B'
 00010101   31'1'   30'0'
 TOP     ADDRESS = 00010000
 BOTTOM  ADDRESS = 0001063F
 ***  I23:VERIFY  ERROR
:
```

３．３．５８	WUP_TSK	Wake UP task
	WUP	タスクを起床する

オンライン

【コマンドフォーマット】

WUP_TSK△<タスクID>[,<タスクID>...](RET)

<タスクID>　　：　対象タスクID（H'0001～H'03FF）

【説　　明】

指定した待ち状態のタスクを起床（slp_tsk, wai_tskシステムコールの発行により待ち（WAIT）状態になっているタスクを実行可能（READY）状態へ遷移）します。指定したタスクが待ち状態でない場合、この起床要求はキューイングされます。 キューイングの最大値はH'Fで、これを越えた場合は、"E_QOVR"を表示します。

タスクIDは8個まで繰り返し指定することができます。

本コマンドの実行結果は、以下のフォーマットで表示します。

(1)　正常に処理された場合

aaaa=E_OK

(2)　正常に処理されなかった場合

aaaa=<応答メッセージ>

aaaa　　：　タスクID

本コマンドは、wup_tskシステムコールを使用して実現しています。

【応答メッセージ】

項番	応答メッセージ	内　　容
1	*** I05:DEBUG MODE ERROR	デバッグモードエラー（オフラインモードでは実行できません）
2	E_NOEXS	タスクIDエラー ・ID範囲外（タスクID<0, タスクID>最大タスクID[*1]） ・予約ID（タスクID=0） ・未登録（タスクが生成されていない）
3	ED_DBGID	タスクIDエラー（指定したタスクは本デバッガで使用している）
4	E_DMT	タスクが休止（DORMANT）状態である
5	E_QOVR	起床要求のキューイングオーバーフロー

【注】*1　最大タスクIDは、HI-SH7カーネルセットアップテーブルのhi_maxtskidに定義した値です。

【関連コマンド】

CAN_WUP, TSK_STS, ISTATUS

【使　用　例】

(1)　タスクID=1のタスクを起床します。

```
#WUP_TSK 1(RET)
0001=E_OK
#
```

(2)　タスクID=1, 2, 3のタスクを起床します。

```
#WUP_TSK 1,2,3(RET)
0001=E_OK, 0002=E_OK, 0003=E_OK
#
```

４．　システムの構築

４．１　システム構築の概要

本デバッガは、次に示すシステム構築作業を行なって、ユーザシステムに組込みます。

(1)　共通ヘッダファイル（hdcommon.h）の作成
　本デバッガとHI-SH7システムで共通利用するヘッダファイルを作成します。このファイルを作成することで、システムの構築作業が軽減します。

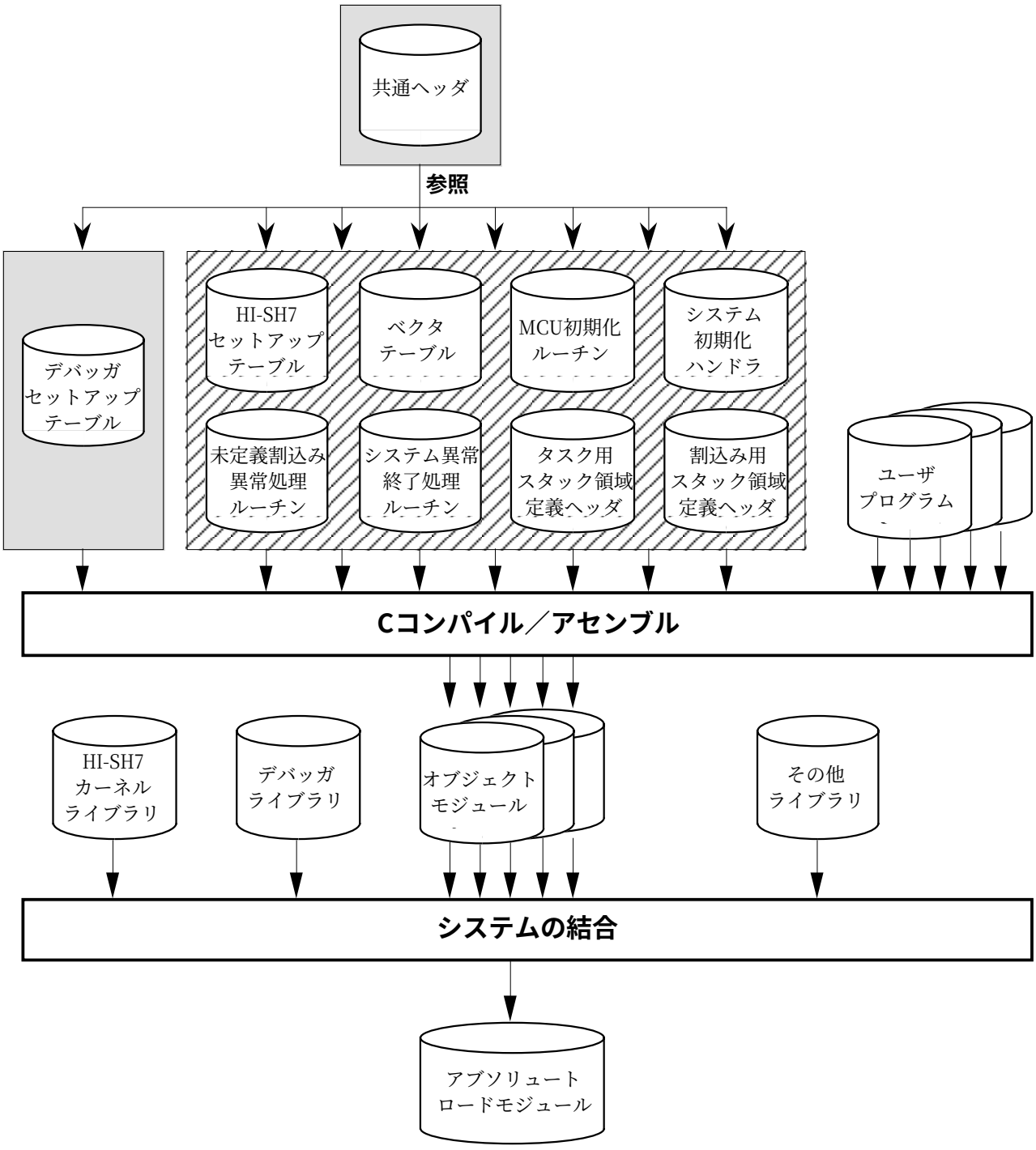
(2)　デバッガセットアップテーブル（hdsuptbl.c）の作成
　本デバッガのコマンド選択や、環境を定義するセットアップテーブルを作成します。このテーブルを作成することで、ユーザシステムに適したデバッガを組込むことができます。

(3)　HI-SH7セットアップテーブル（hisuptbl.c）の修正
　本デバッガは、HI-SH7のユーザ資源であるタスク、メイルボックスを一部使用します。したがって、本デバッガを利用できるようにするため、HI-SH7の環境を修正します。

(4)　HI-SH7システムのユーザプログラムの修正
　本デバッガを利用できるようにするため、以下のHI-SH7システムのユーザプログラムを修正します。
　・ベクタテーブル（hivcttbl.c）
　・MCU初期化ルーチン（himcuini.c）
　・システム初期化ハンドラ（hisysini.c）
　・未定義割込み異常処理ルーチン（hiintdwn.c）
　・システム異常終了処理ルーチン（hisysdwn.c）
　・タスク用スタック領域定義ヘッダ（hitskstk.h）
　・割込みハンドラ用スタック領域定義ヘッダ（hiintstk.h）

(5)　実行形式プログラムの作成
　(1)～(4)で作成したプログラムをコンパイルして、オブジェクトモジュールを作成します。次に、作成したオブジェクトモジュールと、ユーザプログラム、HI-SH7カーネルライブラリ、およびデバッガライブラリを結合して、実行可能なロードモジュールを作成します。

図４－１にシステム構築の概要を示します。



【注】 ▭ で囲んだファイルは、本製品でサンプルプログラムを提供していることを表わします。

▨ で囲んだファイルは、本製品でサンプルプログラム（HI-SH7のサンプログラムを本製品用に修正）を提供していることを表わします。

図４－１　システム構築の概要

４．２　共通ヘッダファイルの作成

４．２．１　共通ヘッダファイルの定義内容

共通ヘッダファイル（hdcommon.h）は、デバッガの環境とHI-SH7の環境を定義するファイルです。

このファイルには、すでに、例題システムのデバッガ環境とHI-SH7の環境が定義されています。ユーザは、このファイルを修正して（再定義して）、ユーザシステムに応じたデバッガ環境とHI-SH7の環境を定義してください。共通ヘッダファイル（hdcommon.h）は、インストール時の作業ディレクトリ（shdbg）に格納されています。

本ファイルは以下のプログラムからそれぞれ共通に読み込まれ、定義した内容が反映されます。

・デバッガセットアップテーブル（hdsuptbl.c）
・HI-SH7セットアップテーブル（hisuptbl.c）
・ベクタテーブル（hivcttbl.c）
・MCU初期化ルーチン（himcuini.c）
・システム初期化ハンドラ（hisysini.c）
・未定義割込み異常処理ルーチン（hiintdwn.c）
・システム異常終了処理ルーチン（hisysdwn.c）
・タスク用スタック領域定義ヘッダ（hitskstk.h）
・割込みハンドラ用スタック領域定義ヘッダ（hiintstk.h）

表４－１に共通ヘッダファイルの定義内容と反映プログラムを示します。

表４－１　共通ヘッダファイルの定義内容と反映プログラム

項番	定　義　内　容	反映プログラム
1	デバッガの使用／未使用	全対象プログラム
2	デバッガが使用するタスクID、メイルボックスID ・オンラインデバッガタスクのタスクID ・コンソールタスクのタスクID ・オンラインデバッガタスクが使用するメイルボックスID ・コンソールタスクが使用するメイルボックスID	・デバッガセットアップテーブル ・HI-SH7セットアップテーブル ・システム初期化ハンドラ
3	デバッガが使用するタスクのスタックサイズ ・オンラインデバッガタスクのスタックサイズ ・コンソールタスクのスタックサイズ	・デバッガセットアップテーブル ・HI-SH7セットアップテーブル
4	HI-SH7環境の補正値 ・割込みネスト数（≦カーネル割込みマスクレベル）の加算値 ・最大タスクIDの加算値 ・初期登録タスク数の加算値 ・最大メイルボックスIDの加算値	・HI-SH7セットアップテーブル
5	ユーザスタックサイズの加算値 ・タスク用スタックサイズ加算値 ・割込みハンドラ用スタックサイズ加算値	・タスク用スタック領域定義ヘッダ ・割込みハンドラ用スタック領域定義ヘッダ

４．２．２　デバッガの使用／未使用の定義

デバッガの使用／未使用を定義します。

この定義によって、HI-SH7システムにデバッガを組込むか、組込まないかを選択することができます。したがって、デバッガを組込むために修正したHI-SH7セットアップテーブルやHI-SH7システムのユーザプログラムを、再修正（元に戻す）することなく、デバッガを組込まないHI-SH7システムを容易に再構築することができます。

デバッガの使用／未使用は、hd_dbgに"USE"（使用）、または"NOTUSE"（未使用）を定義します。

図４－２にデバッガの使用／未使用の定義例を示します。

前省略

```
/*****************************************************************************/
/*                                                                           */
/*          Definition debugger use in user system                           */
/*                                                                           */
/*          Usage    :   #define hd_dbg { USE | NOTUSE }                    */
/*                       USE      :   use the debugger                        */
/*                       NOTUSE   :   not use the debugger                    */
/*                                                                           */
/*****************************************************************************/
#define hd_dbg           USE
```

以降省略

図４－２　デバッガの使用／未使用の定義例（hdcommon.h）

４．２．３　デバッガが使用するタスクID、メイルボックスIDの定義

次のタスクIDとメイルボックスIDを定義します。

(1)　オンラインデバッガタスクのタスクID
　オンラインデバッガタスクのタスクIDをhd_dbgtskidに定義します。
　通常は、HI-SH7セットアップテーブル（hdsuptbl.c）で定義した最大タスクID（hi_maxtskid）に1を加算した値を定義してください。

(2)　コンソールタスクのタスクID
　コンソールタスクのタスクIDをhd_cnstskidに定義します。
　通常は、HI-SH7セットアップテーブル（hdsuptbl.c）で定義した最大タスクID（hi_maxtskid）に2を加算した値を定義してください。

(3)　オンラインデバッガタスクが使用するメイルボックスID
　オンラインデバッガタスクが使用するメイルボックスIDをhd_dbgmbxidに定義します。
　通常は、HI-SH7セットアップテーブル（hdsuptbl.c）で定義した最大メイルボックスID（hi_maxmbxid）に1を加算した値を定義してください。

(4)　コンソールタスクが使用するメイルボックスID
　コンソールタスクが使用するメイルボックスIDをhd_cnsmbxidに定義します。
　通常は、HI-SH7セットアップテーブル（hdsuptbl.c）で定義した最大メイルボックスID（hi_maxmbxid）に2を加算した値を定義してください。

(1), (2)で定義した値は、デバッガセットアップテーブル、HI-SH7セットアップテーブル（hdsuptbl.c）の初期登録タスクテーブル（INITSK）、およびシステム初期化ハンドラに反映されます。(3), (4)で定義した値は、デバッガセットアップテーブルに反映されます。

図４－３にデバッガが使用するタスクID、メイルボックスIDの定義例を示します。

前省略

```
/************************************************************************/
/*                                                                      */
/*          Definition ID of debugger use                               */
/*                                                                      */
/************************************************************************/
/*------------------------------- task id --------------------------------*/
#define hd_dbgtskid     6               /* task id of debugger task         */
#define hd_cnstskid     7               /* task id of console task          */
                                        /*      range : [1...hi_maxtskid]   */

/*------------------------------ mail box id -----------------------------*/
#define hd_dbgmbxid     4               /* mail box id of debugger task use */
#define hd_cnsmbxid     5               /* mail box id of console task use  */
                                        /*      range : [1...hi_maxmbxid]   */
```

以降省略

図４－３　デバッガが使用するタスクID、メイルボックスIDの定義例（hdcommon.h**）**

４．２．４　デバッガが使用するタスクのスタックサイズの定義

次のタスクのスタックサイズを定義します。

(1)　オンラインデバッガタスクのスタックサイズ
　　オンラインデバッガタスクのスタックサイズをDBGTSKSTKSZに定義します。
　　オンラインデバッガタスクのスタックサイズは、必ず1496以上を定義してください。すでに定義してある値は、絶対に変更しないでください。

(2)　コンソールタスクのスタックサイズ
　　コンソールタスクのスタックサイズをCNSTSKSTKSZに定義します。
　　コンソールタスクのスタックサイズは、必ず780以上を定義してください。すでに定義してある値は、絶対に変更しないでください。

定義した値は、デバッガセットアップテーブル、HI-SH7セットアップテーブル（hdsuptbl.c）の初期タスクスタックポインタテーブル（INITSP）に反映されます。
なお、オンラインデバッガのタスクスタック領域は、デバッガセットアップテーブルの内部で独自に確保します。したがって、ユーザは、タスク用スタック領域定義プログラムで確保する必要はありません。

図４－４にデバッガが使用するタスクのスタックサイズの定義内容を示します。

前省略

```
/************************************************************************/
/*                                                                      */
/*          Definition stack size of debugger, console task             */
/*                                                                      */
/************************************************************************/
#define DBGTSKSTKSZ     1496            /* stack size of debugger task      */
#define CNSTSKSTKSZ      780            /* stack size of console task       */
```

以降省略

図４－４　デバッガが使用するタスクのスタックサイズの定義内容（hdcommon.h）

４．２．５　HI-SH7環境の補正値の定義

次のHI-SH7環境の補正値を定義します。

(1)　割込みネスト数（≦カーネル割込みレベル）の加算値

デバッガは、コンソール入出力を行なうために、シリアル送受信割込みを使用します。このため、割込みネスト数（≦カーネル割込みレベル）の加算値をADDLOWINTNSTに定義します。

通常は、1を定義してください。ただし、コンソールドライバのシリアル送受信の割込みレベルを、ユーザが使用する他の割込みのレベルと同一とする場合は、0を定義してください。

定義した値は、HI-SH7セットアップテーブル（hdsuptbl.c）のhi_lowintnstに加算されます。

(2)　最大タスクIDの加算値

最大タスクIDの加算値をADDMAXTSKIDに定義します。

通常は、2を定義してください。ただし、HI-SH7セットアップテーブルに定義したタスクIDの範囲内（1～hi_maxtskid）に、デバッガが使用するタスクのIDを割当てる場合は、0を定義してください。

定義した値は、HI-SH7セットアップテーブル（hdsuptbl.c）のhi_maxtskidに加算されます。

(3)　初期登録タスク数の加算値

初期登録タスク数の加算値をADDINITSKNUMに定義します。

初期登録タスク数の加算値は、必ず2を定義してください。

定義した値は、HI-SH7セットアップテーブル（hdsuptbl.c）のhi_initsknumに加算されます。

(4)　最大メイルボックスIDの加算値

最大メイルボックスIDの加算値をADDMAXMBXIDに定義します。

通常は、2を定義してください。ただし、HI-SH7セットアップテーブルに定義したメイルボックスIDの範囲内（1～hi_maxmbxid）に、デバッガが使用するメイルボックスのIDを割当てる場合は、0を定義してください。

定義した値は、HI-SH7セットアップテーブル（hdsuptbl.c）のhi_maxmbxidに加算されます。

図４－５にHI-SH7環境の補正値の定義例を示します。

```
                                前省略

/*************************************************************************/
/*                                                                       */
/*          Definition parameter for HI-SH7 setup table                  */
/*                                                                       */
/*************************************************************************/
#define ADDLOWINTNST    1               /* add value to hi_lowintnst      */
#define ADDMAXTSKID     2               /* add value to hi_maxtskid       */
#define ADDINITSKNUM    2               /* add value to hi_initsknum      */
#define ADDMAXMBXID     2               /* add value to hi_maxmbxid       */

                                以降省略
```

図４－５　HI-SH7環境の補正値の定義例（hdcommon.h）

４．２．６　ユーザスタックサイズの補正値の定義

次のユーザスタックサイズの補正値を定義します。

(1)　タスク用スタックサイズの加算値

本デバッガは、デバッガが使用する割込み／例外によって、ユーザタスクのスタック領域を消費します。このため、タスク用スタックサイズの加算値をADDTSKSTKSZに定義します。

タスク用スタックサイズの加算値は、必ず60以上を定義してください。すでに定義してある値は、絶対に変更しないでください。

定義した値は、タスク用スタック領域定義ヘッダ（hitskstk.h）のhi_tskstkszxxに加算されます。

(2)　割込みハンドラ用スタックサイズの加算値

本デバッガは、デバッガが使用する割込み／例外によって、ユーザ割込みハンドラのスタック領域を消費します。このため、割込みハンドラ用スタックサイズの加算値をADDINTSTKSZに定義します。

割込みハンドラ用スタックサイズの加算値は、必ず60以上を定義してください。すでに定義してある値は、絶対に変更しないでください。

定義した値は、割込みハンドラ用スタック領域定義ヘッダ（hiintstk.h）のhi_inistkszxxに加算されます。

図４－６にユーザスタックサイズの補正値の定義例を示します。

```
                                前省略
/************************************************************************/
/*                                                                      */
/*          Definition parameter for task, interrupt stack size         */
/*                                                                      */
/************************************************************************/
#define ADDTSKSTKSZ     60              /* add value to hi_tskstkszxx       */
#define ADDINTSTKSZ     60              /* add value to hi_intstkszxx       */

                                以降省略
```

図４－６　ユーザスタックサイズの補正値の定義例（hdcommon.h）

４．３　デバッガセットアップテーブルの作成

４．３．１　デバッガセットアップテーブルの定義内容

デバッガセットアップテーブルは、デバッガのコマンド選択や、環境を定義するテーブルです。ただし、HI-SH7システムと共通する環境は、共通ヘッダ（hdcommon.h）で定義します。

表４－２にデバッガセットアップテーブルの定義内容を示します。

表４－２　デバッガセットアップテーブルの定義内容

項番	定　義　内　容
1	デバッガの各機能の使用／未使用
2	デバッガの各コマンドの使用／未使用
3	BREAKコマンドのブレークポイント数
4	オフラインデバッガの起動／非起動
5	コンソールドライバ情報

４．３．２　デバッガセットアップテーブルの構成

デバッガセットアップテーブルは、次の3つのC言語記述したファイルから構成しています。

(a)　hdsuptbl.c

ユーザは、このファイルを修正してデバッガセットアップテーブルを作成します。

このファイルには、すでに、例題システムのデバッガ環境が定義されています。ユーザは、このファイルを修正して（再定義して）、ユーザシステムに必要なデバッガ環境を定義してください。

(b)　hdsuptbl.h

このファイルでは、ユーザがデバッガ環境を定義するために必要なシステム情報を定義しています。
このファイルは、絶対に変更しないでください。

(c)　hdsuptbl.inc

このファイルでは、ユーザが定義したデバッガ環境を解析します。
このファイルは、絶対に変更しないでください。

図４－７にデバッガセットアップテーブルの構成を示します。

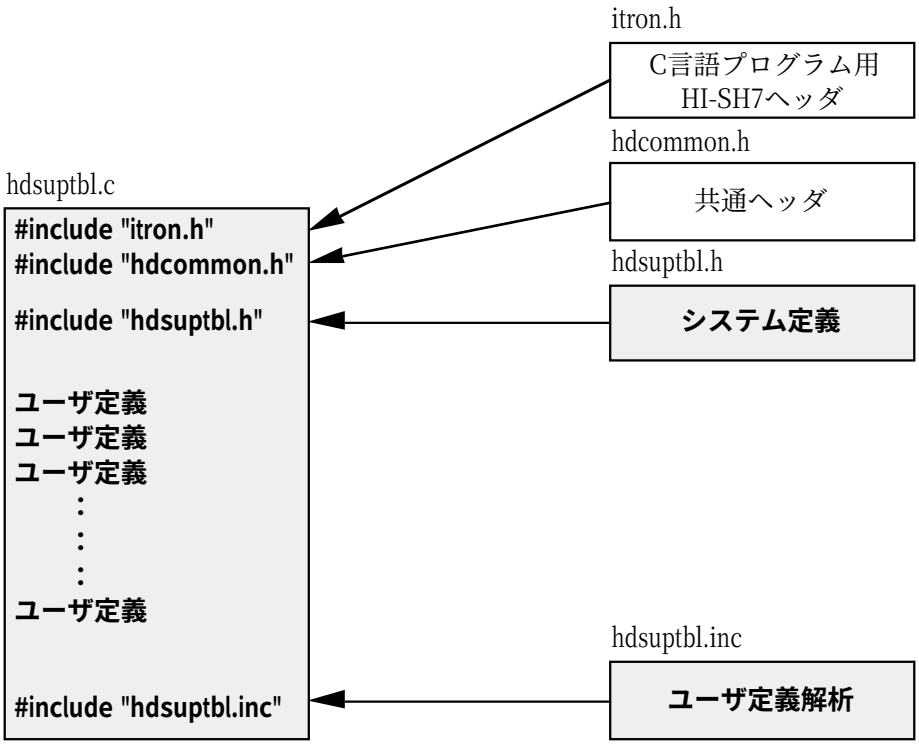


図４－７　デバッガセットアップテーブルの構成

４．３．３　デバッガの各機能の使用／未使用の定義

デバッガの各機能の使用／未使用を定義します。この定義によって、ユーザシステムで必要とするデバッガの機能を機能単位に選択することができます。

デバッガの各機能の使用／未使用の定義は、hd_xxxxxxx_func（xxxxxxxはデバッガの機能名称）に"USE"（使用）、または"NOTUSE"（未使用）を定義します。

以下にデバッガ機能とその定義変数を示します。

・メッセージ機能　　：　hd_message_func
・マルチタスク機能　　：　hd_multi_func
・メモリ操作機能　　：　hd_memory_func
・実行制御機能　　：　hd_execute_func
・ホスト機能　　：　hd_host_func

図４－８に各コマンドの機能単位の使用／未使用の定義例を示します。

前省略

```
/*****************************************************************************/
/*                                                                           */
/*          Definition function use in user system                           */
/*                                                                           */
/*          Usage    :   #define hd_xxxxxxxx { USE | NOTUSE }               */
/*                       USE      :   use the function (xxxxxxxx)            */
/*                       NOTUSE   :   not use the function (xxxxxxxx)        */
/*                                                                           */
/*****************************************************************************/
#define hd_message_func  USE            /* message              function    */
#define hd_multi_func    USE            /* multi-task           function    */
#define hd_memory_func   USE            /* memory               function    */
#define hd_execute_func  USE            /* execute              function    */
#define hd_host_func     NOTUSE         /* host                 function    */
```

以降省略

図４－８　各コマンドの機能単位の使用／未使用の定義例（hdsuptbl.c**）**

４．３．４　デバッガの各コマンドの使用／未使用の定義

デバッガの各コマンドの使用／未使用を定義します。この定義によって、ユーザシステムで必要とするデバッガの機能をコマンド単位に選択することができます。

デバッガの各コマンドの使用／未使用の定義は、hd_xxxxxxxxxx（xxxxxxxxxxはデバッガのコマンド名称）に"USE"（使用）、または"NOTUSE"（未使用）を定義します。

なお、「４．３．３　デバッガの各機能の使用／未使用の定義」で、未使用を定義した機能に含まれるコマンドは、本定義は無視されます（未使用となります）。

図４－９にデバッガの各コマンドの使用／未使用の定義例を示します。

前省略

```
/************************************************************************/
/*                                                                      */
/*          Definition command use in user system                       */
/*                                                                      */
/*          Usage   :   #define hd_xxxxxxxx { USE | NOTUSE }            */
/*                      USE     :   use the command (xxxxxxxx)          */
/*                      NOTUSE  :   not use the command (xxxxxxxx)      */
/*                                                                      */
/************************************************************************/
```

メッセージ機能

```
/*--------------------------- message function ---------------------------*/
#if     hd_message_func                 /* case of hd_message_func = USE    */
#define hd_ihelp        USE             /* IHELP                command     */
#define hd_iid          USE             /* IID                  command     */
#endif
```

マルチタスク機能

