お客様各位

# カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

# ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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

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

RENESAS

# ユーザーズ・マニュアル

保守／廃止

# HSM77016

## ハイスピード・シミュレータ

---

μPD77015
μPD77016
μPD77017
μPD77018
μPD77018A
μPD77019
μPD77110
μPD77111
μPD77112
μPD77113
μPD77114

**資料番号**　U11602JJ3V0UM00（第 3 版）
**発行年月**　July　2000 NS　CP（K）

[メ　モ]

保守／廃止

# 目 次 要 約

[メ　モ]

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

**M7A　98.8**

保守／廃止

# 本版で改訂された主な箇所

| 箇　　所 | 内　　容 |
|---|---|
| 全般 | 対象デバイスにμ PD77018A, 77019, 77110, 77111, 77112, 77113,77114, 77116 を追加。 |
| p.26 | 1.3　**オーダ名称と対象** OS を変更。 |
| p.27 | 1.5　**対象デバイスに**μ PD77018A, 77019, 77110, 77111, 77112, 77113,77114 **を追加。** |
| p.30 | 2.3　SP77016 **のセットアップを追加。** |
| p.51 | **第** 5 **章　データ・ウインドウの説明を変更。** |
| p.85 | **第** 6 **章　メニューとメニュー・コマンドの説明を変更。** |
| p.153 | **第** 7 **章　モデルを追加。** |
| p.183 | **第** 9 **章　アプリケーション・プログラム・インタフェースを追加。** |
| p.239 | 11.7 **変数を追加。** |
| p. 259 | **付録** B　**リリース・ノートを追加。** |

**本文欄外の★印は，本版で改訂された主な箇所を示しています。**

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

保守／廃止

# はじめに

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

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

このマニュアルには，μPD77116に関する記述がありますが，本バージョンのHSM77016は，μPD77116には正式に対応しておりませんので，注意してください。

**目　的**　HSM77016は，μPD77016ファミリの応用システムを設計，開発する際に，プログラムを効率よくディバグするための開発ツールです。
このマニュアルは,HSM77016を用いて効率よくプログラムを開発していただくことを目的としています。

**構　成**　このマニュアルでは，大きく分けて次の内容で構成しています。

- 第1章　概　説
- 第2章　セットアップとインストール
- 第3章　ユーザ・インタフェース
- 第4章　ファイル形式
- 第5章　データ・ウインドウ
- 第6章　メニューとメニュー・コマンド
- 第7章　モデル
- 第8章　タイミング・ファイル
- 第9章　アプリケーション・プログラム・インタフェース
- 第10章　プロファイリング
- 第11章　レファレンス
- 付録A　HSM77016のキー
- 付録B　リリース・ノート
- 付録C　索　引

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

HSM77016 **の機能を理解しようとするとき**
**→目次に従ってお読みください。**

**凡　　例**　**↵：キャリッジ・リターン・キーを示します。**
**↑：↑キーを示します。**
**↓：↓キーを示します。**
**→：→キーを示します。**
**←：←キーを示します。**
Esc：Esc **キーを示します。**
Ctrl：**コントロール・キーを示します。**
**カナ：かなキーを示します。**
Alt：Alt **キーを示します。**
Ctrl＋　**：コントロール・キーを押しながら，ほかのキーを押すことを示します。**
Shift＋　**：**Shift **キーを押しながら，ほかのキーを押すことを示します。**
Alt＋　**：**Alt **キーを押しながら，ほかのキーを押すことを示します。**
F1－F12：**ファンクション・キーを示します。**
**アクティブ・ロウの表記　：** $\overline{\times\times\times}$ **（端子，信号の名称に上線）**

| | |
|---|---|
| **注** | **：本文中につけた注の説明** |
| **注意** | **：気をつけて読んでいただきたい内容** |
| **備考** | **：本文中の補足説明** |
| **数の表記** | **：** 2 **進数…××××または** 0b×××× |
| | 10 **進数…××××** |
| | 16 **進数…**0x×××× |

保守／廃止

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

μPD77016 **ファミリに関する資料**

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

**開発ツールに関する資料**

| 資　料　名 | | 資料番号 |
|---|---|---|
| HSM77016　**ユーザーズ・マニュアル** | | **このマニュアル** |
| WB77016　**ユーザーズ・マニュアル** | **言語編** | U10078J |
| | **操作編** | U11506J |
| ID77016　**ユーザーズ・マニュアル** | | U10118J |
| IE-77016-98，IE-77016-PC　**ユーザーズ・マニュアル** | **ハードウエア編** | U13044J |
| μPD77016　**スタータ・キット　ユーザーズ・マニュアル** | | U13032J |
| IE-77016-CM-LC　**ユーザーズ・マニュアル** | | U14139J |
| RX77016　**ユーザーズ・マニュアル** | **機能編** | U14397J |
| | **コンフィギュレーション・ツール編** | U14404J |
| RX77016　**アプリケーション・ノート** | HOST API 編 | U14371J |