```
/*-------------------------- multi-task function -------------------------*/
#if     hd_multi_func                   /* case of hd_multi_func = USE      */
#define hd_can_wup      USE             /* CAN_WUP              command     */
#define hd_chg_pri      USE             /* CHG_PRI              command     */
#define hd_clr_flg      USE             /* CLR_FLG              command     */
#define hd_cre_tsk      USE             /* CRE_TSK              command     */
#define hd_del_tsk      USE             /* DEL_TSK              command     */
#define hd_flg_sts      USE             /* FLG_STS              command     */
#define hd_get_blk      USE             /* GET_BLK              command     */
#define hd_get_tim      NOTUSE          /* GET_TIM              command     */
#define hd_istatus      USE             /* ISTATUS              command     */
#define hd_itrace       USE             /* ITRACE               command     */
#define hd_itrace_act   USE             /* ITRACE_ACTIVATE      command     */
#define hd_itrace_buf   USE             /* ITRACE_BUFFER        command     */
#define hd_itrace_sea   USE             /* ITRACE_SEARCH        command     */
#define hd_mbx_sts      USE             /* MBX_STS              command     */
```

図４－９　デバッガの各コマンドの使用／未使用の定義例（hdsuptbl.c）（１／３）

```
#define hd_mpldef       USE             /* MPLDEF               command     */
#define hd_mpl_sts      USE             /* MPL_STS              command     */
#define hd_pol_flg      USE             /* POL_FLG              command     */
#define hd_rcv_msg      USE             /* RCV_MSG              command     */
#define hd_rel_blk      USE             /* REL_BLK              command     */
#define hd_rel_wai      USE             /* REL_WAI              command     */
#define hd_req_sem      USE             /* REQ_SEM              command     */
#define hd_rot_rdq      USE             /* ROT_RDQ              command     */
#define hd_rsm_tsk      USE             /* RSM_TSK              command     */
#define hd_run_tsk      USE             /* RUN_TSK              command     */
#define hd_sem_sts      USE             /* SEM_STS              command     */
#define hd_set_flg      USE             /* SET_FLG              command     */
#define hd_set_tim      USE             /* SET_TIM              command     */
#define hd_sig_sem      USE             /* SIG_SEM              command     */
#define hd_snd_msg      USE             /* SND_MSG              command     */
#define hd_sta_tsk      USE             /* STA_TSK              command     */
#define hd_stp_tsk      USE             /* STP_TSK              command     */
#define hd_sus_tsk      USE             /* SUS_TSK              command     */
#define hd_svcdef       USE             /* SVCDEF               command     */
#define hd_sysdef       USE             /* SYSDEF               command     */
#define hd_ter_tsk      USE             /* TER_TSK              command     */
#define hd_tskdef       USE             /* TSKDEF               command     */
#define hd_tsk_sts      USE             /* TSK_STS              command     */
#define hd_wup_tsk      USE             /* WUP_TSK              command     */
#endif
```

メモリ操作機能

```
/*---------------------------- memory function ----------------------------*/
#if     hd_memory_func                  /* case of hd_memory_func = USE     */
#define hd_assemble     USE             /* ASSEMBLE             command     */
#define hd_disassemble  USE             /* DISASSEMBLE          command     */
#define hd_dump         USE             /* DUMP                 command     */
#define hd_fill         NOTUSE          /* FILL                 command     */
#define hd_memory       USE             /* MEMORY               command     */
#define hd_move         USE             /* MOVE                 command     */
#endif
```

図４－９　デバッガの各コマンドの使用／未使用の定義例（hdsuptbl.c**）（２／３）**

実行制御機能

```
/*--------------------------- execute function ---------------------------*/
#if     hd_execute_func                 /* case of hd_execute_func = USE    */
#define hd_regupd       USE             /* . (register update)  command     */
#define hd_break        USE             /* BREAK                command     */
#define hd_break_ubc    USE             /* BREAK_UBC            command     */
#define hd_go           USE             /* GO                   command     */
#define hd_register     USE             /* REGISTER             command     */
#define hd_reset        USE             /* RESET                command     */
#define hd_step         USE             /* STEP                 command     */
#define hd_step_over    USE             /* STEP_OVER            command     */
#define hd_step_ubc     USE             /* STEP_UBC             command     */
#endif
```

ホスト機能

```
/*---------------------------- host function ----------------------------*/
#if     hd_host_func                    /* case of hd_host_func = USE       */
#define hd_load         USE             /* LOAD                 command     */
#define hd_save         USE             /* SAVE                 command     */
#define hd_verify       NOTUSE          /* VERIFY               command     */
#endif
```

以降省略

図４－９　デバッガの各コマンドの使用／未使用の定義例（hdsuptbl.c**）（３／３）**

４．３．５　BREAKコマンドのブレークポイント数の定義

BREAKコマンドのブレークポイント数をhd_breaknumに定義します。この定義によって、BREAKコマンドで登録できるブレークポイント数が決定します。

BREAKコマンドのブレークポイント数は、1～32を定義してください。定義する値が小さいほど、デバッガの作業領域は小さくなります。

なお、実行制御機能、またはBREAKコマンドを未使用に定義した場合は、本定義は無視されます。

図４－１０にBREAKコマンドのブレークポイント数の定義例を示します。

```
                                前省略
/************************************************************************/
/*                                                                      */
/*          Definition command information                              */
/*                                                                      */
/************************************************************************/
#if     hd_execute_func                 /* case of hd_execute_func = USE    */
#if     hd_break                        /* case of hd_break = USE           */
#define hd_breaknum     32              /* number of break point            */
                                        /*                 for BREAK command */
                                        /*      range : [1...32]            */
#endif
#endif
                                以降省略
```

図４－１０　BREAKコマンドのブレークポイント数の定義例（hdsuptbl.c）

４．３．６　オフラインデバッガの起動／非起動の定義

オフラインデバッガの起動／非起動をhd_stadbgflgに定義します。

オフラインデバッガの起動／非起動の定義は、hd_stadbgflgに"TRUE"（起動）、または"FALSE"（非起動）を定義します。

"TRUE"（起動）を定義すると、CPUのリセット直後にオフラインデバッガが起動します。（MCU初期化ルーチン、デバッガリセットルーチンを実行後、オフラインデバッガが起動します）

図４－１１にオフラインデバッガの起動／非起動の定義例を示します。

```
                                前省略
/************************************************************************/
/*                                                                      */
/*          Definition offline debugger start mode at CPU reset         */
/*                                                                      */
/*          Usage   :   #define hd_stadbgflg { TRUE | FALSE }           */
/*                      TRUE    :   start offline debugger at CPU reset */
/*                      FALSE   :   not start offline debugger at CPU reset */
/*                                                                      */
/************************************************************************/
#define hd_stadbgflg    TRUE
                                以降省略
```

図４－１１　オフラインデバッガの起動／非起動の定義例（hdsuptbl.c）

４．３．７　コンソールドライバ情報の定義

次のコンソールドライバ情報を定義します。定義した値によって、コンソール入出力が行われます。

(1)　CPUクロック

ユーザシステム上のCPUのクロック数をhd_cnsclkに定義します。これは、SCIを初期化するために必要な情報です。

(2)　SCIチャンネル

使用するSCIのチャンネルをhd_cnssciに定義します。

SCIチャンネルは、0または1（SCI0またはSCI1）を定義してください。

(3)　シリアル転送速度

シリアル転送速度をhd_cnsbpsnoに定義します。

シリアル転送速度は、0～10（110bps～38400bps）を定義してください。

(4)　シリアル送受信の割込みレベル

シリアル送受信の割込みレベルをhd_cnsintlvlに定義します。

シリアル送受信の割込みレベルは、1からHI-SH7セットアップテーブルで定義したカーネル割込みマスクレベル（hi_knlmsklvl）までの値を定義してください。

図４－１２にコンソールドライバの定義例を示します。

前省略

```
/************************************************************************/
/*                                                                      */
/*          Definition environment of console                           */
/*                                                                      */
/************************************************************************/
#define hd_cnsclk       20000000        /* CPU clock = 20MHz            */
#define hd_cnssci       1               /* SCI channel                  */
                                        /*          0 : SCI0            */
                                        /*          1 : SCI1            */
#define hd_cnsbpsno     7               /* bit rate number              */
                                        /*          0 :   110 bps       */
                                        /*          1 :   150 bps       */
                                        /*          2 :   300 bps       */
                                        /*          3 :   600 bps       */
                                        /*          4 :  1200 bps       */
                                        /*          5 :  2400 bps       */
                                        /*          6 :  4800 bps       */
                                        /*          7 :  9600 bps       */
                                        /*          8 : 19200 bps       */
                                        /*          9 : 31250 bps       */
                                        /*         10 : 38400 bps       */
#define hd_cnsintlvl    14              /* interrupt level = 14         */
                                        /*      range : [1...hi_knlmsklvl] */
```

以降省略

図４－１２　コンソールドライバ情報の定義例（hdsuptbl.c）

４．４　HI-SH7セットアップテーブルの修正

４．４．１　HI-SH7セットアップテーブルの修正項目

表４－３にHI-SH7セットアップテーブルの修正項目を示します。

表４－３　HI-SH7セットアップテーブルの修正項目

項番	修　正　項　目
1	インクルードファイル構成
2	システムコールの使用／未使用
3	割込みネスト数（≦カーネル割込みマスクレベル）（hi_lowintnst）
4	最大タスクID（hi_maxtskid）
5	初期登録タスク数（hi_initsknum）
6	初期タスクスタックポインタテーブル（INITSP）
7	初期登録タスクテーブル（INITSK）
8	最大メイルボックスID（hi_maxmbxid）

４．４．２　インクルードファイル構成の修正

HI-SH7セットアップテーブルのインクルードファイル構成を修正します。
図４－１３にHI-SH7セットアップテーブルの構成を示します。

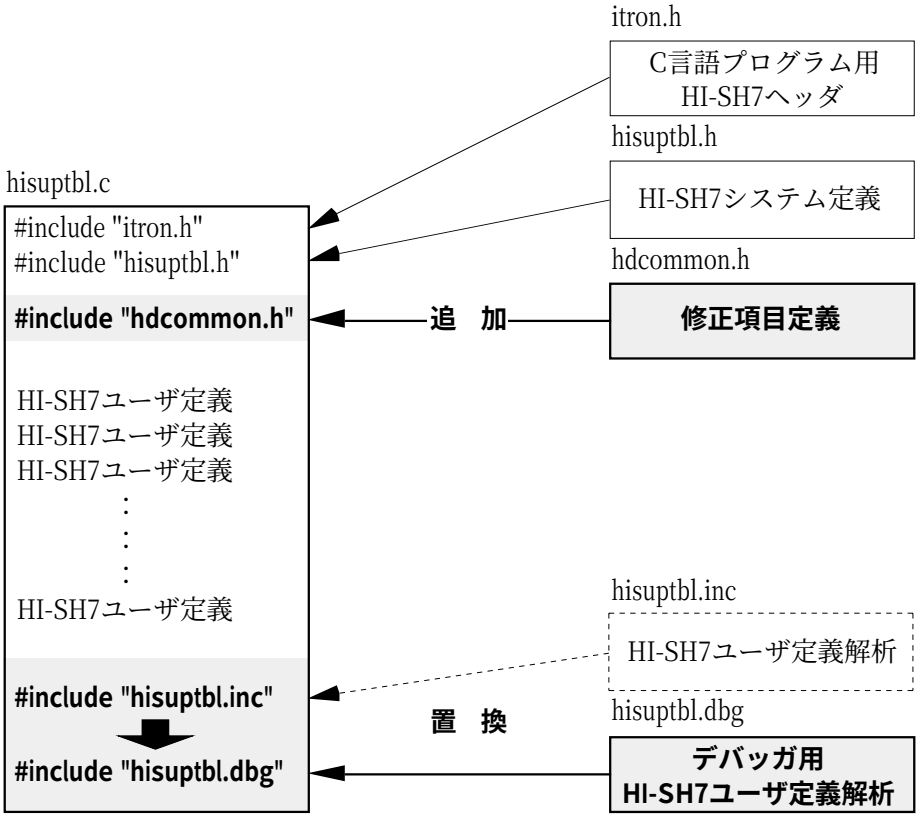


図４－１３　HI-SH7セットアップテーブルの構成

４．４．３　HI-SH7セットアップテーブルの修正

HI-SH7セットアップテーブルを修正します。

図４－１４にHI-SH7セットアップテーブルの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*            HI-SH7 setup table ( Ver. 1.0 )                           */
/*                                                                      */
/*                                                                      */
/*            Copyright (c)  Hitachi, Ltd. 1993.                        */
/*            Licensed Material of Hitachi, Ltd.                        */
/*                                                                      */
/*            HI-SH7(HS0700ITCN1SM) V1.0                                */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hisuptbl.c ;                                            */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 setup table ;                                    */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hisuptbl.h"
#include    "hdcommon.h"                                                ･･･(a)

                                中省略

/************************************************************************/
/*                                                                      */
/*          Definition SVC use in user system                           */
/*                                                                      */
/*          Usage   :   #define hi_xyyy_zzz { USE | NOTUSE }            */
/*                      USE     :   use the SVC (xyyy_zzz)              */
/*                      NOTUSE  :   not use the SVC (xyyy_zzz)          */
/*                                                                      */
/************************************************************************/
/*---------------------- task management function ----------------------*/
#define hi_cre_tsk      USE             /* cre_tsk  SVC                 */
#define hi_sta_tsk      USE             /* sta_tsk  SVC                 */･･･(b)
#define hi_ista_tsk     USE             /* ista_tsk SVC                 */･･･(b)
#define hi_del_tsk      USE             /* del_tsk  SVC                 */
#define hi_ext_tsk      USE             /* ext_tsk  SVC                 */･･･(b)
#define hi_exd_tsk      USE             /* exd_tsk  SVC                 */
#define hi_ter_tsk      USE             /* ter_tsk  SVC                 */･･･(b)
#define hi_chg_pri      USE             /* chg_pri  SVC                 */
#define hi_ichg_pri     USE             /* ichg_pri SVC                 */
```

図４－１４　HI-SH7セットアップテーブル（hisuptbl.c）の修正内容（１／３）

```
#define hi_rot_rdq      USE             /* rot_rdq  SVC                         */
#define hi_irot_rdq     USE             /* irot_rdq SVC                         */
#define hi_rel_wai      USE             /* rel_wai  SVC                         */
#define hi_irel_wai     USE             /* irel_wai SVC                         */
#define hi_get_tid      USE             /* get_tid  SVC                         */
#define hi_tsk_sts      USE             /* tsk_sts  SVC                        */・・・(b)

/*--------------- task synchronization management function -----------------*/
#define hi_sus_tsk      USE             /* sus_tsk  SVC                         */
#define hi_isus_tsk     USE             /* isus_tsk SVC                         */
#define hi_rsm_tsk      USE             /* rsm_tsk  SVC                         */
#define hi_irsm_tsk     USE             /* irsm_tsk SVC                         */
#define hi_slp_tsk      USE             /* slp_tsk  SVC                        */・・・(b)
#define hi_wai_tsk      USE             /* wai_tsk  SVC                         */
#define hi_wup_tsk      USE             /* wup_tsk  SVC                         */
#define hi_iwup_tsk     USE             /* iwup_tsk SVC                        */・・・(b)
#define hi_can_wup      USE             /* can_wup  SVC                         */

/*-------------- synchronization and communication function ---------------*/
#define hi_set_flg      USE             /* set_flg  SVC                         */
#define hi_iset_flg     USE             /* iset_flg SVC                         */
#define hi_clr_flg      USE             /* clr_flg  SVC                         */
#define hi_wai_flg      USE             /* wai_flg  SVC                         */
#define hi_pol_flg      USE             /* pol_flg  SVC                         */
#define hi_flg_sts      USE             /* flg_sts  SVC                         */
#define hi_sig_sem      USE             /* sig_sem  SVC                         */
#define hi_isig_sem     USE             /* isig_sem SVC                         */
#define hi_wai_sem      USE             /* wai_sem  SVC                         */
#define hi_preq_sem     USE             /* preq_sem SVC                         */
#define hi_sem_sts      USE             /* sem_sts  SVC                         */
#define hi_snd_msg      USE             /* snd_msg  SVC                        */・・・(b)
#define hi_isnd_msg     USE             /* isnd_msg SVC                         */
#define hi_rcv_msg      USE             /* rcv_msg  SVC                        */・・・(b)
#define hi_prcv_msg     USE             /* prcv_msg SVC                         */
#define hi_mbx_sts      USE             /* mbx_sts  SVC                         */


                                        中省略


/************************************************************************/
/*                                                                      */
/*          Definition kernel information                               */
/*                                                                      */
/************************************************************************/
#define hi_knlmsklvl    14              /* kernel mask level of interrupt   */
                                        /*      range : [1...15]            */
#define hi_uppintnst    0               /* nest number of upper interrupt   */
                                        /*            than kernel mask level*/
                                        /*      range : [0...]              */
#define hi_lowintnst    (2 + ADDLOWINTNST)                                  ・・・(b)
                                        /* nest number of lower interrupt   */
                                        /*            than kernel mask level*/
                                        /*      range : [0...15]            */
```

図４－１４　HI-SH7セットアップテーブル（hisuptbl.c）の修正内容（１／３）

```
/*************************************************************************/
/*                                                                       */
/*          Definition task information                                  */
/*                                                                       */
/*************************************************************************/
#define hi_maxtskid     (5 + ADDMAXTSKID)                                   ･･･(c)
                                        /* max task id                  */
                                        /*      range : [0...1023]      */
#define hi_maxtskpri    5               /* max task priority            */
                                        /*      range : [1...255]       */
#define hi_initsknum    (1 + ADDINITSKNUM)                                  ･･･(d)
                                        /* number of initial task       */
                                        /*      range : [0...1023]      */

/*************************************************************************/
/*                                                                       */
/*          Definition task information to setup table (INITSP, INITSK)  */
/*                                                                       */
/*************************************************************************/
/*-------------------- define task stack end address ---------------------*/
#if     hi_maxtskid                     /* case of hi_maxtskid > 0      */
#include        "hitskstk.h"            /* include "hitskstk.h"         */
extern  VW      hi_tskstk1[];           /* task stack of tskid = 1      */
extern  VW      hi_tskstk2[];           /* task stack of tskid = 2      */
extern  VW      hi_tskstk3[];           /* task stack of tskid = 3      */
extern  VW      hi_tskstk4[];           /* task stack of tskid = 4, 5   */
#if     hd_dbg                          /* case of hd_dbg = USE         */  ･･･(e)
extern  VW      _ODdbgtskstk[];         /* task stack of debugger task  */
extern  VW      _ODcnstskstk[];         /* task stack of console task   */
#endif

const   INITSP  _OHinitsp[hi_maxtskid] = {
        /* define task stack end address    */
            (VW *)&hi_tskstk1[(hi_tskstksz1) / sizeof(VW)], /* tskid = 1    */
            (VW *)&hi_tskstk2[(hi_tskstksz2) / sizeof(VW)], /* tskid = 2    */
            (VW *)&hi_tskstk3[(hi_tskstksz3) / sizeof(VW)], /* tskid = 3    */
            (VW *)&hi_tskstk4[(hi_tskstksz4) / sizeof(VW)], /* tskid = 4    */
            (VW *)&hi_tskstk4[(hi_tskstksz4) / sizeof(VW)], /* tskid = 5    */
#if     hd_dbg                          /* case of hd_dbg = USE         */  ･･･(f)
            (VW *)&_ODdbgtskstk[(DBGTSKSTKSZ) / sizeof(VW)],/* debugger task*/
            (VW *)&_ODcnstskstk[(CNSTSKSTKSZ) / sizeof(VW)],/* console task */
#endif
        };
#else                                   /* case of hi_maxtskid = 0      */
#define _OHinitsp       NADR            /* not define task stack        */
#endif
```

図４－１４　HI-SH7セットアップテーブル（hisuptbl.c）の修正内容（２／３）

```
/*------------------- define initial task information ---------------------*/
#if     hi_initsknum                    /* case of hi_initsknum > 0         */
extern  TASK    hi_cnsdrv();            /* console driver task              */
#if     hd_dbg                          /* case of hd_dbg = USE             */  ･･･(g)
extern  TASK    _0Ddbgtsk();            /* debugger task                    */
extern  TASK    _0Dcnstsk();            /* console task                     */
#endif

const   INITSK  _0Hinitsk[hi_initsknum] = {
        /* define console driver task   */
            (ID)1,                      /* task id              = 1         */
            (TPRI)2,                    /* initial task priority = 2        */
            (TASKP)hi_cnsdrv,           /* task start address   = hi_cnsdrv*/
#if     hd_dbg                          /* case of hd_dbg = USE             */  ･･･(h)
            (ID)hd_dbgtskid,            /* task id              = hd_dbgtskid*/
            (TPRI)1,                    /* initial task priority = 1        */
            (TASKP)_0Ddbgtsk,           /* task start address   = _0Ddbgtsk*/
            (ID)hd_cnstskid,            /* task id              = hd_cnstskid*/
            (TPRI)1,                    /* initial task priority = 1        */
            (TASKP)_0Dcnstsk,           /* task start address   = _0Dcnstsk*/
#endif
        };
#else                                   /* case of hi_initsknum = 0         */
#define _0Hinitsk       NADR            /* not define initial task          */
#endif

/*****************************************************************************/
/*                                                                           */
/*          Definition synchronization and communication object information */
/*                                                                           */
/*****************************************************************************/
#define hi_maxflgid     3               /* max event flag id                */
                                        /*      range : [0...1023]          */
#define hi_maxsemid     3               /* max semaphore id                 */
                                        /*      range : [0...1023]          */
#define hi_maxmbxid     (3 + ADDMAXMBXID)                                    ･･･(i)
                                        /* max mail box id                  */
                                        /*      range : [0...1023]          */


                                        中省略


#if     hd_dbg                          /* case of hd_dbg = USE             */  ･･･(j)
#include    "hisuptbl.dbg"
#else                                   /* case of hd_dbg = NOTUSE          */
#include    "hisuptbl.inc"
#endif
```

図４－１４　HI-SH7セットアップテーブル（hisuptbl.c）の修正内容（３／３）

(a)　共通ヘッダ（hdcommon.h）のインクルード文を追加します。以下の文を追加してください。

追加文

```
#include    "hdcommon.h"
```

なお、以下の定義変数は、共通ヘッダで定義します。

・hd_dbg（デバッガの使用／未使用の定義変数）
・ADDLOWINTNST（割込みネスト数（≦カーネル割込みマスクレベル）の加算値）
・ADDMAXTSKID（最大タスクIDの加算値）
・ADDINITSKNUM（初期登録タスク数の加算値）
・ADDMAXMBXID（最大メイルボックスIDの加算値）
・DBGTSKSTKSZ（オンラインデバッガタスクのスタックサイズ）
・CNSTSKSTKSZ（コンソールタスクのスタックサイズ）

(b)　システムコールの使用／未使用を定義します。

以下のHI-SH7デバッガが使用しているシステムコールを"USE"（使用）に定義してください。

なお、ret_intシステムコールは、ベクタテーブルのTRAPA #61のベクタにret_intシステムコール処理のシンボル名(_0Hret_int)を定義してください。

表４－４　HI-SH7デバッガが使用しているシステムコール

項番	機能	システムコール
1	タスク管理機能	・sta_tsk,ista_tsk,ext_tsk,ter_tsk,tsk_sts
2	タスク同期機能	・slp_tsk,iwup_tsk
3	同期通信機能	・snd_msg,rcv_msg
4	割込み管理機能	・ret_int

また、デバッガのマルチタスク機能のコマンドは、HI-SH7のシステムコールを使用しています。

デバッガの各コマンドの定義で"USE"（使用）にしたコマンドと同じ名称のシステムコールを"USE"（使用）に定義してください。

デバッガの各コマンドの定義については、「４．３．４　デバッガの各コマンドの使用／未使用の定義」を参照してください。

(c)　hi_lowintnst（割込みネスト数（≦カーネル割込みマスクレベル））にADDLOWINTNSTを加算します。以下のように修正してください。

修正前

```
#define hi_lowintnst    2
```

修正後

```
#define hi_lowintnst    (2 + ADDLOWINTNST)
```

(d)　hi_maxtskid（最大タスクID）にADDMAXTSKIDを加算します。以下のように修正してください。

修正前

```
#define hi_maxtskid     5
```

修正後

```
#define hi_maxtskid     (5 + ADDMAXTSKID)
```

(e) hi_initsknum（初期登録タスク数）にADDINITSKNUMを加算します。以下のように修正してください。

修正前

```
#define hi_initsknum    1
```

修正後

```
#define hi_initsknum    (1 + ADDINITSKNUM)
```

(f) デバッガが使用するタスクのスタックアドレスをシンボル参照します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  VW      _0Ddbgtskstk[];         /* task stack of debugger task      */
extern  VW      _0Dcnstskstk[];         /* task stack of console task       */
#endif
```

(g) INITSPテーブル（初期タスクスタックポインタテーブル）にデバッガが使用するタスクのスタックアドレスを追加します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
            (VW *)&_0Ddbgtskstk[(DBGTSKSTKSZ) / sizeof(VW)],/* debugger task*/
            (VW *)&_0Dcnstskstk[(CNSTSKSTKSZ) / sizeof(VW)],/* console task */
#endif
```

(h) デバッガが使用するタスクの開始アドレスをシンボル参照します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  TASK        _0Ddbgtsk();        /* debugger task                    */
extern  TASK        _0Dcnstsk();        /* console task                     */
#endif
```

(i) INITSKテーブル（初期登録タスクテーブル）にデバッガが使用するタスクの開始アドレスを追加します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
            (ID)hd_dbgtskid,            /* task id              = hd_dbgtskid*/
            (TPRI)1,                    /* initial task priority = 1        */
            (TASKP)_0Ddbgtsk,           /* task start address    = _0Ddbgtsk*/
            (ID)hd_cnstskid,            /* task id              = hd_cnstskid*/
            (TPRI)1,                    /* initial task priority = 1        */
            (TASKP)_0Dcnstsk,           /* task start address    = _0Dcnstsk*/
#endif
```

(j) hi_maxmbxid（最大メイルボックスID）にADDMAXMBXIDを加算します。以下のように修正してください。

修正前

```
#define hi_maxmbxid     3
```

修正後

```
#define hi_maxmbxid     (3 + ADDMAXMBXID)
```

(k)　デバッガ用HI-SH7ユーザ定義インクルード（hisuptbl.dbg）を読み込むようにします。以下のように修正してください。

修正前

```
#include    "hisuptbl.inc"
```

修正後

```
#if         hd_dbg                      /* case of hd_dbg = USE          */
#include    "hisuptbl.dbg"
#else                                   /* case of hd_dbg = NOTUSE       */
#include    "hisuptbl.inc"
#endif
```

４．５　HI-SH7システムのユーザプログラムの修正

４．５．１　ベクタテーブルの修正

ベクタテーブルを修正します。
図４－１５にベクタテーブルの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*            HI-SH7 vector table ( Ver. 1.0 )                          */
/*                                                                      */
/*                                                                      */
/*            Copyright (c)  Hitachi, Ltd. 1993.                        */
/*            Licensed Material of Hitachi, Ltd.                        */
/*                                                                      */
/*            HI-SH7(HS0700ITCN1SM) V1.0                                */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hivcttbl.c ;                                            */
/*  NAME      = hi_vcttbl ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 vector table ;                                   */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hdcommon.h"                                                ･･･(a)

#if     hd_dbg                          /* case of hd_dbg = USE         */･･･(b)
extern  _0Dbrk_nmi();                   /* NMI handler                  */
extern  _0Dbrk_hbk();                   /* hardware break handler(UBC)  */
extern  _0Dbrk_sbk();                   /* software break handler(TRAPA #57)*/
extern  _0Derr_ill();                   /* illegal instruction handler  */
extern  _0Derr_slt();                   /* slot illegal instruction handler */
extern  _0Derr_cpu();                   /* CPU bus error handler        */
extern  _0Derr_dma();                   /* DMA bus error handler        */
extern  _0Hend_svc();                   /* debug SVC handler            */
extern  _0Dcnshdrer();                  /* receive error handler        */
extern  _0Dcnshdrrx();                  /* receive handler              */
extern  _0Dcnshdrtx();                  /* trans handler                */
#endif

/*-------------- handler of MCU initialize routine for HI-SH7 --------------*/
extern  hi_mcuini();                    /* MCU initialize routine       */
```

図４－１５　ベクタテーブル（hivcttbl.c）の修正内容（１／３）

中省略

```
/************************************************************************/
/*                                                                      */
/*          Definition vector table                                     */
/*                                                                      */
/************************************************************************/
const   VP  hi_vcttbl[] = {
            (VP)hi_mcuini,  /* power on reset                  (vctno = 00) */
            (VP)0x10000000, /* SP value at power on reset                   */
            (VP)hi_mcuini,  /* manual reset                    (vctno = 02) */
            (VP)0x10000000, /* SP value at manual reset                     */
#if     hd_dbg                          /* case of hd_dbg = USE             */   ・・・(c)
            (VP)_0Derr_ill, /* illegal instruction            (vctno = 04) */
            (VP)hi_intdwn05,/* reserve                        (vctno = 05) */
            (VP)_0Derr_slt, /* slot illegal instruction       (vctno = 06) */
            (VP)hi_intdwn07,/* reserve                        (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                        (vctno = 08) */
            (VP)_0Derr_cpu, /* CPU bus error                  (vctno = 09) */
            (VP)_0Derr_dma, /* DMA bus error                  (vctno = 0A) */
            (VP)_0Dbrk_nmi, /* NMI                            (vctno = 0B) */
            (VP)_0Dbrk_hbk, /* user break                     (vctno = 0C) */
#else                               /* case of hd_dbg = NOTUSE              */
            (VP)hi_intdwn04,/* illegal instruction            (vctno = 04) */
            (VP)hi_intdwn05,/* reserve                        (vctno = 05) */
            (VP)hi_intdwn06,/* slot illegal instruction       (vctno = 06) */
            (VP)hi_intdwn07,/* reserve                        (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                        (vctno = 08) */
            (VP)hi_intdwn09,/* CPU bus error                  (vctno = 09) */
            (VP)hi_intdwn0A,/* DMA bus error                  (vctno = 0A) */
            (VP)hi_intdwn0B,/* NMI                            (vctno = 0B) */
            (VP)hi_intdwn0C,/* user break                     (vctno = 0C) */
#endif
            (VP)hi_intdwn0D,/* reserve                        (vctno = 0D) */
            (VP)hi_intdwn0E,/* reserve                        (vctno = 0E) */
            (VP)hi_intdwn0F,/* reserve                        (vctno = 0F) */
```

図４－１５　ベクタテーブル（hivcttbl.c）の修正内容（２／３）

```
                中省略

        (VP)hi_intdwn37,/* trapa #55                          (vctno = 37) */
        (VP)hi_intdwn38,/* trapa #56                          (vctno = 38) */
#if     hd_dbg                          /* case of hd_dbg = USE             */  ･･･(d)
        (VP)_0Dbrk_sbk, /* trapa #57                          (vctno = 39) */
        (VP)hi_intdwn3A,/* trapa #58                          (vctno = 3A) */
        (VP)_0Hend_svc, /* trapa #59 (reserve HI-SH7)         (vctno = 3B) */
#else                               /* case of hd_dbg = NOTUSE          */
        (VP)hi_intdwn39,/* trapa #57                          (vctno = 39) */
        (VP)hi_intdwn3A,/* trapa #58                          (vctno = 3A) */
        (VP)hi_intdwn3B,/* trapa #59 (reserve HI-SH7)         (vctno = 3B) */
#endif
        (VP)_0Hsys_clk, /* trapa #60 (use HI-SH7)             (vctno = 3C) */
        (VP)_0Hret_int, /* trapa #61 (use HI-SH7)             (vctno = 3D) */
        (VP)_0Henn_svc, /* trapa #62 (use HI-SH7)             (vctno = 3E) */
        (VP)_0Hent_svc, /* trapa #63 (use HI-SH7)             (vctno = 3F) */
        (VP)hi_intdwn40,/* IRQ0                               (vctno = 40) */
        (VP)hi_intdwn41,/* IRQ1                               (vctno = 41) */

                中省略

        (VP)hi_cnshdrer,/* SCI0 ERI0 (use console driver)     (vctno = 64) */
        (VP)hi_cnshdrrx,/* SCI0 RXI0 (use console driver)     (vctno = 65) */
        (VP)hi_cnshdrtx,/* SCI0 TXI0 (use console driver)     (vctno = 66) */
        (VP)hi_intdwn67,/* SCI0 TEI0                          (vctno = 67) */
#if     hd_dbg                          /* case of hd_dbg = USE             */  ･･･(e)
        (VP)_0Dcnshdrer,/* SCI1 ERI1                          (vctno = 68) */
        (VP)_0Dcnshdrrx,/* SCI1 RXI1                          (vctno = 69) */
        (VP)_0Dcnshdrtx,/* SCI1 TXI1                          (vctno = 6A) */
#else                               /* case of hd_dbg = NOTUSE          */
        (VP)hi_intdwn68,/* SCI1 ERI1                          (vctno = 68) */
        (VP)hi_intdwn69,/* SCI1 RXI1                          (vctno = 69) */
        (VP)hi_intdwn6A,/* SCI1 TXI1                          (vctno = 6A) */
#endif
        (VP)hi_intdwn6B,/* SCI1 TEI1                          (vctno = 6B) */
        (VP)hi_intdwn6C,/* PRT PEI                            (vctno = 6C) */
        (VP)hi_intdwn6D,/* A/D ADI                            (vctno = 6D) */
        (VP)hi_intdwn6E,/* reserve                            (vctno = 6E) */
        (VP)hi_intdwn6F,/* reserve                            (vctno = 6F) */
        (VP)hi_intdwn70,/* WDT ITI                            (vctno = 70) */
        (VP)hi_intdwn71,/* REF CMI                            (vctno = 71) */
        (VP)hi_intdwn72,/* reserve                            (vctno = 72) */
        (VP)hi_intdwn73,/* reserve                            (vctno = 73) */
        (VP)hi_intdwn74,/* reserve                            (vctno = 74) */

                中省略

        (VP)hi_intdwnFD,/* reserve                            (vctno = FD) */
        (VP)hi_intdwnFE,/* reserve                            (vctno = FE) */
        (VP)hi_intdwnFF,/* reserve                            (vctno = FF) */
    };
```

図４－１５　ベクタテーブル（hivcttbl.c）の修正内容（３／３）

(a) 共通ヘッダ（hdcommon.h）のインクルード文を追加します。以下の文を追加してください。

追加文

```
#include    "hdcommon.h"
```

なお、hd_dbg（デバッガの使用／未使用の定義変数）定義変数は、共通ヘッダで定義します。

(b) デバッガが使用する割込み／例外ハンドラの開始アドレスをシンボル参照します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  _0Dbrk_nmi();                   /* NMI handler                      */
extern  _0Dbrk_hbk();                   /* hardware break handler(UBC)      */
extern  _0Dbrk_sbk();                   /* software break handler(TRAPA #57)*/
extern  _0Derr_ill();                   /* illegal instruction handler      */
extern  _0Derr_slt();                   /* slot illegal instruction handler */
extern  _0Derr_cpu();                   /* CPU bus error handler            */
extern  _0Derr_dma();                   /* DMA bus error handler            */
extern  _0Hend_svc();                   /* debug SVC handler                */
extern  _0Dcnshdrer();                  /* receive error handler            */
extern  _0Dcnshdrrx();                  /* receive handler                  */
extern  _0Dcnshdrtx();                  /* trans handler                    */
#endif
```

(c) NMIハンドラ、ユーザブレークハンドラの開始アドレスを定義します。以下のように修正してください。

修正前

```
            (VP)hi_intdwn04,/* illegal instruction            (vctno = 04) */
            (VP)hi_intdwn05,/* reserve                        (vctno = 05) */
            (VP)hi_intdwn06,/* slot illegal instruction       (vctno = 06) */
            (VP)hi_intdwn07,/* reserve                        (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                        (vctno = 08) */
            (VP)hi_intdwn09,/* CPU bus error                  (vctno = 09) */
            (VP)hi_intdwn0A,/* DMA bus error                  (vctno = 0A) */
            (VP)hi_intdwn0B,/* NMI                            (vctno = 0B) */
            (VP)hi_intdwn0C,/* user break                     (vctno = 0C) */
```

修正後

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
            (VP)_0Derr_ill, /* illegal instruction            (vctno = 04) */
            (VP)hi_intdwn05,/* reserve                        (vctno = 05) */
            (VP)_0Derr_slt, /* slot illegal instruction       (vctno = 06) */
            (VP)hi_intdwn07,/* reserve                        (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                        (vctno = 08) */
            (VP)_0Derr_cpu, /* CPU bus error                  (vctno = 09) */
            (VP)_0Derr_dma, /* DMA bus error                  (vctno = 0A) */
            (VP)_0Dbrk_nmi, /* NMI                            (vctno = 0B) */
            (VP)_0Dbrk_hbk, /* user break                     (vctno = 0C) */
#else                                   /* case of hd_dbg = NOTUSE          */
            (VP)hi_intdwn04,/* illegal instruction            (vctno = 04) */
            (VP)hi_intdwn05,/* reserve                        (vctno = 05) */
            (VP)hi_intdwn06,/* slot illegal instruction       (vctno = 06) */
            (VP)hi_intdwn07,/* reserve                        (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                        (vctno = 08) */
            (VP)hi_intdwn09,/* CPU bus error                  (vctno = 09) */
            (VP)hi_intdwn0A,/* DMA bus error                  (vctno = 0A) */
            (VP)hi_intdwn0B,/* NMI                            (vctno = 0B) */
            (VP)hi_intdwn0C,/* user break                     (vctno = 0C) */
#endif
```

(d) TRAPA #57ハンドラ、TRAPA #59ハンドラの開始アドレスを定義します。以下のように修正してください。

修正前

```
        (VP)hi_intdwn39,/* trapa #57                          (vctno = 39) */
        (VP)hi_intdwn3A,/* trapa #58                          (vctno = 3A) */
        (VP)hi_intdwn3B,/* trapa #59 (reserve HI-SH7)         (vctno = 3B) */
```

修正後

```
#if     hd_dbg                          /* case of hd_dbg = USE          */
        (VP)_0Dbrk_sbk, /* trapa #57                          (vctno = 39) */
        (VP)hi_intdwn3A,/* trapa #58                          (vctno = 3A) */
        (VP)_0Hend_svc, /* trapa #59 (reserve HI-SH7)         (vctno = 3B) */
#else                                   /* case of hd_dbg = NOTUSE       */
        (VP)hi_intdwn39,/* trapa #57                          (vctno = 39) */
        (VP)hi_intdwn3A,/* trapa #58                          (vctno = 3A) */
        (VP)hi_intdwn3B,/* trapa #59 (reserve HI-SH7)         (vctno = 3B) */
#endif
```

(e) SCI1送受信割込みハンドラの開始アドレスを定義します。以下のように修正してください。

修正前

```
        (VP)hi_intdwn68,/* SCI1 ERI1                          (vctno = 68) */
        (VP)hi_intdwn69,/* SCI1 RXI1                          (vctno = 69) */
        (VP)hi_intdwn6A,/* SCI1 TXI1                          (vctno = 6A) */
```

修正後

```
#if     hd_dbg                          /* case of hd_dbg = USE          */
        (VP)_0Dcnshdrer,/* SCI1 ERI1                          (vctno = 68) */
        (VP)_0Dcnshdrrx,/* SCI1 RXI1                          (vctno = 69) */
        (VP)_0Dcnshdrtx,/* SCI1 TXI1                          (vctno = 6A) */
#else                                   /* case of hd_dbg = NOTUSE       */
        (VP)hi_intdwn68,/* SCI1 ERI1                          (vctno = 68) */
        (VP)hi_intdwn69,/* SCI1 RXI1                          (vctno = 69) */
        (VP)hi_intdwn6A,/* SCI1 TXI1                          (vctno = 6A) */
#endif
```

４．５．２　MCU初期化ルーチンの修正

MCU初期化ルーチンを修正します。
図４－１６にMCU初期化ルーチンの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*            HI-SH7 MCU initialize routine ( Ver. 1.0 )                */
/*                                                                      */
/*                                                                      */
/*            Copyright (c)  Hitachi, Ltd. 1993.                        */
/*            Licensed Material of Hitachi, Ltd.                        */
/*                                                                      */
/*            HI-SH7(HS0700ITCN1SM) V1.0                                */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = himcuini.c ;                                            */
/*  NAME      = hi_mcuini ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 MCU initialize routine ;                         */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    <machine.h>
#include    "itron.h"
#include    "hdcommon.h"                                                ･･･(a)

#define VCTBASE             0x00000000    /* vector base address = 0x00000000 */
#define RAMSTA       (VW *)0x0fffe000     /* RAM start address   = 0x0fffe000 */
#define RAMEND       (VW *)0x0fffffff     /* RAM end   address   = 0x0fffffff */

#if     hd_dbg                           /* case of hd_dbg = USE             */･･･(b)
extern  void    _0Drs_sbk(void);         /* software break reset routine     */
extern  void    _0Drs_dbg(void);         /* debugger reset routine           */

static  UW      hot_start_flag;          /* hot start flag                   */
                                         /*      TRUE(1)     :   hot start   */
                                         /*      other       :   cold start  */
#endif

extern  void    _0Hrs_knl(void);         /* kernel reset routine             */
```

図４－１６　MCU初期化ルーチン（himcuini.c）の修正内容（１／２）

```
void    hi_mcuini(void)
{
    register    VW      *p;             /* pointer to memory                */

/*  set_vbr((VP *)VCTBASE); */          /* set vector base address to VBR   */

#if     hd_dbg                          /* case of hd_dbg = USE             */ ･･･(c)
    if(hot_start_flag == TRUE)          /* if hot_start_flag is TRUE        */
    {
        _ODrs__sbk();                   /* call software break reset routine*/
        goto hot_start;                 /* goto hot_start                   */
    }
#endif

    for(p = RAMSTA; p <= RAMEND; p++)   /* RAM clear                        */
        *p = 0x00000000;

#if     hd_dbg                          /* case of hd_dbg = USE             */ ･･･(d)
    hot_start_flag = TRUE;              /* set TRUE to hot_start_flag       */

hot_start:
    _ODrs__dbg();                       /* call debugger reset routine      */
#endif

    _OHrs__knl();                       /* go to kernel reset routine       */
}
```

図４－１６　MCU初期化ルーチン（himcuini.c）の修正内容（２／２）

(a) 共通ヘッダ（hdcommon.h）のインクルード文を追加します。以下の文を追加してください。

追加文

```
#include    "hdcommon.h"
```

なお、hd_dbg（デバッガの使用／未使用の定義変数）定義変数は、共通ヘッダで定義します。

(b) ソフトウェアブレーク解除ルーチン（_0Drs__sbk）、デバッガリセットルーチン（_0Drs__dbg）の開始アドレスをシンボル参照します。
また、ホット／コールドスタートの判断を行なうフラグ（hot_start_flag）を宣言します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  void    _0Drs__sbk(void);       /* software break reset routine     */
extern  void    _0Drs__dbg(void);       /* debugger reset routine           */

static  UW      hot_start_flag;         /* hot start flag                   */
                                        /*      TRUE(1)     :   hot start   */
                                        /*      other       :   cold start  */
#endif
```

(c) ホットスタートのときは、ソフトウェアブレーク解除ルーチン（_0Drs__sbk）をコール後、RAMクリアしない（hot_startラベルにジャンプ）ようにします。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
    if(hot_start_flag == TRUE)          /* if hot_start_flag is TRUE        */
    {
        _0Drs__sbk();                   /* call software break reset routine*/
        goto hot_start;                 /* goto hot_start                   */
    }
#endif
```

(d) ホット／コールドスタートフラグ（hot_start_flag）にTRUEを設定後、デバッガリセットルーチン（_0Drs__dbg）をコールするようにします。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
    hot_start_flag = TRUE;              /* set TRUE to hot_start_flag       */

hot_start:
    _0Drs__dbg();                       /* call debugger reset routine      */
#endif
```

４．５．３　システム初期化ハンドラの修正

システム初期化ハンドラを修正します。

図４－１７にシステム初期化ハンドラの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*            HI-SH7 system initial handler ( Ver. 1.0 )                */
/*                                                                      */
/*                                                                      */
/*            Copyright (c)  Hitachi, Ltd. 1993.                        */
/*            Licensed Material of Hitachi, Ltd.                        */
/*                                                                      */
/*            HI-SH7(HS0700ITCN1SM) V1.0                                */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hisysini.c ;                                            */
/*  NAME      = hi_sysini ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 system initial handler ;                         */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hdcommon.h"                                          ･･･(a)

extern  void    hi_tmrini(void);        /* timer initialize routine     */
extern  void    hi_sysdwn(W, ER, UW);   /* system down routine          */

INIHDR  hi_sysini(void)
{
    ER      ercd;                       /* error code                   */

#if     hd_dbg                          /* case of hd_dbg = USE         */  ･･･(b)
    ercd = ista_tsk(hd_dbgtskid);       /* start debugger task          */
    if(ercd < E_OK)                     /* if ista_tsk fail             */
        hi_sysdwn((W)2, ercd, (UW)0);   /* go to system down routine    */

    ercd = ista_tsk(hd_cnstskid);       /* start console task           */
    if(ercd < E_OK)                     /* if ista_tsk fail             */
        hi_sysdwn((W)3, ercd, (UW)0);   /* go to system down routine    */
#endif

    hi_tmrini();                        /* call timer initialize routine*/

    ercd = ista_tsk((ID)1);             /* start task (tskid = 1)       */
    if(ercd < E_OK)                     /* if ista_tsk fail             */
        hi_sysdwn((W)1, ercd, (UW)0);   /* go to system down routine    */
}
```

図４－１７　システム初期化ハンドラ（hisysini.c）の修正内容

(a)　共通ヘッダ（hdcommon.h）のインクルード文を追加します。以下の文を追加してください。

追加文

```
#include    "hdcommon.h"
```

なお、以下の定義変数は、共通ヘッダで定義します。
・hd_dbg（デバッガの使用／未使用の定義変数）
・hd_dbgtskid（オンラインデバッガタスクID）
・hd_cnstskid（コンソールタスクID）

(b)　オンラインデバッガタスク、コンソールタスクを起動するようにします。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE          */
    ercd = ista_tsk(hd_dbgtskid);       /* start debugger task           */
    if(ercd < E_OK)                     /* if ista_tsk fail              */
        hi_sysdwn((W)2, ercd, (UW)0);   /* go to system down routine     */

    ercd = ista_tsk(hd_cnstskid);       /* start console task            */
    if(ercd < E_OK)                     /* if ista_tsk fail              */
        hi_sysdwn((W)3, ercd, (UW)0);   /* go to system down routine     */
#endif
```

４．５．４　未定義割込み異常処理ルーチンの修正

未定義割込み異常処理ルーチンを修正します。

図４－１８に未定義割込み異常処理ルーチンの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 interrupt down routine ( Ver. 1.0 )              */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hiintdwn.c ;                                            */
/*  NAME      = hi_intdwn ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 interrupt down routine (undefine interrupt handler) ;*/
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    <machine.h>
#include    "itron.h"
#include    "hdcommon.h"                                          ･･･(a)

#if     hd_dbg                          /* case of hd_dbg = USE     */･･･(b)
extern  void    _0Dintdwn(UB);          /* intdwn routine of debugger */
#endif

void    hi_intdwn(vctno)
UB      vctno;                          /* interrupt vector number  */
{
    set_imask(15);                      /* mask all interrupt       */

#if     hd_dbg                          /* case of hd_dbg = USE     */･･･(c)
    _0Dintdwn(vctno);                   /* call intdwn routine of debugger */
#endif

    while(TRUE);                        /* endless loop             */
}
```

図４－１８　未定義割込み異常処理ルーチン（hiintdwn.c）の修正内容

(a) 共通ヘッダ（hdcommon.h）のインクルード文を追加します。以下の文を追加してください。

追加文

```
#include    "hdcommon.h"
```

なお、以下の定義変数は、共通ヘッダで定義します。
・hd_dbg（デバッガの使用／未使用の定義変数）

(b) デバッガの未定義割込み異常処理ルーチン（_0Dintdwn）の開始アドレスをシンボル参照します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  void    _0Dintdwn(UB);          /* intdwn. routine of debugger      */
#endif
```

(c) デバッガの未定義割込み異常処理ルーチン（_0Dintdwn）をコールするようにします。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
    _0Dintdwn(vctno);                   /* call intdwn. routine of debugger */
#endif
```

４．５．５　システム異常終了処理ルーチンの修正

システム異常終了処理ルーチンを修正します。

図４－１９にシステム異常終了処理ルーチンの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 system down routine ( Ver. 1.0 )                 */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hisysdwn.c ;                                            */
/*  NAME      = hi_sysdwn ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 system down routine ;                            */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    <machine.h>
#include    "itron.h"
#include    "hdcommon.h"                                                ･･･(a)

#if     hd_dbg                          /* case of hd_dbg = USE         */･･･(b)
extern  void    _0Dsysdwn(W, ER, UW);   /* sysdwn routine of debugger   */
#endif

void    hi_sysdwn(type, ercd, inf)
W       type;               /* type of system down                      */
                            /*      type >=  1 : system down of user program*/
                            /*      type ==  0 : setup table error      */
                            /*      type == -1 : context error of ext_tsk   */
                            /*      type == -2 : context error of exd_tsk   */
                            /*      type == -3 : context error of ret_int   */
                            /*      type == -4 : context error of sys_clk   */
                            /*      type <= -5 : system reserve         */
ER      ercd;               /* error code of system down                */
                            /*      type ==  0 : error code of setup table  */
                            /*      type == -1 : error code of ext_tsk      */
                            /*      type == -2 : error code of exd_tsk      */
                            /*      type == -3 : error code of ret_int      */
                            /*      type == -4 : error code of sys_clk      */
UW      inf;                /* information of system down               */
                            /*      type == -1 : address of ext_tsk call    */
                            /*      type == -2 : address of exd_tsk call    */
                            /*      type == -3 : address of ret_int call    */
                            /*      type == -4 : address of sys_clk call    */
{
    set_imask(15);          /* mask all interrupt                       */

#if     hd_dbg                          /* case of hd_dbg = USE         */･･･(c)
    _0Dsysdwn(type, ercd, inf);         /* call sysdwn routine of debugger */
#endif

    while(TRUE);            /* endless loop                             */
}
```

図４－１９　システム異常終了処理ルーチン（hisysdwn.c）の修正内容

(a) 共通ヘッダ（hdcommon.h）のインクルード文を追加します。以下の文を追加してください。

追加文

```
#include    "hdcommon.h"
```

なお、以下の定義変数は、共通ヘッダで定義します。
・hd_dbg（デバッガの使用／未使用の定義変数）

(b) デバッガのシステム異常終了処理ルーチン（_0Dsysdwn）の開始アドレスをシンボル参照します。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  void    _0Dsysdwn(W, ER, UW);   /* sysdwn. routine of debugger      */
#endif
```

(c) デバッガのシステム異常終了処理ルーチン（_0Dsysdwn）をコールするようにします。以下の文を追加してください。

追加文

```
#if     hd_dbg                          /* case of hd_dbg = USE             */
    _0Dsysdwn(type, ercd, inf);         /* call sysdwn. routine of debugger */
#endif
```

４．５．６　タスク用スタック領域定義ヘッダの修正

タスク用スタック領域定義ヘッダを修正します。

図４－２０にタスク用スタック領域定義ヘッダの修正内容を示します。 の網かけ部分を修正してください。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 header file for task stack ( Ver. 1.0 )          */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hitskstk.h ;                                            */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 header file for task stack ;                     */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#ifndef  hd_dbg                                                  ･･･(a)
#include "hdcommon.h"
#endif

/************************************************************************/
/*                                                                      */
/*          Definition constant size of task stack                      */
/*                                                                      */
/************************************************************************/
#define KNLTUSESZ       108                 /* size of kernel use       */