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

[メ　モ]

保守／廃止

# 目　　次

保守／廃止

保守／廃止

# 図の目次（1/3）

保守／廃止

# 図の目次（2/3）

保守／廃止

# 図の目次（3/3）

保守／廃止

# 表の目次

# 第1章　概　　説

μPD77016 ハイスピード・シミュレータ（以降 HSM77016 といいます）はディジタル・シグナル・プロセッサ μPD77016 ファミリ[注]用の開発ツールです。

ハードウエアの条件とは関係なくμPD77016 ファミリを利用したシステムを検証するために，HSM77016 は μPD77016 ファミリを機能レベルで完全にシミュレートします。

このマニュアルは，Microsoft Windows 95, 98, Windows NT 4.0 オペレーティング・システムのもとで稼動する HSM77016 の操作方法について述べています。

**注**　μPD77016 ファミリはμ PD7701x ファミリ（μ PD77015, 77016, 77017, 77018, 77018A, 77019）とμ PD77111 ファミリ（μ PD77110, 77111, 77112, 77113, 77114）の総称です。

**注意**　このドキュメント中にμPD77116 についての記述がありますが，現在の HSM77016 ではμPD77116 に正式には対応していません。

## 1.1 特　　徴

HSM77016 では，メモリのレイアウトと対像となるプロセッサのタイプを構成するため，システム・アーキテクチャ・ファイル（.model）が使用されています。このファイルにはターゲット・システムに関する基本情報が格納されます。所定の時間に実行された I/O 動作やイベントによって生じた I/O 動作に関する情報を含むタイミング・ファイルを利用できます。シミュレーション状態をセッション・イメージ・ファイルに格納して，あとでセッションを再開することもできます。そのほかにも次のプログラム開発機能を提供します。

- ソース・ファイル内で実行をトレースするソース・レベル・シミュレーション
- 様々な形式のメモリ表示とレジスタ・データ表示
- 多様な実行モードとアニメート・モードでのシミュレーション
- 定義済み条件と動作を含むブレークポイント
- プログラムを分析，改良するプロファイリング方法
- カスタム関数（プラグイン）をシミュレーションに接続するアプリケーション・インタフェース
- 様々な形式でプログラムとデータをロード／セーブ
- 複数のデータ・ウインドウの任意位置への配置（表示できるデータ・ウインドウ）
  - メモリ・データとレジスタ・データ
  - ソース・コード・データ
  - ブレークポイント・プロパティ
  - イベント・ログ・データ
  - Watch 式
  - ソース・モジュール
  - タイミング・ファイル・データ
  - プロファイリング・マーカ・プロパティ

保守／廃止

## 1.2 表記法

このマニュアルで使用されている記号／用語の意味を表 1－1 に示します。

表1－1　表記法

| 表記法／用語 | 使用方法 |
|---|---|
| モノスペース | モノスペース・テキストは，プログラム例やプログラム出力に使用されます。 |
| **太字** | 太字は，文字どおりに使用すべき固有の用語を示します。これらの用語は，表示のとおり正確に入力する必要があります。ただし，大文字と小文字の区別は必要ありません。 |
| *斜体* | 斜体文字を使用したテキストは，仮の表現です。ユーザは実際に入力する値に置き換える必要があります。 |
| かぎかっこ　[...] | かぎかっこによって任意のフィールドやパラメータが表現されます。 |
| 垂直バー　\| | 垂直バーは，バーの両側に表されたエントリの 1 つを入力することを要求します。 |
| Windows | Windows 95，98，Windows NT4.0 オペレーティング・システム。 |
| μPD77016 ファミリ | μPD77015, 77016, 77017, 77018, 77018A, 77019, 77110, 77111, 77112, 77113, 77114。 |
| μPD77116 | 命令キャッシュ，命令 DMA，データ DMA を持ったディジタル・シグナル・プロセッサ。 |
| DSP | ディジタル・シグナル・プロセッサ μPD77016 ファミリ，μPD77116。 |
| IE-77016 | インサーキット・エミュレータ（IE-77016-PC または IE-77016-98）。 |

## ★ 1.3 オーダ名称と対象 OS

| ホスト・マシン | 対象 OS | オーダ名称（媒体） |
|---|---|---|
| PC-9800 シリーズ | Windows 95, 98,および Windows NT 4.0 | μSAB17SM77016-H（CD-ROM） |
| IBM PC/AT™ | | |