/************************************************************************/
/*                                                                      */
/*          Definition task stack size of interrupt use                 */
/*                                                                      */
/*  Usage   :                                                           */
/*                                                                      */
/*  #define inttusesz       12 * <hi_uppintnst> + 20 * <hi_lowintnst>   */
/*                                                                      */
/*      <hi_uppintnst>  :   nest number of upper interrupt              */
/*                          (>  kernel mask level)                      */
/*      <hi_lowintnst>  :   nest number of lower interrupt              */
/*                          (<= kernel mask level)                      */
/*                                                                      */
/************************************************************************/
#define inttusesz       12 * 0 + 20 * 2 + ADDTSKSTKSZ                ･･･(b)
                                        /* size of interrupt use       */
```

図４－２０　タスク用スタック領域定義ヘッダ（hitskstk.h）の修正内容（１／２）

```
/*****************************************************************************/
/*                                                                           */
/*          Definition task stack size (multiple of 4)                       */
/*                                                                           */
/*  Usage   :                                                                */
/*                                                                           */
/*  #define hi_tskstk<n>    KNLTUSESZ + inttusesz + <tskusesz> + <excusesz>  */
/*                                                                           */
/*      <n>             :   task stack number                                */
/*      <tskusesz>      :   size of task use                                 */
/*            (caution !!  if using chg_ims by the task, add 28 to tskusesz)*/
/*      <excusesz>      :   size of exception use                            */
/*                                                                           */
/*****************************************************************************/
#define hi_tskstksz1    (KNLTUSESZ + inttusesz +  44 + 0)  /* stack no. = 1 */
#define hi_tskstksz2    (KNLTUSESZ + inttusesz + 108 + 0)  /* stack no. = 2 */
#define hi_tskstksz3    (KNLTUSESZ + inttusesz + 108 + 0)  /* stack no. = 3 */
#define hi_tskstksz4    (KNLTUSESZ + inttusesz + 108 + 0)  /* stack no. = 4 */
```

図４－２０　タスク用スタック領域定義ヘッダ（hitskstk.h）の修正内容（２／２）

(a)　以下の文を追加してください。

追加文

```
#ifndef     hd_dbg
#include    "hdcommon.h"
#endif
```

なお、以下の定義変数は、共通ヘッダで定義します。

・hd_dbg（デバッガの使用／未使用の定義変数）

・ADDTSKSTKSZ（タスク用スタックサイズの加算値の定義変数）

(b)　割込みが使用するタスクのスタックサイズ（inttusesz）を補正するため、以下のように修正してください。

修正前

```
#define inttusesz       12 * 0 + 20 * 2     /* size of interrupt use        */
```

修正後

```
#define inttusesz       12 * 0 + 20 * 2 + ADDTSKSTKSZ
                                            /* size of interrupt use        */
```

４．５．７　割込みハンドラ用スタック領域定義ヘッダの修正

割込みハンドラ用スタック領域定義ヘッダを修正します。

図４－２１に割込みハンドラ用スタック領域定義ヘッダの修正内容を示します。 の網かけ部分を修正してください。

```
/*****************************************************************************/
/*                                                                           */
/*                                                                           */
/*             HI-SH7 header file for interrupt handler stack ( Ver. 1.0 ) */
/*                                                                           */
/*                                                                           */
/*             Copyright (c)  Hitachi, Ltd. 1993.                            */
/*             Licensed Material of Hitachi, Ltd.                            */
/*                                                                           */
/*             HI-SH7(HS0700ITCN1SM) V1.0                                    */
/*                                                                           */
/*                                                                           */
/*****************************************************************************/
/*****************************************************************************/
/*SPECIFICATIONS ;                                                           */
/*  FILE      = hiintstk.h ;                                                 */
/*  DATE      = 93/02/01 ;                                                   */
/*  AUTHOR    = Hitachi, Ltd. ;                                              */
/*  FUNCTION  = HI-SH7 header file for interrupt handler stack ;             */
/*  ATTRIBUTE = PUBLIC ;                                                     */
/*  HISTORY   = V1.0 ;                                                       */
/*END OF SPECIFICATIONS ;                                                    */
/*****************************************************************************/
#ifndef  hd_dbg                                                        ･･･(a)
#include "hdcommon.h"
#endif

/*****************************************************************************/
/*                                                                           */
/*          Definition constant size of interrupt handler stack              */
/*                                                                           */
/*****************************************************************************/
#define KNLIUSESZ       132                 /* size of kernel use          */
```

図４－２１　割込みハンドラ用スタック領域定義ヘッダ（hiintstk.h）の修正内容（１／２）

```
/************************************************************************/
/*                                                                      */
/*          Definition stack size of interrupt handler (multiple of 4)  */
/*                                                                      */
/*  Usage   :                                                           */
/*  case of interrupt level =  not use                                  */
/*    #define hi_intstk<n>  0                                           */
/*                                                                      */
/*  case of interrupt level >  kernel mask level                        */
/*    #define hi_intstk<n>  <othusesz> + <curusesz> + <excusesz>        */
/*                                                                      */
/*  case of interrupt level <= kernel mask level                        */
/*    #define hi_intstk<n>  KNLIUSESZ + <othusesz> + <curusesz> + <excusesz>*/
/*                                                                      */
/*      <n>             :   interrupt level number                      */
/*      <othusesz>      :   size of other interrupt use                 */
/*                      =   12 * <uppintnst> + 20 * <lowintnst>         */
/*          <uppintnst> :   nest number of upper interrupt              */
/*                          (>  kernel mask level, > current interrupt level)*/
/*          <lowintnst> :   nest number of lower interrupt              */
/*                          (<= kernel mask level, > current interrupt level)*/
/*      <curusesz>      :   size of current interrupt handler use       */
/*      <excusesz>      :   size of exception use                       */
/*                                                                      */
/************************************************************************/
#define hi_intstksz1    0                                       /* level =  1 */
#define hi_intstksz2    0                                       /* level =  2 */
#define hi_intstksz3    0                                       /* level =  3 */
#define hi_intstksz4    0                                       /* level =  4 */
#define hi_intstksz5    0                                       /* level =  5 */
#define hi_intstksz6    0                                       /* level =  6 */
#define hi_intstksz7    0                                       /* level =  7 */
#define hi_intstksz8    0                                       /* level =  8 */
#define hi_intstksz9    0                                       /* level =  9 */
#define hi_intstksz10   0                                       /* level = 10 */
#define hi_intstksz11   (KNLIUSESZ + 12 * 0 + 20 * 1 + ADDINTSTKSZ + 68 + 0)   ･･･(b)
                                                                /* level = 11 */
#define hi_intstksz12   0                                       /* level = 12 */
#define hi_intstksz13   (KNLIUSESZ + 12 * 0 + 20 * 0 + ADDINTSTKSZ + 12 + 0)   ･･･(b)
                                                                /* level = 13 */
#define hi_intstksz14   0                                       /* level = 14 */
#define hi_intstksz15   0                                       /* level = 15 */
```

図４－２１　割込みハンドラ用スタック領域定義ヘッダ（hiintstk.h）の修正内容（２／２）

(a)　以下の文を追加してください。

追加文

```
#ifndef     hd_dbg
#include    "hdcommon.h"
#endif
```

なお、以下の定義変数は、共通ヘッダで定義します。

・hd_dbg（デバッガの使用／未使用の定義変数）

・ADDINTSTKSZ（割込みハンドラ用スタックサイズの加算値の定義変数）

(b)　使用する割込みレベルのスタックサイズ（hi_inistkszxx）を補正するため、以下のように修正してください。

修正前

```
#define hi_intstkszxx   (KNLIUSESZ + 12 * 0 + 20 * 1 + 68 + 0)
```

修正後

```
#define hi_intstkszxx   (KNLIUSESZ + 12 * 0 + 20 * 1 + ADDINTSTKSZ + 68 + 0)
```

４．６　実行形式プログラムの作成

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。
HI-SH7のmakeファイルを利用して、本デバッガが提供しているファイルのコンパイル、およびデバッガライブラリとの結合を自動的に行うことができます。
makeファイルの修正方法については、**「４．８　makeファイルの修正」**を参照してください。

４．６．１　プログラムのコンパイル

(1)　デバッガセットアップテーブル（hdsuptbl.c）のコンパイル
デバッガセットアップテーブルをコンパイルして、オブジェクトモジュールを作成します。
以下のコマンドを入力して、コンパイルしてください。

% shc△-section=p=hddbgp,c=hddbgc,d=hddbgd,b=hddbgwrk△hdsuptbl.c(RET)

(2)　HI-SH7セットアップテーブルおよびHI-SH7システムのユーザプログラムのコンパイル
本デバッガを利用するために修正したHI-SH7セットアップテーブルおよびHI-SH7システムのユーザプログラムをコンパイルします。

(a)　HI-SH7セットアップテーブル（hisuptbl.c）
(b)　HI-SH7システムのユーザプログラム
・ベクタテーブル（hivcttbl.c）
・MCU初期化ルーチン（himcuini.c）
・システム初期化ハンドラ（hisysini.c）
・未定義割込み異常処理ルーチン（hiintdwn.c）
・システム異常終了処理ルーチン（hisysdwn.c）
・タスク用スタック領域定義（hitskstk.c）
・割込みハンドラ用スタック領域定義（hiintstk.c）

HI-SH7セットアップテーブル、およびHI-SH7ユーザプログラムのコンパイルは、HI-SH7システムの構築作業と同じ方法で行います。コンパイル方法については、**『HI-SH7　構築マニュアル』**の**「３．６　実行形式プログラムの作成」**を参照してください。

４．６．２　システムの結合

HI-SH7システムと本デバッガをリンケージエディタで結合し、実行形式プログラム（アブソリュートロードモジュール）を作成します。提供しているリンケージエディタサブコマンドファイル（himake.sub）を参考に、結合したいオブジェクトモジュール、セクションのメモリ配置を指定してください。

(1)　リンケージエディタの実行
リンケージエディタの実行は、以下のコマンドを入力します。

% lnk△-subcommand=<リンケージサブコマンドファイル名>(RET)

(2)　リンケージサブコマンドファイルの修正
HI-SH7が提供しているリンケージサブコマンドファイル（himake.sub）を修正して、本デバッガを結合します。

図４－１５にリンケージサブコマンドファイルの修正内容を示します。 の網かけ部分を追加、修正してください。

```
;************************************************************************;
;*                                                                      *;
;*                                                                      *;
;*            HI-SH7 subcommand file for linkage editor ( Ver. 1.0 )    *;
;*                                                                      *;
;*                                                                      *;
;*            Copyright (c)  Hitachi, Ltd. 1993.                        *;
;*            Licensed Material of Hitachi, Ltd.                        *;
;*                                                                      *;
;*            HI-SH7(HS0700ITCN1SM) V1.0                                *;
;*                                                                      *;
;*                                                                      *;
;************************************************************************;
;************************************************************************;
;*SPECIFICATIONS ;                                                      *;
;*  FILE      = himake.sub ;                                            *;
;*  DATE      = 93/02/01 ;                                              *;
;*  AUTHOR    = Hitachi, Ltd. ;                                         *;
;*  FUNCTION  = HI-SH7 subcommand file for linkage editor ;             *;
;*  ATTRIBUTE = PUBLIC ;                                                *;
;*  HISTORY   = V1.0 ;                                                  *;
;*END OF SPECIFICATIONS ;                                               *;
;************************************************************************;
;************************************************************************;
;*                                                                      *;
;*                              object file                             *;
;*                                                                      *;
;************************************************************************;
input   hivcttbl.obj,&                  ;* vector table                 *;
        himcuini.obj,&                  ;* MCU initialize routine       *;
        hiintdwn.obj,&                  ;* interrupt down routine       *;
        hisysdwn.obj,&                  ;* system down routine          *;
        hisuptbl.obj,&                  ;* setup table                  *;
        hdsuptbl.obj,&                  ;* setup table of debugger      *;…(a)
        hisysini.obj,&                  ;* system initial handler       *;
        hitmrdrv.obj,&                  ;* timer driver                 *;
        hicnsdrv.obj,&                  ;* console driver               *;
        himempol.obj,&                  ;* memory pool area            *;
        hitskstk.obj,&                  ;* task stack area              *;
        hiintstk.obj,&                  ;* interrupt stack area         *;
        hitrcbuf.obj                    ;* trace buffer area           *;

;************************************************************************;
;*                                                                      *;
;*                              library file                            *;
;*                                                                      *;
;************************************************************************;
library hiknlp.lib,&                    ;* kernel (parameter check)     *;
        hddbg.lib,&                     ;* debugger                     *;…(b)
        hicif.lib                       ;* C language interface         *;
```

図４－２２　リンケージサブコマンドファイル（himake.sub）の修正内容（１／２）

```
;************************************************************************;
;*                                                                      *;
;*                              section                                 *;
;*                                                                      *;
;************************************************************************;
;*------------------ ROM area (top address : 0x00000000) ----------------*;
start   hivcttbl,&                      ;* vector table                  *;
        himcuini,&                      ;* MCU initialize routine        *;
        hiintdwn,&                      ;* interrupt down routine        *;
        hisysdwn,&                      ;* system down routine           *;
        hisuptbl,&                      ;* setup table                   *;
        hiromtop,&                    ;* ROM top    area of HI-SH7      *;···(c)
        hiknl,&                         ;* kernel                        *;
        hirombtm,&                    ;* ROM bottom area of HI-SH7      *;···(c)
        hisysini,&                      ;* system initial handler       *;
        hdromtop,&                    ;* ROM top    area of debugger    *;···(d)
        hddbgp,hddbgc,hddbgd,&          ;* debugger                      *;···(e)
        hdrombtm,&                    ;* ROM bottom area of debugger   *;···(d)
        hitmrdrv,&                      ;* timer driver                 *;
        hitmrcst,&                      ;* constant area of timer driver *;
        hicnsdrv,&                      ;* console driver               *;
        hicnscst,&                      ;* constant area of console driver *;
        hicif(00000000)                 ;* C language interface         *;

;*------------------ RAM area (top address : 0x01000000) ----------------*;···(f)
start   hdramtop,&                    ;* RAM top    area of debugger    *;···(g)
        hddbgwrk,&                      ;* work area of debugger        *;···(h)
        hdrambtm,&                    ;* RAM bottom area of debugger   *;···(g)
        hitrcbuf(01000000)            ;* trace buffer area            *;···(i)

;*------------------ RAM area (top address : 0x0fffe000) ----------------*;
start   hiramtop,&                    ;* RAM top    area of HI-SH7      *;···(j)
        hiknlwrk,&                      ;* work area of kernel          *;
        hirambtm,&                    ;* RAM bottom area of HI-SH7      *;···(j)
        hicnswrk,&                      ;* work area of console driver  *;
        himempol,&                      ;* memory pool area             *;
        hitskstk,&                      ;* task stack area              *;
        hiintstk(0fffe000)            ;* interrupt stack area          *;···(j)

;************************************************************************;
;*                                                                      *;
;*                              other                                   *;
;*                                                                      *;
;************************************************************************;
entry   _hi_mcuini                      ;* entry address     : _hi_mcuini   *;
output  hisystem.abs                    ;* output file       : hisystem.abs *;
print   hisystem.map                    ;* linkage map file  : hisystem.map *;
form    a                               ;* form              : absolute     *;
debug                                   ;* debug information : output       *;
exit                                    ;* exit                           *;
```

図４－２２　リンケージサブコマンドファイル（himake.sub）の修正内容（２／２）

(a) 結合するオブジェクトモジュールに、デバッガセットアップテーブルのオブジェクトモジュール（hdsuptbl.obj）を追加します。

(b) 結合するライブラリに、デバッガライブラリ（hddbg.lib）を追加します。

(c) HI-SH7カーネルのROM領域を保護するため、先頭アドレスセクション（hiromtop）と最終アドレスセクション（hirombtm）を追加します。この2つのセクション間に、HI-SH7カーネルのセクション（hiknl）を指定してください。*1

(d) デバッガのROM領域を保護するため、先頭アドレスセクション（hdromtop）と最終アドレスセクション（hdrombtm）を追加します。*1

(e) デバッガのROM領域のセクション（hddbgp, hddbgc, hddbgd）を追加します。

(f) 外部RAMメモリ（先頭アドレス=H'01000000）に、デバッガのRAM領域とHI-SH7のトレースバッファ領域を定義します。*2

(g) デバッガのRAM領域を保護するため、先頭アドレスセクション（hdramtop）と最終アドレスセクション（hdrambtm）を追加します。*1

(h) デバッガのRAM領域のセクション（hddbgwrk）を追加します。

(i) HI-SH7のトレースバッファ領域のセクション（hitrcbuf）を外部RAM領域に移動します。

(j) HI-SH7カーネルのRAM領域を保護するため、先頭アドレスセクション（hiramtop）と最終アドレスセクション（hirambtm）を追加します。この2つのセクション間に、HI-SH7カーネルの作業領域のセクション（hiknlwrk）を指定してください。*1
内蔵RAM（先頭アドレス=H'0fffe000）に、HI-SH7カーネルのRAM領域とHI-SH7コンソールドライバ作業領域、メモリプール領域、タスク用スタック領域、割込みハンドラ用スタック領域を定義します。

【注】*1 指定した保護領域は、デバッグコマンドによるブレークポイントの設定やデータの書込みが禁止されます。
*2 SHマイコンの内蔵ROMや内蔵RAMだけでは、システム全体のメモリ容量が不足します。このため、外部メモリを用意してください。

４．７　システムの起動

ROMを実機に搭載する方法を説明します。

(1)　アブソリュートロードモジュールをSタイプロードモジュールに変換します。

% cnvs△<アブソリュートロードモジュールのファイル名>(RET)

(2)　SタイプロードモジュールをROMライタへ転送し、ROMに書き込みます。

(3)　ROMを実機に搭載します。電源投入（パワーオンリセット）により、システムが起動します。

４．８　makeファイルの修正

HI-SH7では、システムを自動的に構築するためのmakeファイル（himake）を提供しています。ホストコンピュータのmake機能を利用すれば、プログラムの更新状況（ファイル日付）から、必要なソースプログラムのコンパイル（またはアセンブル）とシステムの結合を自動的に行なうことができ、プログラムの変更管理が容易になります。

本デバッガは、HI-SH7の提供しているmakeファイルを修正することで、HI-SH7システムへの組込みが容易になります。

図４－２３にmakeファイルの修正内容を示します。 の網かけ部分を追加、修正してください。なお、本製品は、この修正を行なったファイルを、サンプルプログラムとして提供しています。

```
#*****************************************************************************#
#*                                                                           *#
#*                                                                           *#
#*             HI-SH7 make file ( Ver. 1.0 )                                 *#
#*                                                                           *#
#*                                                                           *#
#*             Copyright (c)  Hitachi, Ltd. 1993.                            *#
#*             Licensed Material of Hitachi, Ltd.                            *#
#*                                                                           *#
#*             HI-SH7(HS0700ITCN1SM) V1.0                                    *#
#*                                                                           *#
#*                                                                           *#
#*****************************************************************************#
#*****************************************************************************#
#*SPECIFICATIONS ;                                                           *#
#*  FILE      = himake ;                                                     *#
#*  DATE      = 93/02/01 ;                                                   *#
#*  AUTHOR    = Hitachi, Ltd. ;                                              *#
#*  FUNCTION  = HI-SH7 make file ;                                           *#
#*  ATTRIBUTE = PUBLIC ;                                                     *#
#*  HISTORY   = V1.0 ;                                                       *#
#*END OF SPECIFICATIONS ;                                                    *#
#*****************************************************************************#
hisystem: \
        hisystem.abs
        lnk -subcommand=himake.sub
        cnvs hisystem.abs hisystem.mot
hisystem.abs: \
        hicnsdrv.obj hiintdwn.obj hiintstk.obj himcuini.obj himempol.obj \
        hisuptbl.obj hisysdwn.obj hisysini.obj hitmrdrv.obj hitrcbuf.obj \
        hitskstk.obj hivcttbl.obj :\                                    ･･･(a)
        hdsuptbl.obj
hicnsdrv.obj: \
        hicnsdrv.c itron.h hicnsdrv.h hiintstk.h
        shc hicnsdrv.c -debug -section=p=hicnsdrv,c=hicnscst,b=hicnswrk
hiintdwn.obj: \
        hiintdwn.c itron.h :\                                           ･･･(b)
        hdcommon.h
        shc hiintdwn.c -debug -section=p=hiintdwn
hiintstk.obj: \
        hiintstk.c itron.h hiintstk.h :\
        hdcommon.h                                                      ･･･(b)
        shc hiintstk.c -debug -section=b=hiintstk
```

図４－２３　makeファイル（himake）の修正内容（１／２）