保守／廃止

## 1.4 HSM77016 パッケージ内容

HSM77016 のパッケージには，次のものが含まれています。

- HSM77016 Windows アプリケーション：μPD77016 ファミリのソフトウエアまたはハードウエア開発をサポートします。
- Log File Viewer アプリケーション：DSP ツールで作成されたログ・ファイルの管理に使用します。
- Model Editor：シミュレーションの基礎になる DSP システム・アーキテクチャを記述するモデルの新規作成または既存モデルの修正に使用します。
- シミュレータ・オンライン・マニュアル：ユーザ・インタフェース，ファイル・タイプ，およびコマンド構文に関する包括的なオンライン情報を提供します。
- DSP オンライン・マニュアル[注]：μPD77016 ファミリの製品を記載したデータ・ブックを含み，あらゆるプログラミングの状況に応じたヘルプを提供します。

**注　開発にあたっては，最新のデータ・シートを使用してください。**

- HSM77016 マニュアル
- セットアップ・プログラム：GUI ベースのインストーラ
- アンインストール・プログラム：ユーザ・システムからインストール済みコンポーネントを削除するために必要なタスクすべてを実行します。
- API（アプリケーション・プログラム・インタフェース）サンプル・ファイル：HSM77016 の API 関数の実行例を提供します。

## ★ 1.5 対象デバイス

HSM77016 の対象デバイスは次のとおりです。

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

**注意　このドキュメント中にμPD77116 についての記述がありますが，現在の HSM77016 ではμPD77116 に正式には対応していません。**

〔メ　モ〕

保守／廃止

# 第2章　セットアップとインストール

## 2.1　HSM77016 のインストール

インストール手順を示します。

（1）OS（Windows 95/98, Windows NT4.0）を起動してください。

**注意**　Windows NT4.0 の場合 Administrator 権限でログインします。

（2）CD-ROM を CD-ROM ドライブにセットします。

（3）エクスプローラなどから，CD-ROM ドライブの “ ¥HSM77016¥DISKI¥setup.exe”を起動してください。

（4）セットアップ・ウィザード手順に従ってインストール作業を進めてください。

セットアップによって，すべてのファイルがハードディスクにコピーされ，タスク・バーのプログラム・メニューに新しいメニューが作成されます。さらに，HSM77016 アプリケーションのアンインストールに必要なレジストリ・エントリも作成されます。

＜補足＞

Japanese ディレクトリに次のファイルがあります。HSM77016 がインストールされたディレクトリに上書きまたは追加することで使用できます。

- DSP オンライン・マニュアル（日本語版）
- モデル・エディタ HELP（日本語版）

## 2.2　HSM77016 のアンインストール

（1）コントロール・パネルから “ アプリケーションの追加と削除 ” を選択し，プログラムの追加／削除ダイアログ・ボックスを開きます。

（2）インストール／アンインストール・タブを選択したら，Atair μPD77016 High-Speed Simulator をクリックし，追加／削除ボタンを押してください。

アンインストールは，セットアップによって作成されたファイル，フォルダ，メニューを削除します。

アンインストールの実行には，セットアップによって作成されたレジストリ・エントリ・セットアップ・ログ・ファイルが必要になります。

**注意**　SP77016 をご購入のお客様は，2.3　SP77016 のセットアップをお読みください（SP77016 は，WB77016，HSM77016，ID77016 がパッケージされた商品です）。HSM77016 単体のインストール，アンインストール手順とは異なります。

保守／廃止

## ★ 2.3　SP77016 のセットアップ

（1）OS を起動してください。
（2）SP77016 の CD-ROM を CD-ROM ドライブにセットします。
（3）エクスプローラなどから，CD-ROM ドライブの“ ATAIR” ディレクトリにある“ Setup.exe”をダブル・クリックしてインストール・プログラムを起動します。
（4）インストール・プログラムの指示に従ってください。
（5）ID77016 をご使用になる場合は，さらにデバイス・ドライバの組み込みが必要です。2.4 から 2.5 を参照して，デバイス・ドライバの組み込みを行ってください（IE-77016-98/PC がお使いのホスト・マシンに正しく設置されている必要があります）。

## 2.4　デバイス・ドライバ組み込み手順（Windows 95/98 編）

デバイス・ドライバの組み込みには，IE-77016-98/PC がお使いのホスト・マシンに正しく設置されている必要があります（詳細は，IE-77016-98, IE-77016-PC ユーザーズ・マニュアル　ハードウエア編を参照してください）。

### 2.4.1　デバイス・ドライバの組み込み

Windows 95/98 でのデバイス・ドライバの組み込み手順を次に示します。

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

2.4.2　デバイス・ドライバの設定変更に従いデバイス・ドライバの設定を変更してください。

### 2.4.2　デバイス・ドライバの設定変更

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

（4）JTAG クロックの変更：

IE-77016-PC EM1 Evaluation Board のプロパティ・ダイアログで Settings タブを選択します。クロック周波数は，5，2.5，1.25MHz および 625kHz から選択できます（ディフォールトは 5 MHz です）。もし非常に長いケーブルが使われるか，IE-77016-98/PC とのリンクが信頼できないようならば，低速のクロック周波数を選択します。