```
himcuini.obj: \
        himcuini.c itron.h \                                                    ･･･(b)
        hdcommon.h
        shc himcuini.c -debug -section=p=himcuini,b=hddbgwrk                    ･･･(c)
himempol.obj: \
        himempol.c itron.h himempol.h
        shc himempol.c -debug -section=b=himempol
hisuptbl.obj: \
        hisuptbl.c itron.h hisuptbl.h hisuptbl.inc hitskstk.h \
        himempol.h hitrcbuf.h \                                                 ･･･(d)
        hdcommon.h hisuptbl.dbg
        shc hisuptbl.c -debug -section=c=hisuptbl,b=hiknlwrk
hisysdwn.obj: \
        hisysdwn.c itron.h \                                                    ･･･(b)
        hdcommon.h
        shc hisysdwn.c -debug -section=p=hisysdwn
hisysini.obj: \
        hisysini.c itron.h \                                                    ･･･(b)
        hdcommon.h
        shc hisysini.c -debug -section=p=hisysini
hitmrdrv.obj: \
        hitmrdrv.c itron.h hitmrdrv.h hiintstk.h
        shc hitmrdrv.c -debug -section=p=hitmrdrv,c=hitmrcst
hitrcbuf.obj: \
        hitrcbuf.c itron.h hitrcbuf.h
        shc hitrcbuf.c -debug -section=b=hitrcbuf
hitskstk.obj: \
        hitskstk.c itron.h hitskstk.h \                                         ･･･(b)
        hdcommon.h
        shc hitskstk.c -debug -section=b=hitskstk
hivcttbl.obj: \
        hivcttbl.c itron.h \                                                    ･･･(b)
        hdcommon.h
        shc hivcttbl.c -debug -section=c=hivcttbl
hdsuptbl.obj: \                                                                 ･･･(e)
        hdsuptbl.c itron.h hdsuptbl.h hdsuptbl.inc hdcommon.h
        shc hdsuptbl.c -debug -section=p=hddbgp,c=hddbgc,d=hddbgd,b=hddbgwrk
```

図４－２３　makeファイル（himake）の修正内容（２／２）

(a)　アブソリュートロードモジュール（hisystem.abs）の依存ファイルとして、デバッガセットアップテーブルオブジェクトモジュール（hdsuptbl.obj）を追加します。

(b)　依存ファイルとして、共通ヘッダ（hdcommon.h）を追加します。

(c)　MCU初期化ルーチン（himcuini.obj）のセクション指定として、未初期化データ領域セクション（",b=hddbgwrk"）を追加します。このセクションは、ホット／コールドスタートフラグの領域です。詳細は、**「４．５．２　MCU初期化ルーチンの修正」**を参照してください。

(d)　HI-SH7セットアップテーブル（hisuptbl.obj）の依存ファイルとして、共通ヘッダ（hdcommon.h）とデバッガ用HI-SH7ユーザ定義解析インクルード（hisuptbl.dbg）を追加します。

(e)　デバッガセットアップテーブル（hdsuptbl.obj）の依存ファイルとコンパイルコマンドを追加します。

付録Ａ．　メモリ容量の算出

Ａ．１　システム全体のメモリ領域

表Ａ－１にシステム全体のメモリ領域を示します。
システム全体のメモリ容量は、表Ａ－１に示す各領域の総和となります。

表Ａ－１　システム全体のメモリ領域

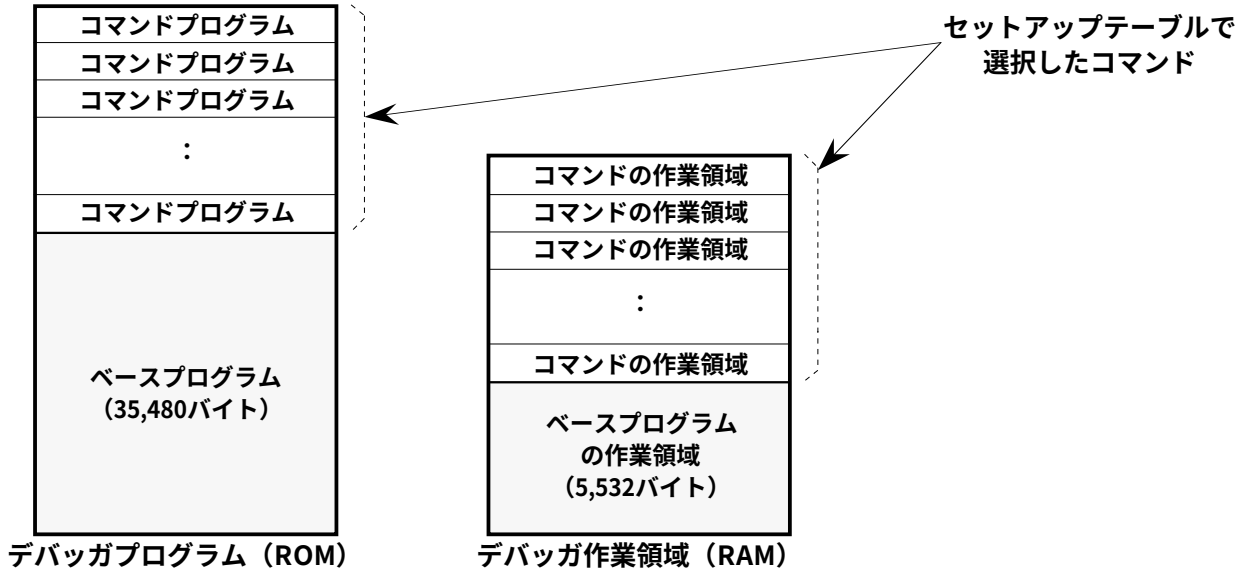
項番	メモリ属性	分　類	領　域　名　称	セクション名
1	ROM	HI-SH7領域	MCU初期化ルーチン	himcuini
2			未定義割込み異常処理ルーチン	hiintdwn
3			システム異常終了処理ルーチン	hisysdwn
4			HI-SH7セットアップテーブル	hisuptbl
5			カーネルプログラム	hiknl
6			システム初期化ハンドラ	hisysini
7		ユーザ領域	ベクタテーブル	hivcttbl
8			HI-SH7・C言語インタフェースライブラリ	hicif
9			ユーザプログラム	ユーザ指定
１０		デバッガ領域	デバッガプログラム	hddbgp, hddbgc, hddbgd
１１	RAM	HI-SH7領域	カーネル作業領域	hiknlwrk
１２			メモリプール領域	himempol
１３			タスク用スタック領域	hitskstk
１４			割込みハンドラ用スタック領域	hiintstk
１５			トレースバッファ領域	hitrcbuf
１６		ユーザ領域	ユーザ作業領域	ユーザ指定
１７		デバッガ領域	デバッガ作業領域	hddbgwrk

Ａ．２　デバッガのメモリ容量

図Ａ－１にデバッガのメモリ構成を示します。

本デバッガは、デバッガの基本的な処理を行なうプログラム（ベースプログラム）と、各コマンドを実行するプログラムで構成しています。このプログラム構成によって、ユーザは、必要とするコマンドをセットアップテーブルで選択することができます（メモリ容量を小さくすることができます）。

ゆえに、デバッガのROM容量は、ベースプログラムとセットアップテーブルで選択したコマンドのプログラムサイズの総和となります。同様に、デバッガのRAM容量は、ベースプログラムとセットアップテーブルで選択したコマンドの作業領域サイズの総和となります。



図Ａ－１　デバッガのメモリ構成

したがって、デバッガのメモリ容量は次式で求めます。各コマンドのメモリ容量は、表Ａ－２を参照してください。

ROM容量　＝　ベースプログラム（35,480バイト）　＋　選択した全コマンドプログラム

RAM容量　＝　ベースプログラムの作業領域（5,532バイト）＋　選択した全コマンドの作業領域

例えば、DUMPコマンドとGOコマンドのみを選択した場合、以下のようになります。

ROM容量 = 35480（ベースプログラム） + 2880（DUMP） + 192（GO）　　= 38552バイト
RAM容量 = 5532（ベースプログラムの作業領域） + 100（DUMP） + 0（GO）　　= 5632バイト

なお、実際に必要となるメモリ容量は、各コマンドで共通利用している内部プログラムや作業領域があるため、各コマンドのメモリサイズの総和よりも小さくなることがあります（算出したメモリ容量は、目安としてください）。

表Ａ－２　各コマンドのメモリ容量（単位：バイト）

項番	コマンド名	ROM	RAM	項番	コマンド名	ROM	RAM
1	.<レジスタ名>	2308	0	30	POL_FLG	560	0
2	ASSEMBLE	13628	96	31	RCV_MSG	500	0
3	BREAK	1712	512[*1]	32	REGISTER	1004	0
4	BREAK_UBC	1468	16	33	REL_BLK	660	0
5	CAN_WUP	496	0	34	REL_WAI	416	0
6	CHG_PRI	516	0	35	REQ_SEM	408	0
7	CLR_FLG	376	0	36	RESET	200	0
8	CRE_TSK	612	0	37	ROT_RDQ	200	0
9	DEL_TSK	416	0	38	RSM_TSK	416	0
10	DISASSEMBLE	480	16	39	RUN_TSK	464	0
11	DUMP	2880	100	40	SAVE	5776	440
12	FILL	2816	0	41	SEM_STS	956	0
13	FLG_STS	948	0	42	SET_FLG	348	0
14	GET_BLK	472	0	43	SET_TIM	224	0
15	GET_TIM	240	0	44	SIG_SEM	380	0
16	GO	192	0	45	SND_MSG	416	0
17	IHELP	5716	0	46	STA_TSK	392	0
18	IID	0	0	47	STEP	276	24
19	ISTATUS	432	0	48	STEP_OVER	276	24
20	ITRACE	10840	0	49	STEP_UBC	276	28
21	ITRACE_ACTIVATE	11168	0	50	STP_TSK	464	0
22	ITRACE_BUFFER	1048	0	51	SUS_TSK	416	0
23	ITRACE_SEARCH	11036	0	52	SVCDEF	1580	0
24	LOAD	7624	3588	53	SYSDEF	388	0
25	MBX_STS	928	0	54	TER_TSK	392	0
26	MEMORY	3192	0	55	TSKDEF	748	0
27	MOVE	1212	0	56	TSK_STS	2008	104
28	MPLDEF	544	0	57	VERIFY	7624	3588
29	MPL_STS	932	0	58	WUP_TSK	416	0

【注】*1　BREAKコマンドのRAM容量は、次式で求めてください

RAM容量 = 16バイト　×　ブレークポイント設定可能数（初期値:32）

付録Ｂ．　システムの構築例

Ｂ．１　概要

HI-SH7の例題システムに、デバッガを組込む例を示します。システムの構築は、UNIX上で行ないます。
なお、HI-SH7の例題システムについての詳細は、**『HI-SH7　UNIX用構築マニュアル』**の**「付録Ｂ．　システムの構築例」**を参照してください。

Ｂ．２　システム構築の環境

Ｂ．２．１　ハードウェア構成

例題システムを構築するために必要なハードウェアを次に示します。
・ホストコンピュータ（UNIX）
・ROMライタ

Ｂ．２．２　ソフトウェア構成

例題システムを構築するために必要なソフトウェアを次に示します。

(1)　ユーティリティソフトウェア
・SHシリーズ　Cコンパイラ
・Hシリーズ　リンケージエディタ
・Hシリーズ　オブジェクトコンバータ
・Hシリーズ　インタフェースソフト

(2)　デバッガのソフトウェア
表Ｂ－１に本製品が提供しているファイルを示します。表Ｂ－１に示した提供ファイルを使用して、例題システムを構築します。

表Ｂ－１　提供ファイル

項番	格納ディレクトリ	ファイル名	内　　容
1	shdbg	hdcommon.h	共通ヘッダ（デバッガ、HI-SH7）
2		hdsuptbl.h	デバッガセットアップテーブルヘッダ
3		hdsuptbl.c	デバッガセットアップテーブルソースプログラム
4		hdsuptbl.inc	デバッガセットアップテーブルインクルード
5		hddbg.lib	デバッガライブラリ
6		hisuptbl.dbg	HI-SH7・デバッガ組込み用セットアップテーブルインクルード
7	shdbg/sample[*1]	himake	HI-SH7・システム構築用makeファイル
8		himake.sub	HI-SH7・システム構築用リンケージサブコマンドファイル
9		hivcttbl.c	HI-SH7・ベクタテーブルソースプログラム
１０		hisuptbl.c	HI-SH7・セットアップテーブルソースプログラム
１１		himcuini.c	HI-SH7・MCU初期化ルーチンソースプログラム
１２		hisysini.c	HI-SH7・システム初期化ハンドラソースプログラム
１３		hiintdwn.c	HI-SH7・未定義割込み異常処理ルーチンソースプログラム
１４		hisysdwn.c	HI-SH7・システム異常終了処理ルーチンソースプログラム
１５		hitskstk.h	HI-SH7・タスク用スタック領域定義ヘッダ
１６		hiintstk.h	HI-SH7・割込みハンドラ用スタック領域定義ヘッダ

*1　HI-SH7サンプルプログラムをデバッガを組込むために修正したファイルが格納されています。

Ｂ．３　例題システムの概要

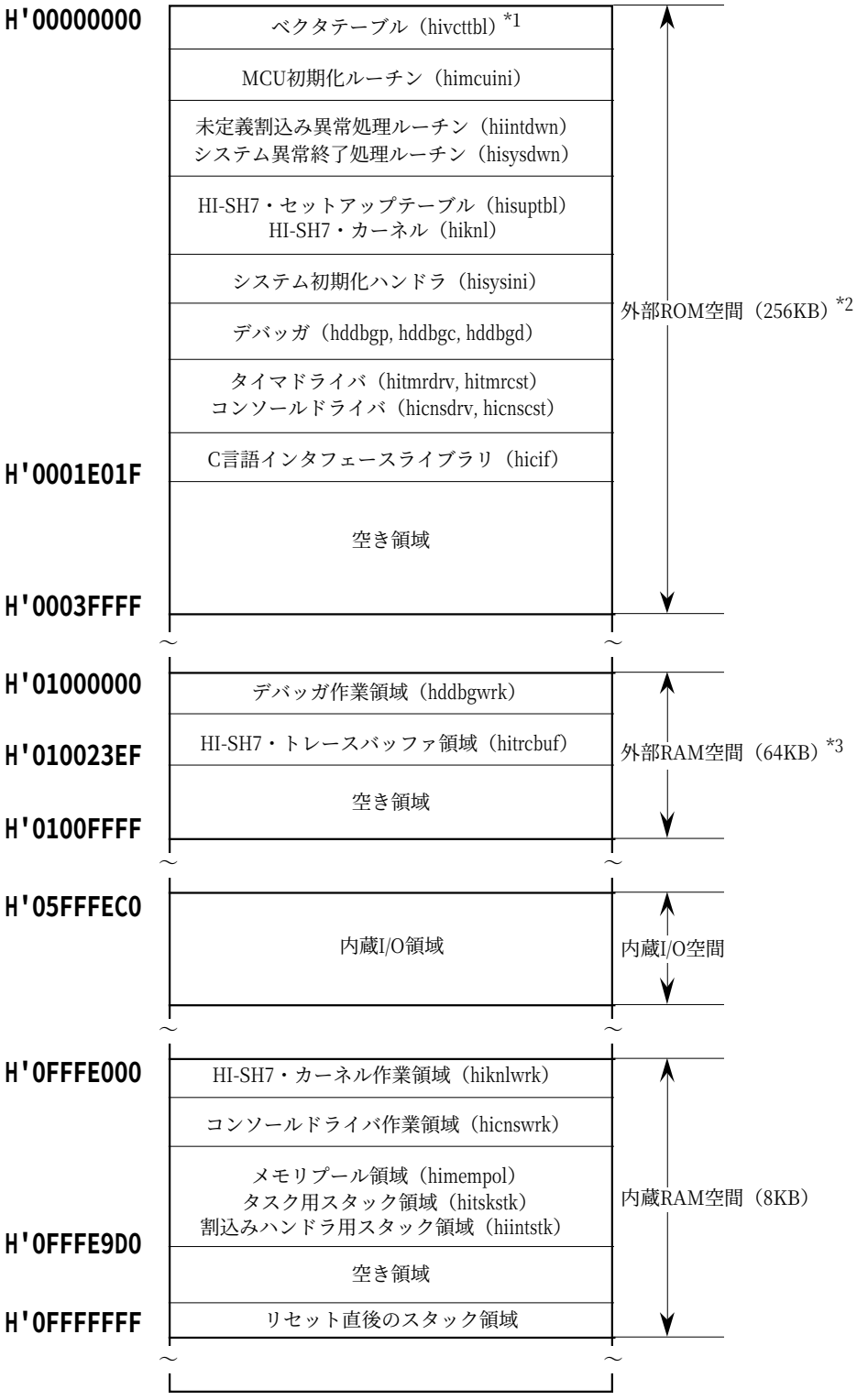
表Ｂ－２に例題システムの概要を示します。

表Ｂ－２　例題システムの概要

項番	項　目	構　築　内　容	
1	MCU	SH7032(20MHz)	
2	ROM	内蔵ROM	なし
		外部ROM	256KB
3	RAM	内蔵RAM	8KB
		外部RAM	64KB
4	予約タスクID	オンラインデバッガタスク	6
		コンソールタスク	7
5	予約メイルボックスID	オンラインデバッガタスク受信用	4
		コンソールタスク受信用	5
6	使用コマンド	全コマンド	
7	BREAKコマンド	ブレークポイント数	32
8	オフラインデバッガ	リセット時に起動	
9	コンソール	使用デバイス	SCI1
		転送速度	9600bps
		割込みレベル	14

B．４　例題システムのメモリマップ

図Ｂ－１に例題システムのメモリマップを示します。



*1　（　）内はセクション名です
*2　外部ROM空間は、1MビットROMを2ケ使用した場合の例です
*3　外部RAM空間は、256KビットRAMを2ケ使用した場合の例です

図Ｂ－１　例題システムのメモリマップ

Ｂ．５　例題システムの共通ヘッダ

図Ｂ－２に例題システムの共通ヘッダ(hdcommon.h)を示します。

```
/*****************************************************************************/
/*                                                                           */
/*                                                                           */
/*             SH7000 HI-SH7 DEBUGGER common header file ( Ver. 1.0 )        */
/*                                                                           */
/*                                                                           */
/*             Copyright (c)  Hitachi, Ltd. 1994.                            */
/*             Licensed Material of Hitachi, Ltd.                            */
/*                                                                           */
/*             SH7000 HI-SH7 DEBUGGER(HS0700IRCN1SMB) V1.0                   */
/*                                                                           */
/*                                                                           */
/*****************************************************************************/
/*****************************************************************************/
/*SPECIFICATIONS ;                                                           */
/*  FILE      = hdcommon.h ;                                                 */
/*  DATE      = 94/02/01 ;                                                   */
/*  AUTHOR    = Hitachi, Ltd. ;                                              */
/*  FUNCTION  = SH7000 HI-SH7 DEBUGGER common header file ;                  */
/*  ATTRIBUTE = PUBLIC ;                                                     */
/*  HISTORY   = V1.0 ;                                                       */
/*END OF SPECIFICATIONS ;                                                    */
/*****************************************************************************/
/*****************************************************************************/
/*                                                                           */
/*          Definition USE/NOTUSE value                                      */
/*                                                                           */
/*****************************************************************************/
#ifndef USE                                                  ←USE/NOTUSE定数の定義
#define USE             1
#endif
#ifndef NOTUSE
#define NOTUSE          0
#endif

/*****************************************************************************/
/*                                                                           */
/*          Definition debugger use in user system                           */
/*                                                                           */
/*          Usage   :   #define hd_dbg { USE | NOTUSE }                      */
/*                      USE     :   use the debugger                         */
/*                      NOTUSE  :   not use the debugger                     */
/*                                                                           */
/*****************************************************************************/
#define hd_dbg          USE                                    ←デバッガの使用／未使用の定義

#if     hd_dbg                          /* case of hd_dbg = USE             */

/*****************************************************************************/
/*                                                                           */
/*          Definition ID of debugger use                                    */
/*                                                                           */
/*****************************************************************************/
/*-------------------------------- task id ----------------------------------*/←デバッガが使用するタスクIDの定義
#define hd_dbgtskid     6               /* task id of debugger task         */
#define hd_cnstskid     7               /* task id of console task          */
                                        /*      range : [1...hi_maxtskid]   */

/*------------------------------ mail box id --------------------------------*/←デバッガが使用する
#define hd_dbgmbxid     4               /* mail box id of debugger task use */          メイルボックスIDの定義
#define hd_cnsmbxid     5               /* mail box id of console task use  */
                                        /*      range : [1...hi_maxmbxid]   */
```

図Ｂ－２　例題システムの共通ヘッダ(hdcommon.h)（１／２）

```
/************************************************************************/
/*                                                                      */
/*          Definition stack size of debugger, console task             */
/*                                                                      */
/************************************************************************/
#define DBGTSKSTKSZ     1496            /* stack size of debugger task      */←変更しないでください
#define CNSTSKSTKSZ      780            /* stack size of console task       */

/************************************************************************/
/*                                                                      */
/*          Definition parameter for HI-SH7 setup table                 */
/*                                                                      */
/************************************************************************/
#define ADDLOWINTNST    1               /* add value to hi_lowintnst        */←HI-SH7環境の補正値
#define ADDMAXTSKID     2               /* add value to hi_maxtskid         */
#define ADDINITSKNUM    2               /* add value to hi_initsknum        */
#define ADDMAXMBXID     2               /* add value to hi_maxmbxid         */

/************************************************************************/
/*                                                                      */
/*          Definition parameter for task, interrupt stack size         */
/*                                                                      */
/************************************************************************/
#define ADDTSKSTKSZ     60              /* add value to hi_tskstkszxx       */←変更しないでください
#define ADDINTSTKSZ     60              /* add value to hi_intstkszxx       */

#else                                   /* case of hd_dbg = NOTUSE          */
#define ADDLOWINTNST    0               /* not add to hi_lowintnst          */←変更しないでください
#define ADDMAXTSKID     0               /* not add to hi_maxtskid           */
#define ADDINITSKNUM    0               /* not add to hi_initsknum          */
#define ADDMAXMBXID     0               /* not add to hi_maxmbxid           */
#define ADDTSKSTKSZ     0               /* not add to hi_tskstkszxx         */
#define ADDINTSTKSZ     0               /* not add to hi_intstkszxx         */
#endif
```

図Ｂ－２　例題システムの共通ヘッダ(hdcommon.h)（２／２）

Ｂ．６　例題システムのデバッガセットアップテーブル

図Ｂ－３に例題システムのデバッガセットアップテーブル(hdsuptbl.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              SH7000 HI-SH7 DEBUGGER setup table ( Ver. 1.0 )         */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1994.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              SH7000 HI-SH7 DEBUGGER(HS0700IRCN1SMB) V1.0             */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hdsuptbl.c ;                                            */
/*  DATE      = 94/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = SH7000 HI-SH7 DEBUGGER setup table ;                    */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hdcommon.h"

#if         hd_dbg                      /* case of hd_dbg = USE        */

#include    "hdsuptbl.h"
/************************************************************************/
/*                                                                      */
/*          Definition function use in user system                      */
/*                                                                      */
/*          Usage   :   #define hd_xxxxxxxx { USE | NOTUSE }            */
/*                      USE     :   use the function (xxxxxxxx)         */
/*                      NOTUSE  :   not use the function (xxxxxxxx)     */
/*                                                                      */
/************************************************************************/
#define hd_message_func USE             /* message          function   */←メッセージ機能=使用
#define hd_multi_func   USE             /* multi-task       function   */←マルチタスク機能=使用
#define hd_memory_func  USE             /* memory           function   */←メモリ操作機能=使用
#define hd_execute_func USE             /* execute          function   */←実行制御機能=使用
#define hd_host_func    USE             /* host             function   */←ホスト機能=使用
```

図Ｂ－３　例題システムのデバッガセットアップテーブル(hdsuptbl.c)（１／４）

```
/*************************************************************************/
/*                                                                       */
/*          Definition command use in user system                        */
/*                                                                       */
/*          Usage   :   #define hd_xxxxxxxx { USE | NOTUSE }             */
/*                      USE     :   use the command (xxxxxxxx)           */
/*                      NOTUSE  :   not use the command (xxxxxxxx)       */
/*                                                                       */
/*************************************************************************/
/*---------------------------- message function ----------------------------*/←メッセージ機能
#if     hd_message_func                 /* case of hd_message_func = USE    */
#define hd_ihelp        USE             /* IHELP                 command    */←全コマンド=使用
#define hd_iid          USE             /* IID                   command    */
#endif

/*---------------------------- multi-task function ----------------------------*/←マルチタスク機能
#if     hd_multi_func                   /* case of hd_multi_func = USE      */
#define hd_can_wup      USE             /* CAN_WUP               command    */←全コマンド=使用
#define hd_chg_pri      USE             /* CHG_PRI               command    */
#define hd_clr_flg      USE             /* CLR_FLG               command    */
#define hd_cre_tsk      USE             /* CRE_TSK               command    */
#define hd_del_tsk      USE             /* DEL_TSK               command    */
#define hd_flg_sts      USE             /* FLG_STS               command    */
#define hd_get_blk      USE             /* GET_BLK               command    */
#define hd_get_tim      USE             /* GET_TIM               command    */
#define hd_istatus      USE             /* ISTATUS               command    */
#define hd_itrace       USE             /* ITRACE                command    */
#define hd_itrace_act   USE             /* ITRACE_ACTIVATE       command    */
#define hd_itrace_buf   USE             /* ITRACE_BUFFER         command    */
#define hd_itrace_sea   USE             /* ITRACE_SEARCH         command    */
#define hd_mbx_sts      USE             /* MBX_STS               command    */
#define hd_mpldef       USE             /* MPLDEF                command    */
#define hd_mpl_sts      USE             /* MPL_STS               command    */
#define hd_pol_flg      USE             /* POL_FLG               command    */
#define hd_rcv_msg      USE             /* RCV_MSG               command    */
#define hd_rel_blk      USE             /* REL_BLK               command    */
#define hd_rel_wai      USE             /* REL_WAI               command    */
#define hd_req_sem      USE             /* REQ_SEM               command    */
#define hd_rot_rdq      USE             /* ROT_RDQ               command    */
#define hd_rsm_tsk      USE             /* RSM_TSK               command    */
#define hd_run_tsk      USE             /* RUN_TSK               command    */
#define hd_sem_sts      USE             /* SEM_STS               command    */
#define hd_set_flg      USE             /* SET_FLG               command    */
#define hd_set_tim      USE             /* SET_TIM               command    */
#define hd_sig_sem      USE             /* SIG_SEM               command    */
#define hd_snd_msg      USE             /* SND_MSG               command    */
#define hd_sta_tsk      USE             /* STA_TSK               command    */
#define hd_stp_tsk      USE             /* STP_TSK               command    */
#define hd_sus_tsk      USE             /* SUS_TSK               command    */
#define hd_svcdef       USE             /* SVCDEF                command    */
#define hd_sysdef       USE             /* SYSDEF                command    */
#define hd_ter_tsk      USE             /* TER_TSK               command    */
#define hd_tskdef       USE             /* TSKDEF                command    */
#define hd_tsk_sts      USE             /* TSK_STS               command    */
#define hd_wup_tsk      USE             /* WUP_TSK               command    */
#endif

/*---------------------------- memory function ----------------------------*/←メモリ操作機能
#if     hd_memory_func                  /* case of hd_memory_func = USE     */
#define hd_assemble     USE             /* ASSEMBLE              command    */←全コマンド=使用
#define hd_disassemble  USE             /* DISASSEMBLE           command    */
#define hd_dump         USE             /* DUMP                  command    */
#define hd_fill         USE             /* FILL                  command    */
#define hd_memory       USE             /* MEMORY                command    */
#define hd_move         USE             /* MOVE                  command    */
#endif
```

図Ｂ－３　例題システムのデバッガセットアップテーブル(hdsuptbl.c)（２／４）

```
/*---------------------------- execute function ----------------------------*/←実行制御機能
#if     hd_execute_func                 /* case of hd_execute_func = USE    */
#define hd_regupd       USE             /* . (register update)  command     */←全コマンド=使用
#define hd_break        USE             /* BREAK                command     */
#define hd_break_ubc    USE             /* BREAK_UBC            command     */
#define hd_go           USE             /* GO                   command     */
#define hd_register     USE             /* REGISTER             command     */
#define hd_reset        USE             /* RESET                command     */
#define hd_step         USE             /* STEP                 command     */
#define hd_step_over    USE             /* STEP_OVER            command     */
#define hd_step_ubc     USE             /* STEP_UBC             command     */
#endif

/*---------------------------- host function ----------------------------*/←ホスト機能
#if     hd_host_func                    /* case of hd_host_func = USE       */
#define hd_load         USE             /* LOAD                 command     */←全コマンド=使用
#define hd_save         USE             /* SAVE                 command     */
#define hd_verify       USE             /* VERIFY               command     */
#endif

/*****************************************************************************/
/*                                                                           */
/*          Definition command information                                   */
/*                                                                           */
/*****************************************************************************/
#if     hd_execute_func                 /* case of hd_execute_func = USE    */
#if     hd_break                        /* case of hd_break = USE           */
#define hd_breaknum     32              /* number of break point            */←BREAKコマンド
                                        /*                 for BREAK command */          ブレークポイント数=32
                                        /*      range : [1...32]            */
#endif
#endif
```

図Ｂ－３　例題システムのデバッガセットアップテーブル(hdsuptbl.c)（３／４）

```
/*****************************************************************************/←オフラインデバッガの定義
/*                                                                           */
/*          Definition offline debugger start mode at CPU reset              */
/*                                                                           */
/*          Usage   :   #define hd_stadbgflg { TRUE | FALSE }                */
/*                      TRUE    :   start offline debugger at CPU reset      */
/*                      FALSE   :   not start offline debugger at CPU reset  */
/*                                                                           */
/*****************************************************************************/
#define hd_stadbgflg    TRUE                                                  ←リセット時のオフラインデバッガ=起動

/*****************************************************************************/←コンソールの定義
/*                                                                           */
/*          Definition environment of console                                */
/*                                                                           */
/*****************************************************************************/
#define hd_cnsclk       20000000        /* CPU clock = 20MHz                 */←CPUクロック=20MHz
#define hd_cnssci       1               /* SCI channel                       */←使用デバイス=SCI1
                                        /*          0 : SCI0                 */
                                        /*          1 : SCI1                 */
#define hd_cnsbpsno     7               /* bit rate number                   */←転送速度=9600bps
                                        /*          0 :   110 bps            */
                                        /*          1 :   150 bps            */
                                        /*          2 :   300 bps            */
                                        /*          3 :   600 bps            */
                                        /*          4 :  1200 bps            */
                                        /*          5 :  2400 bps            */
                                        /*          6 :  4800 bps            */
                                        /*          7 :  9600 bps            */
                                        /*          8 : 19200 bps            */
                                        /*          9 : 31250 bps            */
                                        /*         10 : 38400 bps            */
#define hd_cnsintlvl    14              /* interrupt level = 14              */←割込みレベル=14
                                        /*      range : [1...hi_knlmsklvl]   */

#include    "hdsuptbl.inc"

#endif                                  /* end of hd_dbg                     */
```

図Ｂ－３　例題システムのデバッガセットアップテーブル(hdsuptbl.c)（４／４）

Ｂ．７　例題システムのHI-SH7・ベクタテーブル

図Ｂ－４に例題システムのHI-SH7・ベクタテーブル(hivcttbl.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 vector table ( Ver. 1.0 )                        */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hivcttbl.c ;                                            */
/*  NAME      = hi_vcttbl ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 vector table ;                                   */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hdcommon.h"                                    ←共通ヘッダの読込み