（5）内部および外部リンクの属性の変更：

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

（6） I/O アドレスの変更：

IE-77016-PC EM1 Evaluation Board のプロパティ・ダイアログでリソース・タブを選択します。

[設定の変更（c）…]ボタンをクリックし，IE-77016-98/PC のベース・アドレスを設定します。I/O アドレス値には次のアドレスが使用できます。

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

IE-77016-PC:　0x300 （ディフォールト）, 0x320 など

（7）コンピュータの再起動を行います。

## 2.5　デバイス・ドライバ組み込み手順（Windows NT4.0 編）

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

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

●手動で IE77016 Control Board のコンフィギュレーションを開始するには次のとおりにしてください。

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

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

●IE77016 Control Board のコンフィギュレーションを行うには次のとおりにしてください。

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

Board Name:

IE77016 Control Board に対しての通称を入力します。

Base I/O Address:

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

Clock Rate:

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

Internal link:

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

External Link:

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

●ドライバの再スタート:

コンフィギュレーションの変更を行った場合，ドライバの再スタートが必要です。

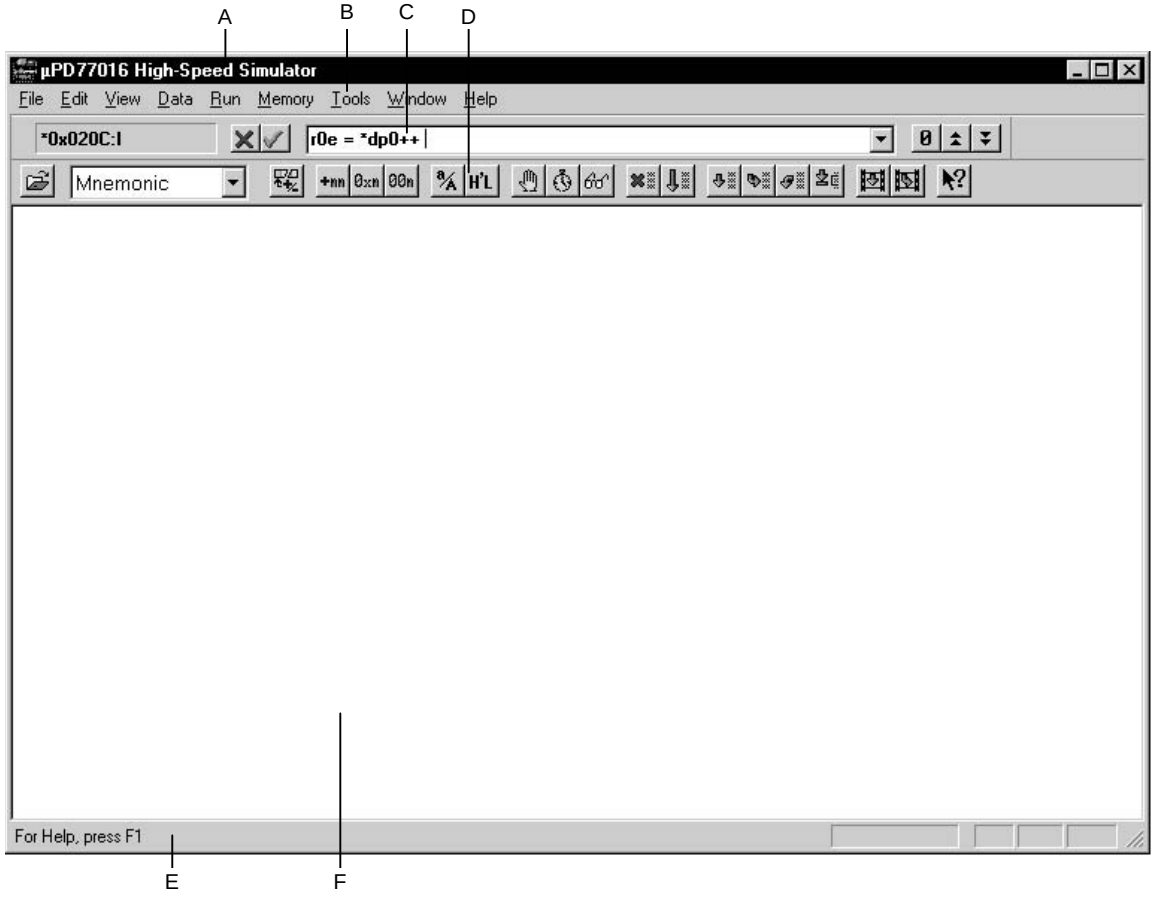
(1) [スタート]ボタン ― [設定] ― [コントロール・パネル]によりコントロール・パネルを開きます。
(2) [デバイス]をダブル・クリックします。
(3) リストから “ IE77016 Control Boards ” を選択します。
(4) [停止]ボタンをクリックします。
(5) [開始]ボタンをクリックします。