#if     hd_dbg                          /* case of hd_dbg = USE         */←デバッガ用ハンドラ開始アドレスの参照
extern  _ODbrk_nmi();                   /* NMI handler                  */←NMI割込み
extern  _ODbrk_hbk();                   /* hardware break handler(UBC)  */←ハードウェアブレーク
extern  _ODbrk_sbk();                   /* software break handler(TRAPA #57)*/←ソフトウェアブレーク
extern  _ODerr_ill();                   /* illegal instruction handler  */←不当命令例外
extern  _ODerr_slt();                   /* slot illegal instruction handler */←スロット不当命令例外
extern  _ODerr_cpu();                   /* CPU bus error handler        */←CPUバスエラー例外
extern  _ODerr_dma();                   /* DMA bus error handler        */←DMAバスエラー例外
extern  _OHend_svc();                   /* debug SVC handler            */←HI-SH7・デバッグSVC
extern  _ODcnshdrer();                  /* receive error handler        */←コンソール送受信割込み
extern  _ODcnshdrrx();                  /* receive handler              */
extern  _ODcnshdrtx();                  /* trans handler                */
#endif

/*-------------- handler of MCU initialize routine for HI-SH7 --------------*/
extern  hi_mcuini();                    /* MCU initialize routine       */

/*---------------------- handler of kernel for HI-SH7 ----------------------*/
extern  _OHsys_clk();                   /* sys_clk   SVC handler        */
extern  _OHret_int();                   /* ret_int   SVC handler        */
extern  _OHenn_svc();                   /* interrupt SVC handler        */
extern  _OHent_svc();                   /* task      SVC handler        */

/*------------------- handler of timer driver for HI-SH7 -------------------*/
extern  hi_tmrhdr();                    /* hardware timer handler       */

/*------------------ handler of console driver for HI-SH7 ------------------*/←HI-SH7付属のコンソールドライバです
extern  hi_cnshdrer();                  /* receive error handler        */
extern  hi_cnshdrrx();                  /* receive handler              */
extern  hi_cnshdrtx();                  /* trans handler                */
```

図Ｂ－４　例題システムのHI-SH7・ベクタテーブル(hivcttbl.c)（１／５）

```
/*---------------------- handler of undefine interrupt ----------------------*/
extern  hi_intdwn04(),hi_intdwn05(),hi_intdwn06(),hi_intdwn07();
extern  hi_intdwn08(),hi_intdwn09(),hi_intdwn0A(),hi_intdwn0B();
extern  hi_intdwn0C(),hi_intdwn0D(),hi_intdwn0E(),hi_intdwn0F();
extern  hi_intdwn10(),hi_intdwn11(),hi_intdwn12(),hi_intdwn13();
extern  hi_intdwn14(),hi_intdwn15(),hi_intdwn16(),hi_intdwn17();
extern  hi_intdwn18(),hi_intdwn19(),hi_intdwn1A(),hi_intdwn1B();
extern  hi_intdwn1C(),hi_intdwn1D(),hi_intdwn1E(),hi_intdwn1F();
extern  hi_intdwn20(),hi_intdwn21(),hi_intdwn22(),hi_intdwn23();
extern  hi_intdwn24(),hi_intdwn25(),hi_intdwn26(),hi_intdwn27();
extern  hi_intdwn28(),hi_intdwn29(),hi_intdwn2A(),hi_intdwn2B();
extern  hi_intdwn2C(),hi_intdwn2D(),hi_intdwn2E(),hi_intdwn2F();
extern  hi_intdwn30(),hi_intdwn31(),hi_intdwn32(),hi_intdwn33();
extern  hi_intdwn34(),hi_intdwn35(),hi_intdwn36(),hi_intdwn37();
extern  hi_intdwn38(),hi_intdwn39(),hi_intdwn3A(),hi_intdwn3B();
extern  hi_intdwn3C(),hi_intdwn3D(),hi_intdwn3E(),hi_intdwn3F();
extern  hi_intdwn40(),hi_intdwn41(),hi_intdwn42(),hi_intdwn43();
extern  hi_intdwn44(),hi_intdwn45(),hi_intdwn46(),hi_intdwn47();
extern  hi_intdwn48(),hi_intdwn49(),hi_intdwn4A(),hi_intdwn4B();
extern  hi_intdwn4C(),hi_intdwn4D(),hi_intdwn4E(),hi_intdwn4F();
extern  hi_intdwn50(),hi_intdwn51(),hi_intdwn52(),hi_intdwn53();
extern  hi_intdwn54(),hi_intdwn55(),hi_intdwn56(),hi_intdwn57();
extern  hi_intdwn58(),hi_intdwn59(),hi_intdwn5A(),hi_intdwn5B();
extern  hi_intdwn5C(),hi_intdwn5D(),hi_intdwn5E(),hi_intdwn5F();
extern  hi_intdwn60(),hi_intdwn61(),hi_intdwn62(),hi_intdwn63();
extern  hi_intdwn64(),hi_intdwn65(),hi_intdwn66(),hi_intdwn67();
extern  hi_intdwn68(),hi_intdwn69(),hi_intdwn6A(),hi_intdwn6B();
extern  hi_intdwn6C(),hi_intdwn6D(),hi_intdwn6E(),hi_intdwn6F();
extern  hi_intdwn70(),hi_intdwn71(),hi_intdwn72(),hi_intdwn73();
extern  hi_intdwn74(),hi_intdwn75(),hi_intdwn76(),hi_intdwn77();
extern  hi_intdwn78(),hi_intdwn79(),hi_intdwn7A(),hi_intdwn7B();
extern  hi_intdwn7C(),hi_intdwn7D(),hi_intdwn7E(),hi_intdwn7F();
extern  hi_intdwn80(),hi_intdwn81(),hi_intdwn82(),hi_intdwn83();
extern  hi_intdwn84(),hi_intdwn85(),hi_intdwn86(),hi_intdwn87();
extern  hi_intdwn88(),hi_intdwn89(),hi_intdwn8A(),hi_intdwn8B();
extern  hi_intdwn8C(),hi_intdwn8D(),hi_intdwn8E(),hi_intdwn8F();
extern  hi_intdwn90(),hi_intdwn91(),hi_intdwn92(),hi_intdwn93();
extern  hi_intdwn94(),hi_intdwn95(),hi_intdwn96(),hi_intdwn97();
extern  hi_intdwn98(),hi_intdwn99(),hi_intdwn9A(),hi_intdwn9B();
extern  hi_intdwn9C(),hi_intdwn9D(),hi_intdwn9E(),hi_intdwn9F();
extern  hi_intdwnA0(),hi_intdwnA1(),hi_intdwnA2(),hi_intdwnA3();
extern  hi_intdwnA4(),hi_intdwnA5(),hi_intdwnA6(),hi_intdwnA7();
extern  hi_intdwnA8(),hi_intdwnA9(),hi_intdwnAA(),hi_intdwnAB();
extern  hi_intdwnAC(),hi_intdwnAD(),hi_intdwnAE(),hi_intdwnAF();
extern  hi_intdwnB0(),hi_intdwnB1(),hi_intdwnB2(),hi_intdwnB3();
extern  hi_intdwnB4(),hi_intdwnB5(),hi_intdwnB6(),hi_intdwnB7();
extern  hi_intdwnB8(),hi_intdwnB9(),hi_intdwnBA(),hi_intdwnBB();
extern  hi_intdwnBC(),hi_intdwnBD(),hi_intdwnBE(),hi_intdwnBF();
extern  hi_intdwnC0(),hi_intdwnC1(),hi_intdwnC2(),hi_intdwnC3();
extern  hi_intdwnC4(),hi_intdwnC5(),hi_intdwnC6(),hi_intdwnC7();
extern  hi_intdwnC8(),hi_intdwnC9(),hi_intdwnCA(),hi_intdwnCB();
extern  hi_intdwnCC(),hi_intdwnCD(),hi_intdwnCE(),hi_intdwnCF();
extern  hi_intdwnD0(),hi_intdwnD1(),hi_intdwnD2(),hi_intdwnD3();
extern  hi_intdwnD4(),hi_intdwnD5(),hi_intdwnD6(),hi_intdwnD7();
extern  hi_intdwnD8(),hi_intdwnD9(),hi_intdwnDA(),hi_intdwnDB();
extern  hi_intdwnDC(),hi_intdwnDD(),hi_intdwnDE(),hi_intdwnDF();
extern  hi_intdwnE0(),hi_intdwnE1(),hi_intdwnE2(),hi_intdwnE3();
extern  hi_intdwnE4(),hi_intdwnE5(),hi_intdwnE6(),hi_intdwnE7();
extern  hi_intdwnE8(),hi_intdwnE9(),hi_intdwnEA(),hi_intdwnEB();
extern  hi_intdwnEC(),hi_intdwnED(),hi_intdwnEE(),hi_intdwnEF();
extern  hi_intdwnF0(),hi_intdwnF1(),hi_intdwnF2(),hi_intdwnF3();
extern  hi_intdwnF4(),hi_intdwnF5(),hi_intdwnF6(),hi_intdwnF7();
extern  hi_intdwnF8(),hi_intdwnF9(),hi_intdwnFA(),hi_intdwnFB();
extern  hi_intdwnFC(),hi_intdwnFD(),hi_intdwnFE(),hi_intdwnFF();
```

図Ｂ－４　例題システムのHI-SH7・ベクタテーブル(hivcttbl.c)（２／５）

```
/************************************************************************/
/*                                                                      */
/*          Definition vector table                                     */
/*                                                                      */
/************************************************************************/
const   VP  hi_vcttbl[] = {
            (VP)hi_mcuini,  /* power on reset                (vctno = 00) */
            (VP)0x10000000, /* SP value at power on reset                */
            (VP)hi_mcuini,  /* manual reset                  (vctno = 02) */
            (VP)0x10000000, /* SP value at manual reset                  */
#if     hd_dbg                          /* case of hd_dbg = USE          */
            (VP)_ODerr_ill, /* illegal instruction          (vctno = 04) */←不当命令例外ハンドラの登録
            (VP)hi_intdwn05,/* reserve                      (vctno = 05) */
            (VP)_ODerr_slt, /* slot illegal instruction     (vctno = 06) */←スロット不当命令例外ハンドラの登録
            (VP)hi_intdwn07,/* reserve                      (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                      (vctno = 08) */
            (VP)_ODerr_cpu, /* CPU bus error                (vctno = 09) */←CPUバスエラー例外ハンドラの登録
            (VP)_ODerr_dma, /* DMA bus error                (vctno = 0A) */←DMAバスエラー例外ハンドラの登録
            (VP)_ODbrk_nmi, /* NMI                          (vctno = 0B) */←NMI割込みハンドラの登録
            (VP)_ODbrk_hbk, /* user break                    (vctno = 0C) */←ハードウェアブレークハンドラの登録
#else                                   /* case of hd_dbg = NOTUSE       */
            (VP)hi_intdwn04,/* illegal instruction          (vctno = 04) */
            (VP)hi_intdwn05,/* reserve                      (vctno = 05) */
            (VP)hi_intdwn06,/* slot illegal instruction     (vctno = 06) */
            (VP)hi_intdwn07,/* reserve                      (vctno = 07) */
            (VP)hi_intdwn08,/* reserve                      (vctno = 08) */
            (VP)hi_intdwn09,/* CPU bus error                (vctno = 09) */
            (VP)hi_intdwn0A,/* DMA bus error                (vctno = 0A) */
            (VP)hi_intdwn0B,/* NMI                          (vctno = 0B) */
            (VP)hi_intdwn0C,/* user break                   (vctno = 0C) */
#endif
            (VP)hi_intdwn0D,/* reserve                      (vctno = 0D) */
            (VP)hi_intdwn0E,/* reserve                      (vctno = 0E) */
            (VP)hi_intdwn0F,/* reserve                      (vctno = 0F) */
            (VP)hi_intdwn10,/* reserve                      (vctno = 10) */
            (VP)hi_intdwn11,/* reserve                      (vctno = 11) */
            (VP)hi_intdwn12,/* reserve                      (vctno = 12) */
            (VP)hi_intdwn13,/* reserve                      (vctno = 13) */
            (VP)hi_intdwn14,/* reserve                      (vctno = 14) */
            (VP)hi_intdwn15,/* reserve                      (vctno = 15) */
            (VP)hi_intdwn16,/* reserve                      (vctno = 16) */
            (VP)hi_intdwn17,/* reserve                      (vctno = 17) */
            (VP)hi_intdwn18,/* reserve                      (vctno = 18) */
            (VP)hi_intdwn19,/* reserve                      (vctno = 19) */
            (VP)hi_intdwn1A,/* reserve                      (vctno = 1A) */
            (VP)hi_intdwn1B,/* reserve                      (vctno = 1B) */
            (VP)hi_intdwn1C,/* reserve                      (vctno = 1C) */
            (VP)hi_intdwn1D,/* reserve                      (vctno = 1D) */
            (VP)hi_intdwn1E,/* reserve                      (vctno = 1E) */
            (VP)hi_intdwn1F,/* reserve                      (vctno = 1F) */
            (VP)hi_intdwn20,/* trapa #32                    (vctno = 20) */
            (VP)hi_intdwn21,/* trapa #33                    (vctno = 21) */
            (VP)hi_intdwn22,/* trapa #34                    (vctno = 22) */
            (VP)hi_intdwn23,/* trapa #35                    (vctno = 23) */
            (VP)hi_intdwn24,/* trapa #36                    (vctno = 24) */
            (VP)hi_intdwn25,/* trapa #37                    (vctno = 25) */
            (VP)hi_intdwn26,/* trapa #38                    (vctno = 26) */
            (VP)hi_intdwn27,/* trapa #39                    (vctno = 27) */
            (VP)hi_intdwn28,/* trapa #40                    (vctno = 28) */
            (VP)hi_intdwn29,/* trapa #41                    (vctno = 29) */
            (VP)hi_intdwn2A,/* trapa #42                    (vctno = 2A) */
            (VP)hi_intdwn2B,/* trapa #43                    (vctno = 2B) */
            (VP)hi_intdwn2C,/* trapa #44                    (vctno = 2C) */
            (VP)hi_intdwn2D,/* trapa #45                    (vctno = 2D) */
            (VP)hi_intdwn2E,/* trapa #46                    (vctno = 2E) */
            (VP)hi_intdwn2F,/* trapa #47                    (vctno = 2F) */
            (VP)hi_intdwn30,/* trapa #48                    (vctno = 30) */
            (VP)hi_intdwn31,/* trapa #49                    (vctno = 31) */
```

図Ｂ－４　例題システムのHI-SH7・ベクタテーブル(hivcttbl.c)（３／５）

```
        (VP)hi_intdwn32,/* trapa #50                        (vctno = 32) */
        (VP)hi_intdwn33,/* trapa #51                        (vctno = 33) */
        (VP)hi_intdwn34,/* trapa #52                        (vctno = 34) */
        (VP)hi_intdwn35,/* trapa #53                        (vctno = 35) */
        (VP)hi_intdwn36,/* trapa #54                        (vctno = 36) */
        (VP)hi_intdwn37,/* trapa #55                        (vctno = 37) */
        (VP)hi_intdwn38,/* trapa #56                        (vctno = 38) */
#if     hd_dbg                          /* case of hd_dbg = USE          */
        (VP)_0Dbrk_sbk, /* trapa #57                        (vctno = 39) */←ソフトウェアブレークハンドラの登録
        (VP)hi_intdwn3A,/* trapa #58                        (vctno = 3A) */
        (VP)_0Hend_svc, /* trapa #59 (reserve HI-SH7)       (vctno = 3B) */←HI-SH7・デバッグSVCハンドラの登録
#else                                   /* case of hd_dbg = NOTUSE       */
        (VP)hi_intdwn39,/* trapa #57                        (vctno = 39) */
        (VP)hi_intdwn3A,/* trapa #58                        (vctno = 3A) */
        (VP)hi_intdwn3B,/* trapa #59 (reserve HI-SH7)       (vctno = 3B) */
#endif
        (VP)_0Hsys_clk, /* trapa #60 (use HI-SH7)           (vctno = 3C) */
        (VP)_0Hret_int, /* trapa #61 (use HI-SH7)           (vctno = 3D) */
        (VP)_0Henn_svc, /* trapa #62 (use HI-SH7)           (vctno = 3E) */
        (VP)_0Hent_svc, /* trapa #63 (use HI-SH7)           (vctno = 3F) */
        (VP)hi_intdwn40,/* IRQ0                             (vctno = 40) */
        (VP)hi_intdwn41,/* IRQ1                             (vctno = 41) */
        (VP)hi_intdwn42,/* IRQ2                             (vctno = 42) */
        (VP)hi_intdwn43,/* IRQ3                             (vctno = 43) */
        (VP)hi_intdwn44,/* IRQ4                             (vctno = 44) */
        (VP)hi_intdwn45,/* IRQ5                             (vctno = 45) */
        (VP)hi_intdwn46,/* IRQ6                             (vctno = 46) */
        (VP)hi_intdwn47,/* IRQ7                             (vctno = 47) */
        (VP)hi_intdwn48,/* DMAC0 DEI0                       (vctno = 48) */
        (VP)hi_intdwn49,/* reserve                          (vctno = 49) */
        (VP)hi_intdwn4A,/* DMAC1 DEI1                       (vctno = 4A) */
        (VP)hi_intdwn4B,/* reserve                          (vctno = 4B) */
        (VP)hi_intdwn4C,/* DMAC2 DEI2                       (vctno = 4C) */
        (VP)hi_intdwn4D,/* reserve                          (vctno = 4D) */
        (VP)hi_intdwn4E,/* DMAC3 DEI3                       (vctno = 4E) */
        (VP)hi_intdwn4F,/* reserve                          (vctno = 4F) */
        (VP)hi_tmrhdr,  /* ITU0 IMIA0 (use timer driver)    (vctno = 50) */
        (VP)hi_intdwn51,/* ITU0 IMIB0                       (vctno = 51) */
        (VP)hi_intdwn52,/* ITU0 OVI0                        (vctno = 52) */
        (VP)hi_intdwn53,/* reserve                          (vctno = 53) */
        (VP)hi_intdwn54,/* ITU1 IMIA1                       (vctno = 54) */
        (VP)hi_intdwn55,/* ITU1 IMIB1                       (vctno = 55) */
        (VP)hi_intdwn56,/* ITU1 OVI1                        (vctno = 56) */
        (VP)hi_intdwn57,/* reserve                          (vctno = 57) */
        (VP)hi_intdwn58,/* ITU2 IMIA2                       (vctno = 58) */
        (VP)hi_intdwn59,/* ITU2 IMIB2                       (vctno = 59) */
        (VP)hi_intdwn5A,/* ITU2 OVI2                        (vctno = 5A) */
        (VP)hi_intdwn5B,/* reserve                          (vctno = 5B) */
        (VP)hi_intdwn5C,/* ITU3 IMIA3                       (vctno = 5C) */
        (VP)hi_intdwn5D,/* ITU3 IMIB3                       (vctno = 5D) */
        (VP)hi_intdwn5E,/* ITU3 OVI3                        (vctno = 5E) */
        (VP)hi_intdwn5F,/* reserve                          (vctno = 5F) */
        (VP)hi_intdwn60,/* ITU4 IMIA4                       (vctno = 60) */
        (VP)hi_intdwn61,/* ITU4 IMIB4                       (vctno = 61) */
        (VP)hi_intdwn62,/* ITU4 OVI4                        (vctno = 62) */
        (VP)hi_intdwn63,/* reserve                          (vctno = 63) */
```

図Ｂ－４　例題システムのHI-SH7・ベクタテーブル(hivcttbl.c)（４／５）

```
        (VP)hi_cnshdrer,/* SCI0 ERI0 (use console driver)  (vctno = 64) */←HI-SH7付属のコンソールドライバです
        (VP)hi_cnshdrrx,/* SCI0 RXI0 (use console driver)  (vctno = 65) */
        (VP)hi_cnshdrtx,/* SCI0 TXI0 (use console driver)  (vctno = 66) */
        (VP)hi_intdwn67,/* SCI0 TEI0                      (vctno = 67) */
#if     hd_dbg                          /* case of hd_dbg = USE          */
        (VP)_0Dcnshdrer,/* SCI1 ERI1                      (vctno = 68) */←コンソール送受信割込みハンドラの登録
        (VP)_0Dcnshdrrx,/* SCI1 RXI1                      (vctno = 69) */
        (VP)_0Dcnshdrtx,/* SCI1 TXI1                      (vctno = 6A) */
#else                                   /* case of hd_dbg = NOTUSE       */
        (VP)hi_intdwn68,/* SCI1 ERI1                      (vctno = 68) */
        (VP)hi_intdwn69,/* SCI1 RXI1                      (vctno = 69) */
        (VP)hi_intdwn6A,/* SCI1 TXI1                      (vctno = 6A) */
#endif
        (VP)hi_intdwn6B,/* SCI1 TEI1                      (vctno = 6B) */
        (VP)hi_intdwn6C,/* PRT PEI                        (vctno = 6C) */
        (VP)hi_intdwn6D,/* A/D ADI                        (vctno = 6D) */
        (VP)hi_intdwn6E,/* reserve                        (vctno = 6E) */
        (VP)hi_intdwn6F,/* reserve                        (vctno = 6F) */
        (VP)hi_intdwn70,/* WDT ITI                        (vctno = 70) */
        (VP)hi_intdwn71,/* REF CMI                        (vctno = 71) */
        (VP)hi_intdwn72,/* reserve                        (vctno = 72) */
        (VP)hi_intdwn73,/* reserve                        (vctno = 73) */
        (VP)hi_intdwn74,/* reserve                        (vctno = 74) */
        (VP)hi_intdwn75,/* reserve                        (vctno = 75) */
        (VP)hi_intdwn76,/* reserve                        (vctno = 76) */
        (VP)hi_intdwn77,/* reserve                        (vctno = 77) */
        (VP)hi_intdwn78,/* reserve                        (vctno = 78) */
        (VP)hi_intdwn79,/* reserve                        (vctno = 79) */
        (VP)hi_intdwn7A,/* reserve                        (vctno = 7A) */
        (VP)hi_intdwn7B,/* reserve                        (vctno = 7B) */
        (VP)hi_intdwn7C,/* reserve                        (vctno = 7C) */
        (VP)hi_intdwn7D,/* reserve                        (vctno = 7D) */
        (VP)hi_intdwn7E,/* reserve                        (vctno = 7E) */
        (VP)hi_intdwn7F,/* reserve                        (vctno = 7F) */
        (VP)hi_intdwn80,/* reserve                        (vctno = 80) */
        (VP)hi_intdwn81,/* reserve                        (vctno = 81) */
        (VP)hi_intdwn82,/* reserve                        (vctno = 82) */
        (VP)hi_intdwn83,/* reserve                        (vctno = 83) */
        (VP)hi_intdwn84,/* reserve                        (vctno = 84) */
        (VP)hi_intdwn85,/* reserve                        (vctno = 85) */
        (VP)hi_intdwn86,/* reserve                        (vctno = 86) */
        (VP)hi_intdwn87,/* reserve                        (vctno = 87) */
        (VP)hi_intdwn88,/* reserve                        (vctno = 88) */
        (VP)hi_intdwn89,/* reserve                        (vctno = 89) */
        (VP)hi_intdwn8A,/* reserve                        (vctno = 8A) */
        (VP)hi_intdwn8B,/* reserve                        (vctno = 8B) */
        (VP)hi_intdwn8C,/* reserve                        (vctno = 8C) */
        (VP)hi_intdwn8D,/* reserve                        (vctno = 8D) */
        (VP)hi_intdwn8E,/* reserve                        (vctno = 8E) */
        (VP)hi_intdwn8F,/* reserve                        (vctno = 8F) */
        (VP)hi_intdwn90,/* reserve                        (vctno = 90) */
        (VP)hi_intdwn91,/* reserve                        (vctno = 91) */
        (VP)hi_intdwn92,/* reserve                        (vctno = 92) */
        (VP)hi_intdwn93,/* reserve                        (vctno = 93) */
        (VP)hi_intdwn94,/* reserve                        (vctno = 94) */
        (VP)hi_intdwn95,/* reserve                        (vctno = 95) */
        (VP)hi_intdwn96,/* reserve                        (vctno = 96) */
        (VP)hi_intdwn97,/* reserve                        (vctno = 97) */

                                中略


        (VP)hi_intdwnFF,/* reserve                        (vctno = FF) */
    };
```

図Ｂ－４　例題システムのHI-SH7・ベクタテーブル(hivcttbl.c)（５／５）

Ｂ．８　例題システムのHI-SH7・セットアップテーブル

図Ｂ－５に例題システムのHI-SH7・セットアップテーブル(hisuptbl.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 setup table ( Ver. 1.0 )                         */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hisuptbl.c ;                                            */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 setup table ;                                    */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hisuptbl.h"
#include    "hdcommon.h"                                    ←共通ヘッダの読込み

/************************************************************************/
/*                                                                      */
/*          Definition setup table check function use in user system    */
/*                                                                      */
/*          Usage   :   #define hi_chksut { USE | NOTUSE }              */
/*                      USE     :   use the setup table check function  */
/*                      NOTUSE  :   not use the setup table check function  */
/*                                                                      */
/************************************************************************/
#define hi_chksut       USE             /* setup table check function   */
```