# 第3章　ユーザ・インタ フェース

## 3.1　メイン・ウインドウの構成要素

**図** 3ー1 **は，**HSM77016 **のメイン・ウインドウです。**

**図** 3ー1　HSM77016 **のメイン・ウインドウ**




A：**タイトル・バー**　　B：**メニュー・バー**　　C：**エディット・バー**

D：**ツール・バー**　　E：**ステータス・バー**　　F：**ワーク・スペース**

**［メイン・ウインドウの構成］**

**タイトル・バー**

HSM77016 **のメイン・ウインドウの上端にあるタイトル・バーには，アプリケーション・プログラム名，システム・メニュー・ボックス，およびメイン・ウインドウを最小化，最大化する** 2 **つのボタンがあります。ファイルをロードした場合は，このバーにアクティブ・ファイルの名前が表示されます。**

**メニュー・バー**

**タイトルバーの下にあるバーで，すべての** HSM77016 **コマンドにアクセスできます。**

**エディット・バー**

**エディット・バーは，**HSM77016 **のデータ・ウインドウのデータを編集，変更するために使用します。データ・ウインドウには，シミュレーション対象の** DSP **に関する情報（レジスタ，メモリなど）が表示されます。**

**ツール・バー**

**マウスをクリックすると，エディット・バーの下に表示されます。使用頻度の高いコマンドに素早くアクセスできます。**

**ステータス・バー**

**メイン・ウインドウの下端にあるステータス・バーには，ユーザ・サポート・メッセージが表示されます。**

**ワーク・スペース**

**データ・ウインドウを表示するスペースです。**

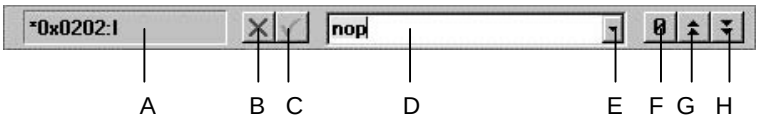
### 3.1.1 メニュー・バー

**メニュー・バーの項目のいくつかは，**HSM77016 **のデータ・ウインドウ（メモリ・ウインドウ，レジスタ・ウインドウなど）に対応しています。これらのメニュー項目は，アクティブなデータ・ウインドウに応じて一時的に表示されるだけです。ウインドウ別のメニュー・タイトルは，メニュー項目の** Run **と** Tools **の間に表示されます。メニュー・バーの項目とメニュー・コマンドは，**HSM77016 **で登録されているプラグイン機能によって変わることもあります。**

### 3.1.2 エディット・バー

**エディット・バーはメニュー・バーの下に表示されます。エディット・バーは，セルにデータを入力したり，セル内のデータを編集したりするために使用します。このバーは重要なので，常に表示されており，非表示にすることはできません。図** 3−2 **にエディット・バーを示します。**

**図** 3−2　**エディット・バー**




A：**参照フィールド**
B：**キャンセル・ボタン**
C：OK **ボタン**
D：**エディット・フィールド**
E：**リストの表示**
F：**ゼロ・ボタン**
G：**インクリメント・ボタン**
H：**デクリメント・ボタン**

### (1) エディット・バーのボタンとフィールド

**参照フィールド**

**現在アクティブなセルの名前が表示されます。**

OK **ボタン，キャンセル・ボタン**

**マウスでクリックすることで入力した内容を有効，無効にするボタンです。同じ動作をキーボードの** Enter **（**OK**）キーと** Esc**（**Cancel**）キーでも行えます。これらのボタンが表示されるのは，データを編集しているときだけです。**

**エディット・フィールド**

**このフィールドでセルへの入力を行います。データを入力するには，セルを選択してから値を入力します。データを編集するには，セルを選択して** F2 **を押すか，マウスでエディット・フィールドをクリックします。**

**リストの表示**

**このボタンをマウスでクリックすると，このセル・クラスに対する最新の入力内容のリスト（履歴リスト）か，使用可能なセル値のリスト（選択リスト）のどちらかが表示されます。**

**ゼロ・ボタン，インクリメント・ボタン，デクリメント・ボタン**

**あらかじめ決まった動作でセルの修正を素早く行うためのボタンです。ボタンの動作は，選択されているセルすべてに適用されます。ゼロ・ボタンはセルの内容をゼロにリセットします。インクリメント・ボタンはセルの内容を** 1 **増分し，デクリメント・ボタンは** 1 **減分します。これらのボタンを使用できるのは，選択されているセルでそのような動作（ゼロ，クリア，数値の増減）がサポートされている場合だけです。それ以外の場合は，これらのボタンを使用できません。マウスを右クリックすれば，ゼロ，インクリメント，デクリメントの各コマンドとも** Data **メニューでも使用できます。**

### (2) 編集例

**次に，編集例をあげ，**HSM77016 **でよく使用される編集方法について説明します。**

#### (a) 値フィールドの編集（履歴リストの表示）

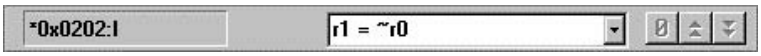
**図** 3－3 **は，値フィールドの編集例です。ここでは，アドレス** 0x202 **番地の命令のニモニック内容が編集されています。**Instruction Memory **ウインドウが手前に表示されており，アドレス** 0x202 **のセルにアンカが付いています。データ形式はニモニックに設定されます。**

**注意　履歴リストには，最後に使用された編集値が示されます。**

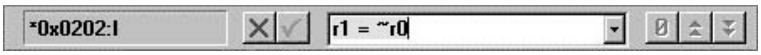
保守／廃止

図 3−3　値フィールドの編集例

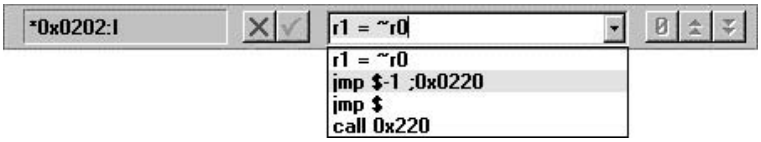
(a) 編集前のエディット・バー




(b) 編集中のエディット・バー




(c) 履歴リストを開いて編集中のエディット・バー




**(b) ビット・フィールドの編集（選択リストの表示）**

図 3−4 はビット・フィールドの編集例です。ここでは，ステータス・レジスタの割り込み許可（EI）ビットの状態が編集されています。CPU Register ウインドウが手前に表示されており，割り込み許可（EI）ビットのセルにアンカが付いています。

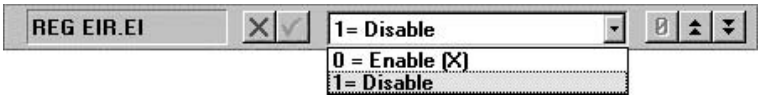
**注意**　選択リストには，選択されたセルに対して編集可能な値が表示されます。これらの値は，ビット・フィールド・グループ(レジスタ内のすべてのビットを含むグループ)に対して選択された表示形式によって決まります。ビット・フィールド表示形式は，Data メニューの Format... コマンドで変更できます。ステータス表示形式と値表示形式を利用できます。詳細については，6.4.1　Format... を参照してください。

図 3−4　ビット・フィールドの編集例

(a) 編集前のエディット・バー


(b) 編集中のエディット・バー


(c) 選択リストを開いて編集中のエディット・バー

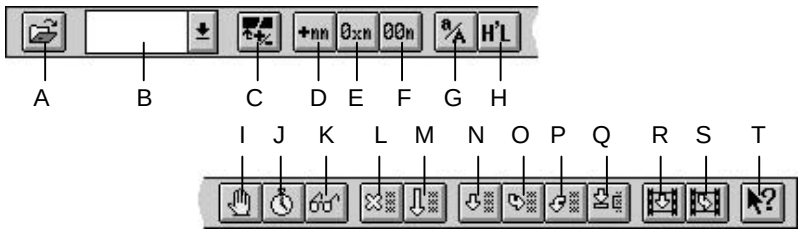


保守／廃止

## 3.1.3　ツール・バー

エディット・バーの下に表示されるツール・バーには，よく使用する HSM77016 コマンドへ迅速にアクセスできるボタンがあります。View メニューの Toolbar コマンドで，ツール・バーのオンとオフを切り替えることができます。

**図 3−5　ツール・バー**




A：Open ボタン
B：数値形式フィールド
C：Signed Value ボタン
D：Show Positive ボタン
E：Show Base ボタン
F：Leading Zero ボタン
G：Uppercase ボタン
H：Split ボタン
I：Toggle Breakpoint ボタン
J：Toggle Marker ボタン
K：Watch ボタン
L：Break ボタン
M：Run ボタン
N：Step ボタン
O：Trace ボタン
P：Until RET ボタン
Q：To Cursor ボタン
R：Animate Step ボタン
S：Animate Trace ボタン
T：Context Help ボタン

### （1）ツール・バーのボタンとフィールド

Open ボタン

セッション・イメージ・ファイル（.SIM）リンク・ファイル（.LNK）をロードしたり，View ウインドウにテキスト・ファイル（アセンブラ・ソース・ファイルなど）を表示するための Open ダイアログ・ボックスを表示します。