図Ｂ－５　例題システムのHI-SH7・セットアップテーブル(hisuptbl.c)（１／４）

```
/************************************************************************/
/*                                                                      */
/*          Definition SVC use in user system                           */
/*                                                                      */
/*          Usage   :   #define hi_xyyy_zzz { USE | NOTUSE }            */
/*                      USE     :   use the SVC (xyyy_zzz)              */
/*                      NOTUSE  :   not use the SVC (xyyy_zzz)          */
/*                                                                      */
/************************************************************************/
/*---------------------- task management function ----------------------*/
#define hi_cre_tsk      USE             /* cre_tsk  SVC                 */
#define hi_sta_tsk      USE             /* sta_tsk  SVC                 */
#define hi_ista_tsk     USE             /* ista_tsk SVC                 */
#define hi_del_tsk      USE             /* del_tsk  SVC                 */
#define hi_ext_tsk      USE             /* ext_tsk  SVC                 */
#define hi_exd_tsk      USE             /* exd_tsk  SVC                 */
#define hi_ter_tsk      USE             /* ter_tsk  SVC                 */
#define hi_chg_pri      USE             /* chg_pri  SVC                 */
#define hi_ichg_pri     USE             /* ichg_pri SVC                 */
#define hi_rot_rdq      USE             /* rot_rdq  SVC                 */
#define hi_irot_rdq     USE             /* irot_rdq SVC                 */
#define hi_rel_wai      USE             /* rel_wai  SVC                 */
#define hi_irel_wai     USE             /* irel_wai SVC                 */
#define hi_get_tid      USE             /* get_tid  SVC                 */
#define hi_tsk_sts      USE             /* tsk_sts  SVC                 */

/*--------------- task synchronization management function -----------------*/
#define hi_sus_tsk      USE             /* sus_tsk  SVC                 */
#define hi_isus_tsk     USE             /* isus_tsk SVC                 */
#define hi_rsm_tsk      USE             /* rsm_tsk  SVC                 */
#define hi_irsm_tsk     USE             /* irsm_tsk SVC                 */
#define hi_slp_tsk      USE             /* slp_tsk  SVC                 */
#define hi_wai_tsk      USE             /* wai_tsk  SVC                 */
#define hi_wup_tsk      USE             /* wup_tsk  SVC                 */
#define hi_iwup_tsk     USE             /* iwup_tsk SVC                 */
#define hi_can_wup      USE             /* can_wup  SVC                 */

/*-------------- synchronization and communication function ----------------*/
#define hi_set_flg      USE             /* set_flg  SVC                 */
#define hi_iset_flg     USE             /* iset_flg SVC                 */
#define hi_clr_flg      USE             /* clr_flg  SVC                 */
#define hi_wai_flg      USE             /* wai_flg  SVC                 */
#define hi_pol_flg      USE             /* pol_flg  SVC                 */
#define hi_flg_sts      USE             /* flg_sts  SVC                 */
#define hi_sig_sem      USE             /* sig_sem  SVC                 */
#define hi_isig_sem     USE             /* isig_sem SVC                 */
#define hi_wai_sem      USE             /* wai_sem  SVC                 */
#define hi_preq_sem     USE             /* preq_sem SVC                 */
#define hi_sem_sts      USE             /* sem_sts  SVC                 */
#define hi_snd_msg      USE             /* snd_msg  SVC                 */
#define hi_isnd_msg     USE             /* isnd_msg SVC                 */
#define hi_rcv_msg      USE             /* rcv_msg  SVC                 */
#define hi_prcv_msg     USE             /* prcv_msg SVC                 */
#define hi_mbx_sts      USE             /* mbx_sts  SVC                 */
```

中略

```
/*---------------------- system management function ----------------------*/
#define hi_get_ver      USE             /* get_ver  SVC                 */
```

図Ｂ－５　例題システムのHI-SH7・セットアップテーブル(hisuptbl.c)（２／４）

```
/*************************************************************************/
/*                                                                       */
/*          Definition kernel information                                */
/*                                                                       */
/*************************************************************************/
#define hi_knlmsklvl    14              /* kernel mask level of interrupt   */
                                        /*      range : [1...15]            */
#define hi_uppintnst    0               /* nest number of upper interrupt   */
                                        /*            than kernel mask level*/
                                        /*      range : [0...]              */
#define hi_lowintnst    (2 + ADDLOWINTNST)                          ←割込みネスト数（≦カーネルマスクレベル）の補正
                                        /* nest number of lower interrupt   */
                                        /*            than kernel mask level*/
                                        /*      range : [0...15]            */

/*************************************************************************/
/*                                                                       */
/*          Definition task information                                  */
/*                                                                       */
/*************************************************************************/
#define hi_maxtskid     (5 + ADDMAXTSKID)                                 ←最大タスクIDの補正
                                        /* max task id                      */
                                        /*      range : [0...1023]          */
#define hi_maxtskpri    5               /* max task priority                */
                                        /*      range : [1...255]           */
#define hi_initsknum    (1 + ADDINITSKNUM)                                ←初期登録タスク数の補正
                                        /* number of initial task           */
                                        /*      range : [0...1023]          */

/*************************************************************************/
/*                                                                       */
/*          Definition task information to setup table (INITSP, INITSK)  */
/*                                                                       */
/*************************************************************************/
/*-------------------- define task stack end address ---------------------*/
#if     hi_maxtskid                     /* case of hi_maxtskid > 0          */
#include        "hitskstk.h"            /* include "hitskstk.h"             */
extern  VW      hi_tskstk1[];           /* task stack of tskid = 1          */
extern  VW      hi_tskstk2[];           /* task stack of tskid = 2          */
extern  VW      hi_tskstk3[];           /* task stack of tskid = 3          */
extern  VW      hi_tskstk4[];           /* task stack of tskid = 4, 5       */
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  VW      _0Ddbgtskstk[];         /* task stack of debugger task      */←デバッガタスクのスタック参照
extern  VW      _0Dcnstskstk[];         /* task stack of console task       */←コンソールタスクのスタック参照
#endif

const   INITSP  _0Hinitsp[hi_maxtskid] = {
        /* define task stack end address    */
            (VW *)&hi_tskstk1[(hi_tskstksz1) / sizeof(VW)], /* tskid = 1    */
            (VW *)&hi_tskstk2[(hi_tskstksz2) / sizeof(VW)], /* tskid = 2    */
            (VW *)&hi_tskstk3[(hi_tskstksz3) / sizeof(VW)], /* tskid = 3    */
            (VW *)&hi_tskstk4[(hi_tskstksz4) / sizeof(VW)], /* tskid = 4    */
            (VW *)&hi_tskstk4[(hi_tskstksz4) / sizeof(VW)], /* tskid = 5    */
#if     hd_dbg                          /* case of hd_dbg = USE             */
            (VW *)&_0Ddbgtskstk[(DBGTSKSTKSZ) / sizeof(VW)],/* debugger task*/←デバッガタスクのスタック定義
            (VW *)&_0Dcnstskstk[(CNSTSKSTKSZ) / sizeof(VW)],/* console task */←コンソールタスクのスタック定義
#endif
        };
#else                                   /* case of hi_maxtskid = 0          */
#define _0Hinitsp       NADR            /* not define task stack            */
#endif
```

図Ｂ－５　例題システムのHI-SH7・セットアップテーブル(hisuptbl.c)（３／４）

```
/*-------------------- define initial task information ---------------------*/
#if     hi_initsknum                    /* case of hi_initsknum > 0         */
extern  TASK    hi_cnsdrv();            /* console driver task              */←HI-SH7付属のコンソールドライバです
#if     hd_dbg                          /* case of hd_dbg = USE             */
extern  TASK       _ODdbgtsk();         /* debugger task                    */←デバッガタスクの参照
extern  TASK       _ODcnstsk();         /* console task                     */←コンソールタスクの参照
#endif

const   INITSK  _OHinitsk[hi_initsknum] = {
        /* define console driver task   */                                  ←HI-SH7付属のコンソールドライバです
            (ID)1,                      /* task id              = 1         */
            (TPRI)2,                    /* initial task priority = 2        */
            (TASKP)hi_cnsdrv,           /* task start address    = hi_cnsdrv*/
#if     hd_dbg                          /* case of hd_dbg = USE             */
            (ID)hd_dbgtskid,            /* task id            = hd_dbgtskid*/←デバッガタスクの初期登録
            (TPRI)1,                    /* initial task priority = 1        */
            (TASKP)_ODdbgtsk,           /* task start address    = _ODdbgtsk*/
            (ID)hd_cnstskid,            /* task id            = hd_cnstskid*/←コンソールタスクの初期登録
            (TPRI)1,                    /* initial task priority = 1        */
            (TASKP)_ODcnstsk,           /* task start address    = _ODcnstsk*/
#endif
        };
#else                                   /* case of hi_initsknum = 0         */
#define _OHinitsk       NADR            /* not define initial task          */
#endif

/*************************************************************************/
/*                                                                       */
/*          Definition synchronization and communication object information */
/*                                                                       */
/*************************************************************************/
#define hi_maxflgid     3               /* max event flag id                */
                                        /*      range : [0...1023]          */
#define hi_maxsemid     3               /* max semaphore id                 */
                                        /*      range : [0...1023]          */
#define hi_maxmbxid     (3 + ADDMAXMBXID)                                   ←最大メイルボックスIDの補正
                                        /* max mail box id                  */
                                        /*      range : [0...1023]          */


                                    中略


#if         hd_dbg                      /* case of hd_dbg = USE             */
#include    "hisuptbl.dbg"                                                  ←HI-SH7・デバッガ組込み用インクルードファイルの読込み
#else                                   /* case of hd_dbg = NOTUSE          */
#include    "hisuptbl.inc"
#endif
```

図Ｂ－５　例題システムのHI-SH7・セットアップテーブル(hisuptbl.c)（４／４）

Ｂ．９　例題システムのHI-SH7・MCU初期化ルーチン

図Ｂ－６に例題システムのHI-SH7・MCU初期化ルーチン(himcuini.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 MCU initialize routine ( Ver. 1.0 )              */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = himcuini.c ;                                            */
/*  NAME      = hi_mcuini ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 MCU initialize routine ;                         */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    <machine.h>
#include    "itron.h"
#include    "hdcommon.h"                                    ←共通ヘッダの読込み

#define VCTBASE           0x00000000    /* vector base address = 0x00000000 */
#define RAMSTA      (VW *)0x0fffe000    /* RAM start address   = 0x0fffe000 */
#define RAMEND      (VW *)0x0fffffff    /* RAM end   address   = 0x0fffffff */

#if     hd_dbg                          /* case of hd_dbg = USE             */
#define RAMSTA2     (VW *)0x01000000    /* RAM2 start address  = 0x01000000 */←外部RAMクリア領域の設定
#define RAMEND2     (VW *)0x01007fff    /* RAM2 end   address  = 0x01007fff */

extern  void    _0Drs__sbk(void);       /* software break reset routine     */←ソフトウェアブレーク解除処理の参照
extern  void    _0Drs__dbg(void);       /* debugger reset routine           */←デバッガリセットルーチンの参照

static  UW      hot_start_flag;         /* hot start flag                   */←ホット／コールドスタートフラグ
                                        /*      TRUE(1)     :   hot start   */
                                        /*      other       :   cold start  */
#endif

extern  void    _0Hrs__knl(void);       /* kernel reset routine             */
```

図Ｂ－６　例題システムのHI-SH7・MCU初期化ルーチン(himcuini.c)（１／２）

```
void    hi_mcuini(void)
{
    register    VW      *p;             /* pointer to memory                */

/*  set_vbr((VP *)VCTBASE); */          /* set vector base address to VBR   */

#if     hd_dbg                          /* case of hd_dbg = USE             */
    if(hot_start_flag == TRUE)          /* if hot_start_flag is TRUE        */←ホット／コールドスタート判定
    {
        _ODrs__sbk();                   /* call software break reset routine*/←ソフトウェアブレーク解除処理の呼出し
        goto hot_start;                 /* goto hot_start                   */
    }
#endif

    for(p = RAMSTA; p <= RAMEND; p++)   /* RAM clear                        */
        *p = 0x00000000;

#if     hd_dbg                          /* case of hd_dbg = USE             */
    for(p = RAMSTA2; p <= RAMEND2; p++) /* RAM2 clear                       */←外部RAMクリア
        *p = 0x00000000;

    hot_start_flag = TRUE;              /* set TRUE to hot_start_flag       */←ホット／コールドスタートフラグ設定

hot_start:
    _ODrs__dbg();                       /* call debugger reset routine      */←デバッガリセットルーチンの呼出し
#endif

    _OHrs__knl();                       /* go to kernel reset routine       */
}
```

図Ｂ－６　例題システムのHI-SH7・MCU初期化ルーチン(himcuini.c)（２／２）

Ｂ．１０　例題システムのHI-SH7・システム初期化ハンドラ

図Ｂ－７に例題システムのHI-SH7・システム初期化ハンドラ(hisysini.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 system initial handler ( Ver. 1.0 )              */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hisysini.c ;                                            */
/*  NAME      = hi_sysini ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 system initial handler ;                         */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    "itron.h"
#include    "hdcommon.h"                                    ←共通ヘッダの読込み

extern  void    hi_tmrini(void);        /* timer initialize routine     */
extern  void    hi_sysdwn(W, ER, UW);   /* system down routine          */

INIHDR  hi_sysini(void)
{
    ER      ercd;                       /* error code                   */

#if     hd_dbg                          /* case of hd_dbg = USE         */
    ercd = ista_tsk(hd_dbgtskid);       /* start debugger task          */←デバッガタスクの起動
    if(ercd < E_OK)                     /* if ista_tsk fail             */
        hi_sysdwn((W)2, ercd, (UW)0);   /* go to system down routine    */

    ercd = ista_tsk(hd_cnstskid);       /* start console task           */←コンソールタスクの起動
    if(ercd < E_OK)                     /* if ista_tsk fail             */
        hi_sysdwn((W)3, ercd, (UW)0);   /* go to system down routine    */
#endif

    hi_tmrini();                        /* call timer initialize routine*/

    ercd = ista_tsk((ID)1);             /* start task (tskid = 1)       */
    if(ercd < E_OK)                     /* if ista_tsk fail             */
        hi_sysdwn((W)1, ercd, (UW)0);   /* go to system down routine    */
}
```

図Ｂ－７　例題システムのHI-SH7・システム初期化ハンドラ(hisysini.c)

Ｂ．１１　例題システムのHI-SH7・未定義割込み異常処理ルーチン

図Ｂ－８に例題システムのHI-SH7・未定義割込み異常処理ルーチン(hiintdwn.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 interrupt down routine ( Ver. 1.0 )              */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hiintdwn.c ;                                            */
/*  NAME      = hi_intdwn ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 interrupt down routine (undefine interrupt handler) ;*/
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    <machine.h>
#include    "itron.h"
#include    "hdcommon.h"                                    ←共通ヘッダの読込み

#if     hd_dbg                          /* case of hd_dbg = USE         */
extern  void    _ODintdwn(UB);          /* intdwn. routine of debugger  */←デバッガ未定義割込みルーチン参照
#endif

void    hi_intdwn(vctno)
UB      vctno;                          /* interrupt vector number      */
{
    set_imask(15);                      /* mask all interrupt           */

#if     hd_dbg                          /* case of hd_dbg = USE         */
    _ODintdwn(vctno);                   /* call intdwn. routine of debugger */←デバッガ未定義割込みルーチン呼出し
#endif

    while(TRUE);                        /* endless loop                 */
}
```

図Ｂ－８　例題システムのHI-SH7・未定義割込み異常処理ルーチン(hiintdwn.c)

Ｂ．１２　例題システムのHI-SH7・システム異常終了処理ルーチン

図Ｂ－９に例題システムのHI-SH7・システム異常終了処理ルーチン(hisysdwn.c)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 system down routine ( Ver. 1.0 )                 */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hisysdwn.c ;                                            */
/*  NAME      = hi_sysdwn ;                                             */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 system down routine ;                            */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#include    <machine.h>
#include    "itron.h"
#include    "hdcommon.h"                                    ←共通ヘッダの読込み

#if     hd_dbg                          /* case of hd_dbg = USE         */
extern  void    _0Dsysdwn(W, ER, UW);   /* sysdwn. routine of debugger  */←デバッガシステム異常終了ルーチン参照
#endif

void    hi_sysdwn(type, ercd, inf)
W       type;               /* type of system down                      */
                            /*      type >=  1 : system down of user program*/
                            /*      type ==  0 : setup table error      */
                            /*      type == -1 : context error of ext_tsk   */
                            /*      type == -2 : context error of exd_tsk   */
                            /*      type == -3 : context error of ret_int   */
                            /*      type == -4 : context error of sys_clk   */
                            /*      type <= -5 : system reserve         */
ER      ercd;               /* error code of system down                */
                            /*      type ==  0 : error code of setup table  */
                            /*      type == -1 : error code of ext_tsk      */
                            /*      type == -2 : error code of exd_tsk      */
                            /*      type == -3 : error code of ret_int      */
                            /*      type == -4 : error code of sys_clk      */
UW      inf;                /* information of system down               */
                            /*      type == -1 : address of ext_tsk call    */
                            /*      type == -2 : address of exd_tsk call    */
                            /*      type == -3 : address of ret_int call    */
                            /*      type == -4 : address of sys_clk call    */
{
    set_imask(15);          /* mask all interrupt                       */

#if     hd_dbg                          /* case of hd_dbg = USE         */
    _0Dsysdwn(type, ercd, inf);         /* call sysdwn. routine of debugger */←デバッガシステム異常終了ルーチン
#endif                                                                  呼出し

    while(TRUE);            /* endless loop                             */
}
```

図Ｂ－９　例題システムのHI-SH7・システム異常終了処理ルーチン(hisysdwn.c)

Ｂ．１３　例題システムのHI-SH7・タスク用スタック領域定義ヘッダ

図Ｂ－１０に例題システムのHI-SH7・タスク用スタック領域定義ヘッダ(hitskstk.h)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*              HI-SH7 header file for task stack ( Ver. 1.0 )          */
/*                                                                      */
/*                                                                      */
/*              Copyright (c)  Hitachi, Ltd. 1993.                      */
/*              Licensed Material of Hitachi, Ltd.                      */
/*                                                                      */
/*              HI-SH7(HS0700ITCN1SM) V1.0                              */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hitskstk.h ;                                            */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 header file for task stack ;                     */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#ifndef     hd_dbg
#include    "hdcommon.h"                                  ←共通ヘッダの読込み
#endif

/************************************************************************/
/*                                                                      */
/*          Definition constant size of task stack                      */
/*                                                                      */
/************************************************************************/
#define KNLTUSESZ       108                 /* size of kernel use       */
```

図Ｂ－１０　例題システムのHI-SH7・タスク用スタック領域定義ヘッダ(hitskstk.h)（１／２）

```
/*****************************************************************************/
/*                                                                           */
/*          Definition task stack size of interrupt use                      */
/*                                                                           */
/*  Usage   :                                                                */
/*                                                                           */
/*  #define inttusesz       12 * <hi_uppintnst> + 20 * <hi_lowintnst>        */
/*                                                                           */
/*      <hi_uppintnst>  :   nest number of upper interrupt                   */
/*                          (>  kernel mask level)                           */
/*      <hi_lowintnst>  :   nest number of lower interrupt                   */
/*                          (<= kernel mask level)                           */
/*                                                                           */
/*****************************************************************************/
#define inttusesz       12 * 0 + 20 * 2 + ADDTSKSTKSZ                        ←割込みが使用するスタックサイズの補正
                                        /* size of interrupt use            */

/*****************************************************************************/
/*                                                                           */
/*          Definition task stack size (multiple of 4)                       */
/*                                                                           */
/*  Usage   :                                                                */
/*                                                                           */
/*  #define hi_tskstk<n>    KNLTUSESZ + inttusesz + <tskusesz> + <excusesz>  */
/*                                                                           */
/*      <n>             :   task stack number                                */
/*      <tskusesz>      :   size of task use                                 */
/*            (caution !!  if using chg_ims by the task, add 28 to tskusesz)*/
/*      <excusesz>      :   size of exception use                            */
/*                                                                           */
/*****************************************************************************/
#define hi_tskstksz1    (KNLTUSESZ + inttusesz +  44 + 0)  /* stack no. = 1 */
#define hi_tskstksz2    (KNLTUSESZ + inttusesz + 108 + 0)  /* stack no. = 2 */
#define hi_tskstksz3    (KNLTUSESZ + inttusesz + 108 + 0)  /* stack no. = 3 */
#define hi_tskstksz4    (KNLTUSESZ + inttusesz + 108 + 0)  /* stack no. = 4 */
```

図Ｂ－１０　例題システムのHI-SH7・タスク用スタック領域定義ヘッダ(hitskstk.h)（２／２）

Ｂ．１４　例題システムのHI-SH7・割込みハンドラ用スタック領域定義ヘッダ

図Ｂ－１１に例題システムのHI-SH7・割込みハンドラ用スタック領域定義ヘッダ(hiintstk.h)を示します。

```
/************************************************************************/
/*                                                                      */
/*                                                                      */
/*             HI-SH7 header file for interrupt handler stack ( Ver. 1.0 ) */
/*                                                                      */
/*                                                                      */
/*             Copyright (c)  Hitachi, Ltd. 1993.                       */
/*             Licensed Material of Hitachi, Ltd.                       */
/*                                                                      */
/*             HI-SH7(HS0700ITCN1SM) V1.0                               */
/*                                                                      */
/*                                                                      */
/************************************************************************/
/************************************************************************/
/*SPECIFICATIONS ;                                                      */
/*  FILE      = hiintstk.h ;                                            */
/*  DATE      = 93/02/01 ;                                              */
/*  AUTHOR    = Hitachi, Ltd. ;                                         */
/*  FUNCTION  = HI-SH7 header file for interrupt handler stack ;        */
/*  ATTRIBUTE = PUBLIC ;                                                */
/*  HISTORY   = V1.0 ;                                                  */
/*END OF SPECIFICATIONS ;                                               */
/************************************************************************/
#ifndef     hd_dbg
#include    "hdcommon.h"                                  ←共通ヘッダの読込み
#endif

/************************************************************************/
/*                                                                      */
/*          Definition constant size of interrupt handler stack         */
/*                                                                      */
/************************************************************************/
#define KNLIUSESZ       132                 /* size of kernel use       */
```

図Ｂ－１１　例題システムのHI-SH7・割込みハンドラ用スタック領域定義ヘッダ(hiintstk.h)（１／２）

```
/*****************************************************************************/
/*                                                                           */
/*         Definition stack size of interrupt handler (multiple of 4)        */
/*                                                                           */
/*  Usage   :                                                                */
/*  case of interrupt level =  not use                                       */
/*    #define hi_intstk<n>  0                                                */
/*                                                                           */
/*  case of interrupt level >  kernel mask level                             */
/*    #define hi_intstk<n>  <othusesz> + <curusesz> + <excusesz>             */
/*                                                                           */
/*  case of interrupt level <= kernel mask level                             */
/*    #define hi_intstk<n>  KNLIUSESZ + <othusesz> + <curusesz> + <excusesz>*/
/*                                                                           */
/*      <n>             :   interrupt level number                           */
/*      <othusesz>      :   size of other interrupt use                      */
/*                      =   12 * <uppintnst> + 20 * <lowintnst>              */
/*          <uppintnst> :   nest number of upper interrupt                   */
/*                         (>  kernel mask level, > current interrupt level)*/
/*          <lowintnst> :   nest number of lower interrupt                   */
/*                         (<= kernel mask level, > current interrupt level)*/
/*      <curusesz>      :   size of current interrupt handler use            */
/*      <excusesz>      :   size of exception use                            */
/*                                                                           */
/*****************************************************************************/
#define hi_intstksz1    0                                       /* level =  1 */
#define hi_intstksz2    0                                       /* level =  2 */
#define hi_intstksz3    0                                       /* level =  3 */
#define hi_intstksz4    0                                       /* level =  4 */
#define hi_intstksz5    0                                       /* level =  5 */
#define hi_intstksz6    0                                       /* level =  6 */
#define hi_intstksz7    0                                       /* level =  7 */
#define hi_intstksz8    0                                       /* level =  8 */
#define hi_intstksz9    0                                       /* level =  9 */
#define hi_intstksz10   0                                       /* level = 10 */
#define hi_intstksz11   (KNLIUSESZ + 12 * 0 + 20 * 1 + ADDINTSTKSZ + 68 + 0)←割込みハンドラ用スタックサイズの補正
                                                                /* level = 11 */
#define hi_intstksz12   0                                       /* level = 12 */
#define hi_intstksz13   (KNLIUSESZ + 12 * 0 + 20 * 0 + ADDINTSTKSZ + 12 + 0)←割込みハンドラ用スタックサイズの補正
                                                                /* level = 13 */
#define hi_intstksz14   0                                       /* level = 14 */
#define hi_intstksz15   0                                       /* level = 15 */
```

図Ｂ－１１　例題システムのHI-SH7・割込みハンドラ用スタック領域定義ヘッダ(hiintstk.h)（２／２）

Ｂ．１５　例題システムのHI-SH7・システム構築用makeファイル

図Ｂ－１２に例題システムのHI-SH7・システム構築用makeファイル(himake)を示します。

```
#*****************************************************************************#
#*                                                                           *#
#*                                                                           *#
#*             HI-SH7 make file ( Ver. 1.0 )                                 *#
#*                                                                           *#
#*                                                                           *#
#*             Copyright (c)  Hitachi, Ltd. 1993.                            *#
#*             Licensed Material of Hitachi, Ltd.                            *#
#*                                                                           *#
#*             HI-SH7(HS0700ITCN1SM) V1.0                                    *#
#*                                                                           *#
#*                                                                           *#
#*****************************************************************************#
#*****************************************************************************#
#*SPECIFICATIONS ;                                                           *#
#*  FILE      = himake ;                                                     *#
#*  DATE      = 93/02/01 ;                                                   *#
#*  AUTHOR    = Hitachi, Ltd. ;                                              *#
#*  FUNCTION  = HI-SH7 make file ;                                           *#
#*  ATTRIBUTE = PUBLIC ;                                                     *#
#*  HISTORY   = V1.0 ;                                                       *#
#*END OF SPECIFICATIONS ;                                                    *#
#*****************************************************************************#
hisystem: \
        hisystem.abs
        lnk -subcommand=himake.sub
        cnvs hisystem.abs hisystem.mot
hisystem.abs: \
        hicnsdrv.obj hiintdwn.obj hiintstk.obj himcuini.obj himempol.obj \
        hisuptbl.obj hisysdwn.obj hisysini.obj hitmrdrv.obj hitrcbuf.obj \
        hitskstk.obj hivcttbl.obj \
        hdsuptbl.obj                                          ←デバッガセットアップテーブルの追加
hicnsdrv.obj: \
        hicnsdrv.c itron.h hicnsdrv.h hiintstk.h
        shc hicnsdrv.c -debug -section=p=hicnsdrv,c=hicnscst,b=hicnswrk
hiintdwn.obj: \
        hiintdwn.c itron.h \
        hdcommon.h                                            ←共通ヘッダの追加
        shc hiintdwn.c -debug -section=p=hiintdwn
hiintstk.obj: \
        hiintstk.c itron.h hiintstk.h \
        hdcommon.h                                            ←共通ヘッダの追加
        shc hiintstk.c -debug -section=b=hiintstk
himcuini.obj: \
        himcuini.c itron.h \
        hdcommon.h                                            ←共通ヘッダの追加
        shc himcuini.c -debug -section=p=himcuini,b=hddbgwrk  ←未初期化データ領域セクションの追加
himempol.obj: \
        himempol.c itron.h himempol.h
        shc himempol.c -debug -section=b=himempol
hisuptbl.obj: \
        hisuptbl.c itron.h hisuptbl.h hisuptbl.inc hitskstk.h \
        himempol.h hitrcbuf.h \
        hdcommon.h hisuptbl.dbg          ←共通ヘッダ、HI-SH7デバッガ組込み用セットアップテーブルインクルードの追加
        shc hisuptbl.c -debug -section=c=hisuptbl,b=hiknlwrk
hisysdwn.obj: \
        hisysdwn.c itron.h \
        hdcommon.h                                            ←共通ヘッダの追加
        shc hisysdwn.c -debug -section=p=hisysdwn
```

図Ｂ－１２　例題システムのHI-SH7・システム構築用makeファイル(himake)（１／２）

```
hisysini.obj: \
        hisysini.c itron.h \
        hdcommon.h                                                    ←共通ヘッダの追加
        shc hisysini.c -debug -section=p=hisysini
hitmrdrv.obj: \
        hitmrdrv.c itron.h hitmrdrv.h hiintstk.h
        shc hitmrdrv.c -debug -section=p=hitmrdrv,c=hitmrcst
hitrcbuf.obj: \
        hitrcbuf.c itron.h hitrcbuf.h
        shc hitrcbuf.c -debug -section=b=hitrcbuf
hitskstk.obj: \
        hitskstk.c itron.h hitskstk.h \
        hdcommon.h                                                    ←共通ヘッダの追加
        shc hitskstk.c -debug -section=b=hitskstk
hivcttbl.obj: \
        hivcttbl.c itron.h \
        hdcommon.h                                                    ←共通ヘッダの追加
        shc hivcttbl.c -debug -section=c=hivcttbl
hdsuptbl.obj: \                                                       ←デバッガセットアップテーブルの追加
        hdsuptbl.c itron.h hdsuptbl.h hdsuptbl.inc hdcommon.h
        shc hdsuptbl.c -debug -section=p=hddbgp,c=hddbgc,d=hddbgd,b=hddbgwrk
```

図Ｂ－１２　例題システムのHI-SH7・システム構築用makeファイル(himake)（２／２）

Ｂ．１６　例題システムのHI-SH7・システム構築用リンケージサブコマンドファイル

図Ｂ－１３に例題システムのHI-SH7・システム構築用リンケージサブコマンドファイル(himake.sub)を示します。

```
;************************************************************************;
;*                                                                      *;
;*                                                                      *;
;*            HI-SH7 subcommand file for linkage editor ( Ver. 1.0 )    *;
;*                                                                      *;
;*                                                                      *;
;*            Copyright (c)  Hitachi, Ltd. 1993.                        *;
;*            Licensed Material of Hitachi, Ltd.                        *;
;*                                                                      *;
;*            HI-SH7(HS0700ITCN1SM) V1.0                                *;
;*                                                                      *;
;*                                                                      *;
;************************************************************************;
;************************************************************************;
;*SPECIFICATIONS ;                                                      *;
;*  FILE      = himake.sub ;                                            *;
;*  DATE      = 93/02/01 ;                                              *;
;*  AUTHOR    = Hitachi, Ltd. ;                                         *;
;*  FUNCTION  = HI-SH7 subcommand file for linkage editor ;             *;
;*  ATTRIBUTE = PUBLIC ;                                                *;
;*  HISTORY   = V1.0 ;                                                  *;
;*END OF SPECIFICATIONS ;                                               *;
;************************************************************************;
;************************************************************************;
;*                                                                      *;
;*                            object file                               *;
;*                                                                      *;
;************************************************************************;
input   hivcttbl.obj,&                  ;* vector table                 *;
        himcuini.obj,&                  ;* MCU initialize routine       *;
        hiintdwn.obj,&                  ;* interrupt down routine       *;
        hisysdwn.obj,&                  ;* system down routine          *;
        hisuptbl.obj,&                  ;* setup table                  *;
        hdsuptbl.obj,&                  ;* setup table of debugger      *;←デバッガセットアップテーブル
        hisysini.obj,&                  ;* system initial handler       *;
        hitmrdrv.obj,&                  ;* timer driver                 *;
        hicnsdrv.obj,&                  ;* console driver               *;
        himempol.obj,&                  ;* memory pool area            *;
        hitskstk.obj,&                  ;* task stack area              *;
        hiintstk.obj,&                  ;* interrupt stack area         *;
        hitrcbuf.obj                    ;* trace buffer area           *;

;************************************************************************;
;*                                                                      *;
;*                            library file                              *;
;*                                                                      *;
;************************************************************************;
library hiknlp.lib,&                    ;* kernel (parameter check)     *;
        hddbg.lib,&                     ;* debugger                     *;←デバッガライブラリ
        hicif.lib                       ;* C language interface         *;
```

図Ｂ－１３　例題システムのHI-SH7・システム構築用リンケージサブコマンドファイル (himake.sub)（１／２）

```
;************************************************************************;
;*                                                                      *;
;*                            section                                   *;
;*                                                                      *;
;************************************************************************;
;*------------------- ROM area (top address : 0x00000000) ------------------*;
start   hivcttbl,&                      ;* vector table                  *;
        himcuini,&                      ;* MCU initialize routine        *;
        hiintdwn,&                      ;* interrupt down routine        *;
        hisysdwn,&                      ;* system down routine           *;
        hisuptbl,&                      ;* setup table                   *;
        hiromtop,&                  ;* ROM top    area of HI-SH7         *;←HI-SH7カーネル領域の先頭セクション
        hiknl,&                         ;* kernel                        *;
        hirombtm,&                  ;* ROM bottom area of HI-SH7         *;←HI-SH7カーネル領域の最終セクション
        hisysini,&                      ;* system initial handler       *;
        hdromtop,&                  ;* ROM top    area of debugger       *;←デバッガ領域の先頭セクション
        hddbgp,hddbgc,hddbgd,&          ;* debugger                      *;←デバッガセクション
        hdrombtm,&                  ;* ROM bottom area of debugger       *;←デバッガ領域の最終セクション
        hitmrdrv,&                      ;* timer driver                 *;
        hitmrcst,&                      ;* constant area of timer driver *;
        hicnsdrv,&                      ;* console driver                *;
        hicnscst,&                      ;* constant area of console driver *;
        hicif(00000000)                 ;* C language interface         *;

;*------------------- RAM area (top address : 0x01000000) ------------------*;←外部RAM領域の定義
start   hdramtop,&                  ;* RAM top    area of debugger       *;←デバッガ作業領域の先頭セクション
        hddbgwrk,&                      ;* work area of debugger         *;←デバッガ作業領域セクション
        hdrambtm,&                  ;* RAM bottom area of debugger       *;←デバッガ作業領域の最終セクション
        hitrcbuf(01000000)              ;* trace buffer area            *;

;*------------------- RAM area (top address : 0x0fffe000) ------------------*;
start   hiramtop,&                  ;* RAM top    area of HI-SH7         *;←HI-SH7カーネル作業領域
        hiknlwrk,&                      ;* work area of kernel           *;              の先頭セクション
        hirambtm,&                  ;* RAM bottom area of HI-SH7         *;←HI-SH7カーネル作業領域
        hicnswrk,&                      ;* work area of console driver  *;              の最終セクション
        himempol,&                      ;* memory pool area             *;
        hitskstk,&                      ;* task stack area              *;
        hiintstk(0fffe000)              ;* interrupt stack area          *;

;************************************************************************;
;*                                                                      *;
;*                            other                                     *;
;*                                                                      *;
;************************************************************************;
entry   _hi_mcuini                      ;* entry address     : _hi_mcuini   *;
output  hisystem.abs                    ;* output file       : hisystem.abs *;
print   hisystem.map                    ;* linkage map file  : hisystem.map *;
form    a                               ;* form              : absolute     *;
debug                                   ;* debug information : output       *;
exit                                    ;* exit                           *;
```

図Ｂ－１３　例題システムのHI-SH7・システム構築用リンケージサブコマンドファイル (himake.sub)（２／２）

Ｂ．１７　例題システムの構築

例題システムは、**「Ｂ．２．２　ソフトウェア構成」**の表Ｂ－１で示した提供ファイルを利用して、構築します。次に構築手順を示します。

なお、以下を仮定して説明します。

・HI-SH7がインストールされているディレクトリ　　：　"hish7"
・デバッガがインストールされているディレクトリ　：　"shdbg"
・作業ディレクトリ　　：　"user"

(1)　作業ディレクトリを作成し、、カレントディレクトリを移動します。

```
% mkdir△user (RET)
% cd△user (RET)
```

(2)　HI-SH7の提供ファイルを(1)で作成した作業ディレクトリへすべてコピーします。

```
% cp△../hish7/*△. (RET)
```

(3)　デバッガの提供ファイルを(1)で作成した作業ディレクトリへすべてコピーします。

```
% cp△../shdbg/*.*△. (RET)
% cp△../shdbg/sample/*△. (RET)
```

(4)　システム構築用makeファイル（himake）を起動します。（ソースプログラムのコンパイルとシステムの結合を行なった後、アブソリュートロードモジュールをSタイプロードモジュールに変換します。）

```
% make△-f△himake (RET)
```

Ｂ．１８　例題システムの起動

(1)　「Ｂ．１７　例題システムの構築」で作成したSタイプロードモジュール（hisystem.mot）をROMライタへ転送し、ROMに書き込みます。

(2)　ROMを実機に搭載します。電源投入（パワーオンリセット）により、システムが起動します。

付録Ｃ．　応答メッセージ

Ｃ．１　デバッグコマンドの応答メッセージ

表Ｃ－１にデバッグコマンドの応答メッセージを示します。

表Ｃ－１　デバッグコマンドの応答メッセージ（１／４）

項番	応答メッセージ	内　容　と　対　策
1	E_DMT	内容：タスクが休止状態である 対策：指定したタスクID、タスク状態を確認してください
2	E_EXS	内容：オブジェクトが存在している（タスクIDのタスクがすでに存在） 対策：指定したタスクIDを確認してください
3	E_IDOVR	内容：ID範囲外（タスクID<0、タスクID>最大タスクID） 対策：指定したタスクIDを確認してください
4	E_ILADR	内容：不正アドレス（アドレスが0、奇数、4の倍数以外） 対策：指定したアドレスを確認してください
5	E_ILBLK	内容：不正メモリブロックの返却 対策：指定したメモリブロックを確認してください
6	E_ILMSG	内容：メッセージの先頭4バイトが0でない 対策：指定したメッセージを確認してください
7	E_ILTIME	内容：年月日時刻データに負の値を指定した 対策：指定した年月日時刻データを確認してください
8	E_NODMT	内容：タスクが休止状態でない 対策：指定したタスクID、タスク状態を確認してください
9	E_NOEXS	内容：・タスク未登録 ・ID範囲外（オブジェクトID<0、オブジェクトID>オブジェクト最大ID） ・予約ID（ID=0） 対策：指定したオブジェクトIDを確認してください
１０	E_NOSUS	内容：タスクが強制待ち状態でない 対策：指定したタスクID、タスク状態を確認してください
１１	E_NOWAI	内容：タスクが待ち状態でない 対策：指定したタスクID、タスク状態を確認してください
１２	E_OBJ	内容：指定したオブジェクトが不正（タスクがデバッグ待ち状態でない） 対策：指定したタスクID、タスク状態を確認してください

表Ｃ－１　デバッグコマンドの応答メッセージ（２／４）

項番	応答メッセージ	内　容　と　対　策
１３	E_OK	内容：コマンドの実行が正常に処理された
１４	E_PAR	内容：パラメータエラー（待ちパターン、待ちモード不正） 対策：指定した待ちパターン、待ちモードを確認してください
１５	E_PLFAIL	内容：ポーリング失敗 対策：指定したオブジェクト状態を確認し、同期通信や資源の獲得ができることを確認してください
１６	E_QOVR	内容：キューイングオーバフロー ・タスクがすでに強制待ち状態である ・タスクがすでにデバッグ待ち状態である ・タスクの起床要求のキューイングオーバフロー ・セマフォのカウント値のオーバフロー 対策：指定したオブジェクトID、オブジェクト状態を確認してください
１７	E_RSFN	内容：コマンドが使用するシステムコールが"NOTUSE"（未使用）である 対策：HI-SH7セットアップテーブルで、コマンドが使用するシステムコールを"USE"（使用）に定義してください
１８	E_RSID	内容：予約ID（タスクID=0） 対策：タスクIDに0は指定できません
１９	E_TNOSPT	内容：タイマがサポートされていない 対策：HI-SH7セットアップテーブルで、タイマ管理システムコールを"USE"（使用）に定義してください
２０	E_TPRI	内容：不正タスク優先度（タスク優先度<0、タスク優先度>最大タスク優先度） 対策：指定したタスク優先度を確認してください
２１	ED_DBGADR	内容：指定したアドレスはHI-SH7またはデバッガの領域である 対策：指定したアドレスを確認してください
２２	ED_DBGID	内容：指定したオブエクトIDは本デバッガで使用している 対策：指定したオブジェクトIDを確認してください
２３	ED_DBGTPRI	内容：指定したタスク優先度は本デバッガで予約している（タスク優先度=1） 対策：タスク優先度=1以外の値を指定してください
２４	ED_OBJ	内容：指定したオブジェクトが不正（タスクが実行状態または共有スタック待ち状態である） 対策：指定したタスクID、タスク状態を確認してください

表Ｃ－１　デバッグコマンドの応答メッセージ（３／４）

項番	応答メッセージ	内　容　と　対　策
２６	*** I01:SYNTAX ERROR	内容：シンタックスエラー 対策：指定したコマンドを確認してください
２７	*** I02:INVALID COMMAND	内容：コマンド不正（指定したコマンドは存在しない） 対策：指定したコマンドを確認してください
２８	*** I03:DATA OVERFLOW	内容：データオーバフロー（指定可能な範囲を越えている） 対策：指定したデータを確認してください
２９	*** I04:INVALID ADDRESS	内容：アドレス不正（アドレスが0、奇数、4の倍数以外） 対策：指定したアドレスを確認してください
３０	*** I05:DEBUG MODE ERROR	内容：デバッグモードエラー ・オフラインモードでオンライン専用コマンドを入力した ・オンラインモードでオフライン専用コマンドを入力した 対策：デバッグモードを確認してください
３１	*** I06:INVALID DATA	内容：データ不正（指定可能なデータサイズを越えている） 対策：指定したデータを確認してください
３２	*** I07:INVALID MASK LEVEL	内容：割込みマスクレベル（SRレジスタのIビット）が15である 対策：割込みマスクレベル（SRレジスタのIビット）を14以下に変更してください
３３	*** I08:INVALID REGISTER	内容：レジスタ不正（タスク指定で、VBRまたはR15を指定した） 対策：オフラインモードでVBRまたはR15を指定してください
３４	*** I09:PARAMETER ERROR	内容：パラメータエラー ・開始オブジェクトID>終了オブジェクトID（範囲指定時） 対策：指定した開始、終了オブジェクトIDを確認してください
３５	*** I10:NOT FOUND	内容：指定したパラメータの対象が見つからない 対策：指定したパラメータを確認してください
３６	*** I11:COUNT ERROR	内容：回数エラー（回数=0） 対策：指定した回数を確認してください
３７	*** I12:DUPLICATE ADDRESS	内容：ソフトウェアブレークポイントとハードウェアブレークポイントのアドレスが重複している 対策：どちらか一方のブレークポイントを解除してください
３８	*** I13:NOT RAM AREA	内容：指定したアドレスがRAM領域でない 対策：指定したアドレスを確認してください
３９	*** I14:NOT USE COMMAND	内容：指定したコマンドは、未使用となっている 対策：デバッガセットアップテーブルで、指定するコマンドを"USE"（使用）に定義してください
４０	*** I15:TOO MANY BREAK POINT	内容：ソフトウェアブレークポイントの設定数が最大値を超えた 対策：不要なソフトウェアブレークポイントを解除してください
４１	*** I16:INVALID ASM MNEMONIC	内容：命令ニーモニック不正（アセンブリ言語） 対策：指定した命令ニーモニックを確認してください
４２	*** I17:INVALID ASM OPERAND	内容：命令オペランド不正（アセンブリ言語） 対策：指定した命令オペランドを確認してください

表Ｃ－１　デバッグコマンドの応答メッセージ（４／４）

項番	応答メッセージ	内　容　と　対　策
４３	*** I20:TIME OUT ERROR	内容：タイムアウトエラー（ホストコンピュータからの応答がない） 対策：シリアル回線が正しく接続されているか確認してください
４４	*** I21:CHECK SUM ERROR	内容：ファイルロードまたはベリファイでチェックサムエラーが発生した 対策：シリアル回線が正しく接続されているか確認してください
４５	*** I22:NOT SAME FILE	内容：ファイルのフォーマットが違う 対策：指定したファイル名を確認してください
４６	*** I23:VERIFY ERROR	内容：ベリファイエラー（メモリとファイルの内容が不一致） 対策：ロードしなおしてください
４７	*** I30:UNDEFINED TRACE BUFFER	内容：トレースバッファ領域が定義されていない 対策：ITRACE_BUFFERコマンドでトレースバッファ領域を定義してください
４８	*** I31:RESERVED ADDRESS AREA	内容：指定したアドレスはHI-SH7またはデバッガの領域である 対策：指定したアドレスを確認してください
４９	*** I32:DEBUGGER USE ID	内容：指定したオブエクトIDは本デバッガで使用している 対策：指定したオブジェクトIDを確認してください
５０	*** I33:DEBUGGER USE TASK PRIORITY	内容：指定したタスク優先度は本デバッガで使用している（タスク優先度=1） 対策：タスク優先度=1以外の値を指定してください
５１	*** I34:UNDEFINED OBJECT	内容：指定したオブジェクト（イベントフラグ、セマフォ、メモリプール）を「未使用」でHI-SH7を構築している 対策：HI-SH7セットアップテーブルで、オブジェクトを使用（最大ID>0）に定義してください
５２	*** I35:RESERVED ID	内容：指定したオブエクトIDは予約されている（ID=0） 対策：指定したオブジェクトIDを確認してください
５３	*** I36:ID OVER	内容：指定したオブエクトIDは範囲外である（ID<0, ID>最大ID） 対策：指定したオブジェクトIDを確認してください
５４	*** I37:TASK IS NO EXIST	内容：指定したタスクは未登録（NON-EXIST）状態である 対策：指定したタスクIDを確認してください
５５	*** I38:TASK IS DORMANT	内容：指定したタスクは休止（DORMANT）状態である 対策：指定したタスクIDを確認してください
５６	*** I39:TASK IS STACK WAIT	内容：指定したタスクは共有スタック待ち状態である 対策：指定したタスクIDを確認してください
５７	*** I40:TASK IS RUN	内容：指定したタスクは実行（RUN）状態である 対策：指定したタスクIDを確認してください
５８	*** I50:MACHINE CODE IS DELAY BRANCH INSTRUCTION	内容：指定したアドレスは遅延分岐命令の次命令アドレスである 対策：指定したアドレスを確認してください
５９	*** I51:MACHINE CODE IS INTERRUPT MASK INSTRUCTION	内容：指定したアドレスは割込み禁止命令の次命令アドレスである 対策：指定したアドレスを確認してください

Ｃ．２　デバッガセットアップテーブルチェックのエラーメッセージ

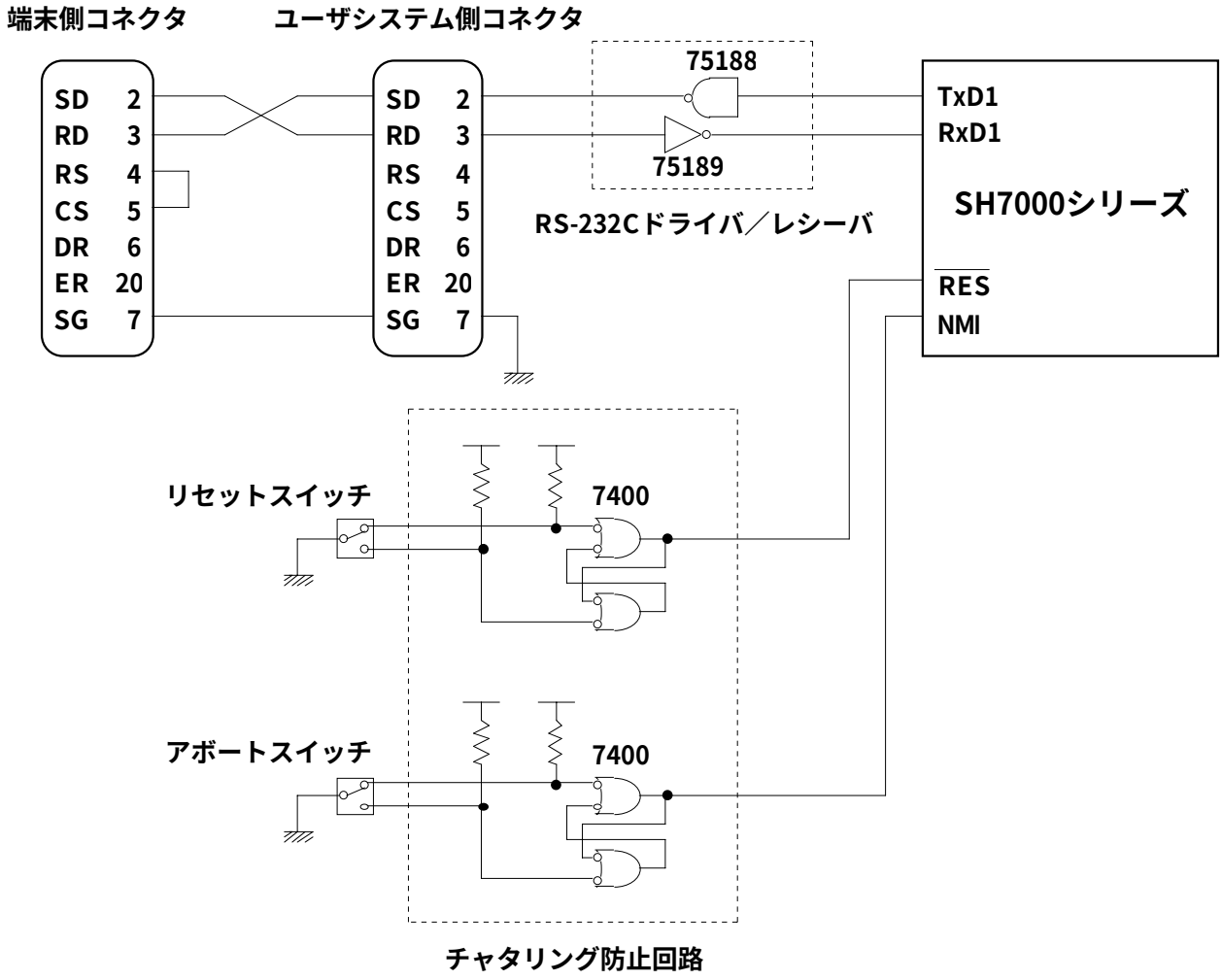
表Ｃ－２にデバッガセットアップテーブルチェックのエラーメッセージを示します。

表Ｃ－２　デバッガセットアップテーブルチェックのエラーメッセージ

項番	エラーメッセージ	対　　策
1	hd_dbgtskid must be [1...hi_maxtskid]	オンラインデバッガタスクのタスクID（hd_dbgtskid）には、HI-SH7セットアップテーブルで定義したタスクIDの範囲内（1～hi_maxtskid）の値を定義してください
2	hd_cnstskid must be [1...hi_maxtskid]	コンソールタスクのタスクID（hd_dbgtskid）には、HI-SH7セットアップテーブルで定義したタスクIDの範囲内（1～hi_maxtskid）の値を定義してください
3	hd_dbgtskid must be [!= hd_cnstskid]	オンラインデバッガタスクのタスクID（hd_dbgtskid）と、コンソールタスクのタスクID（hd_dbgtskid）は、違う値を定義してください
4	hd_dbgmbxid must be [1...hi_maxmbxid]	オンラインデバッガタスクが使用するメイルボックスID（hd_dbgmbxid）には、HI-SH7セットアップテーブルで定義したメイルボックスIDの範囲内（1～hi_maxmbxid）の値を定義してください
5	hd_cnsmbxid must be [1...hi_maxmbxid]	コンソールタスクが使用するメイルボックスID（hd_dbgmbxid）には、HI-SH7セットアップテーブルで定義したメイルボックスIDの範囲内（1～hi_maxmbxid）の値を定義してください
6	hd_dbgmbxid must be [!= hd_cnsmbxid]	オンラインデバッガタスクが使用するメイルボックスID（hd_dbgmbxid）と、コンソールタスクが使用するメイルボックスID（hd_dbgmbxid）は、違う値を定義してください
7	hd_cnsclk must be [> 0]	CPUクロック（hd_cnsclk）には、0以上の値を定義してください（例　20MHz：20000000）
8	hd_cnssci must be [0, 1]	SCIチャンネル（hd_cnssci）には、0または1を定義してください（SCI0：0、SCI1：1）
9	hd_cnsbpsno must be [0...10]	シリアル転送速度番号（hd_cnsbpsno）には、0～10の値を定義してください（例　9600bps：7）
１０	hd_cnsintlvl must be [1...hi_knlmsklvl]	シリアル送受信割込みレベル（hd_cnsintlvl）には、HI-SH7セットアップテーブルで定義したカーネル割込みマスクレベルまで の値（1～hi_knlmsklvl）を定義してください
１１	hd_breaknum must be [1...32]	ソフトウェアブレークの最大ポイント数（hi_breaknum）には、1～32の値を定義してください

付録Ｄ．　ユーザシステムのインタフェース回路例

Ｄ．１　ユーザシステムのインタフェース回路例



図Dー１　ユーザシステムのインタフェース回路例

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	NULL	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

F．索　引

F．１　五十音順・索引

五十音順・索引

タ　行

ハ　行

マ　行

ヤ　行

ラ　行

ワ　行

Ｆ．２　アルファベット順・索引

アルファベット順・索引

A

B

C

D

E

F

G

H

I

L

M

N

P

R

S

T

SH7000 シリーズ　HI-SH デバッガ
ユーザーズマニュアル

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-702-189