数値形式フィールド

現在選択されている数値形式を表示します。数値形式を変更するには，ドロップ・ダウン・リストを開き，希望の数値表示形式を選択します。

Signed Value ボタン

数値を符号付きの 2 の補数で表示します。最上位桁は符号ビットです。

Show Positive ボタン

正符号 “ ＋ ” を付けて正数を表示します。

Show Base ボタン

数値とともにその基数も表示します。たとえば，16 進数には “ 0x ” が付きます。

Leading Zeros **ボタン**
**先行ビットがゼロの場合もゼロを表示します。**

Uppercase **ボタン**
**数値の文字を大文字で表示します。**

Split **ボタン**
40 **ビット値を** 8，16，16 **ビットに分けて表示します。このボタンが有効なのは，値が** 40 **ビットの場合だけです。**

Toggle Breakpoint **ボタン**
Instruction Memory **ウインドウのアンカ／カーソル位置に実行ブレークポイントを設定したり，設定を解除したりします。**

Toggle Marker **ボタン**
Instruction Memory **ウインドウまたは** Module **ウインドウのアドレスに，希望のタイプのプロファイリング・マーカを設定したり，設定を解除したりします。**

Watch **ボタン**
**式を入力するための** Evaluate And Modify Variable **ダイアログ・ボックスを開きます。**

Break **ボタン**
**シミュレーションを中断します。中断したシミュレーションは，**Run **ボタンを押せば再開できます。シミュレーションが停止し，ブレーク発生箇所の状態を反映するように表示が更新されます。**

Run **ボタン**
**シミュレーションを開始します。非明示ブレークポイントは無視されます。明示ブレークポイントはアクティブです。いったんシミュレーションが始まると，このボタンは無効になります。シミュレーションは，**Break **ボタンが選択されるか，ブレークポイントに到達するまで続きます。シミュレーション中は表示の再更新は行われません。**

Step **ボタン**
**サブルーチン内部の実行の途中経過を表示しないで，**1 **ステップ実行します。ただし，現在の命令がサブルーチン・コール命令（**CALL**）の場合は，全サブルーチンを実行します。現在の命令が** LOOP **または** REPEAT **対象命令の最後の命令である場合は，**LOOP **または** REPEAT **命令を完了し，このような命令の次の命令で実行が停止します。**

Trace **ボタン**
**サブルーチン内部の実行の途中経過を表示して，**1 **ステップ実行します。**

Until RET **ボタン**
**現在のサブルーチンからの** RETURN **が実行されるまでシミュレーションを実行します。**

To Cursor ボタン

インストラクション・メモリの現在のカーソル・アドレスに到達するまで，シミュレーションを実行します。このボタンが有効なのは，Instruction Memory または Module ウインドウが手前に開いており，アンカまたはカーソルがそこに表示されている場合だけです。それ以外の場合は無効です。

Animate Step ボタン

ステップ・モードで実行し，画面を更新し，一定の遅延の間待ち，自動的に次の命令から実行します。

Animate Trace ボタン

トレース・モードで実行し，画面を更新し，一定の遅延の間待ち，自動的に次の命令から実行します。

Context Help ボタン

カーソル形状を状況に応じたヘルプ・カーソルに変えます。参照したいオブジェクトをクリックすれば，HSM77016 のユーザ・インタフェースのどの部分に関するヘルプ情報でも参照できます。このボタンの代わりに Shift＋F1 を押しても，状況に応じたヘルプ・カーソルを表示できます。

### 3.1.4　ステータス・バー

HSM77016 のメイン・ウインドウの下端にあるステータス・バーには，ユーザ・サポート・メッセージが表示されます。View メニューの Status Bar コマンドで，ステータス・バー表示のオンとオフを切り替えることができます。

図 3－6 にステータス・バーを示します。

図 3－6　ステータス・バー


A：メッセージ・フィールド
B：ステータス・フィールド
C：キー・ステータス・フィールド（Caps Lock）
D：キー・ステータス・フィールド（Num Lock）
E：キー・ステータス・フィールド（Scroll Lock）

#### （1）ステータス・バーのフィールド

メッセージ・フィールド

すべての HSM77016 メニュー・コマンドに関する説明が表示されます。ただし，希望のコマンドを反転表示しておく必要があります。また，現在の HSM77016 動作に対応するメッセージが表示されます。

ステータス・フィールド

HSM77016 のモードが示されます。モードには次のものがあります。

- Animation：シミュレーションの実行中です（Run メニューの Animate コマンドで開始されました）。
- Break：シミュレーションが中断しました。
- HALT instruction：シミュレーション中に HALT 命令が実行されました。
- Run：シミュレーションの実行中です（Run メニュー・コマンドで開始されました）。
- STOP instruction：シミュレーション中に STOP 命令が実行されました。
- Stop：シミュレーションが中止されました。

**キー・ステータス・フィールド**

キーボードの Caps Lock，Num Lock，Scroll Lock キーの状態が表示されます。

- CAP：文字キーの機能を切り替えます。
- NUM：数値キーパッドの機能を切り替えます。
- SCRL：スクロール・ロックが有効な場合，矢印キーでスクロールできるのはデータ・ウインドウの内容だけです。スクロール・ロックが無効の場合，矢印キーでアンカを移動できます。アンカがデータ・ウインドウの端に達すると，データ・ウインドウの内容が自動的にスクロールします。

## 3.2　データ・ウインドウ

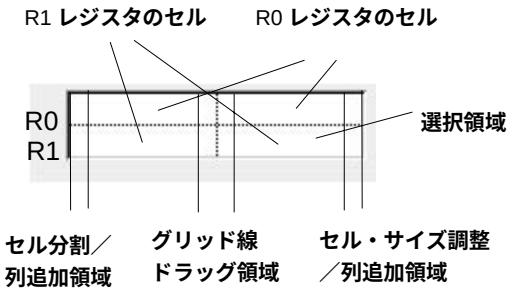
HSM77016 のデータ・ウインドウはすべて，Windows メニューのコマンドで開くことができます。

### 3.2.1　セル

セルにはシステムの状態が表示されます。エディット・バーでセルを修正できます。セルは選択することができ，図 3−8 から図 3−10 に示すように 3 つの状態があります。1 つのウインドウ内で複数のセルを選択できますが，アンカを持つセルは 1 つだけです。

図 3−7 にセルの例を示します。

**図 3−7　セルの例**




隣接するセル同士はグリッド線で分割されます（図 3−7 を参照）。セルの横 1 行には同じ項目についての情報が表示され，縦 1 列には各種情報が表示されます。縦に連続するセルまたはセル行をセル・ブロックと呼びます。同一セル・ブロックでは，すべての行でセル幅と数値表示形式が同じになります（ただし，Watch ウインドウは例外です。5.4　Watch ウインドウを参照してください）。

保守／廃止

## 3.2.2　セルの処理

**(1) 選択領域**

セルの中央にある領域です。この領域をクリックすると，そのセルを選択できます。

**(2) セルの選択**

1 つのセルを選択するには，選択領域をクリックします。

連続する複数のセルを選択するには，最初のセルを選択してからカーソルをその他の選択したいセルにまたがってドラッグするか，最初のセルをクリックしたあとに Shift キーを押したまま最後のセルをクリックします。この方法では，同一ブロック内のセルしか選択できません。

複数のブロックにある連続していないセルを選択するには，Ctrl キーを押したまま選択したいセルをクリックします。

**(3) セル・サイズ調整領域**

この領域は，右端のセルの右側にあります。カーソルをこの領域に移動すると，カーソル形状がサイズ調整カーソル（↔）に変わります。マウス・ボタンを押してサイズ調整カーソルをドラッグすると，セル・ブロックのサイズを変更できます。

新しく設定したセル・ブロック・サイズよりも幅の広い列や最小セル幅よりも狭い列は，閉じられます。ただし，1 つの行にセルを少なくとも 1 つ残す必要があります。

**(4) グリッド線ドラッグ領域**

隣接する 2 つの列の境界にある領域です。カーソルをこの領域に移動すると，カーソル形状がグリッド線ドラッグ・カーソル（⇿）に変わります。このカーソルで，グリッド線の左の列を拡張したり縮小できます。グリッド線の左列の幅が最小セル幅よりも狭く設定されると，その列は閉じられます。

**注意　サイズ調整する列の左側の境界を越えてグリッド線をドラッグしても，サイズ調整した列の左にある列を閉じることはできません。**

**(5) セル分割／列追加領域**

この領域は，左端のセルの左端にあります。それ以上セルを分割できない場合，この領域は表示されず，選択領域が左端セルの左の境界まで伸びます。カーソルをこの領域に移動すると，カーソル形状がセル分割カーソル（⇿）に変わります。新しい列を作成し，ブロックにドラッグすることができます。

新しい列は左端の列として追加され，元の左端セルの表示形式属性をすべて引き継ぎます。ドラッグによるその他のサイズ調整方法は，グリッド線ドラッグ領域で説明したものと同じです。

**(6) 右列追加**

新しい列を追加する別の方法として，列が希望のサイズになるまで Ctrl キーを押したままドラッグ&ドロップする方法があります。右端の列のサイズは変わりませんが，新しい列が追加され，元の右端の列の表示形式属性がすべて引き継がれます。

保守／廃止

### 3.2.3　セル形状

セルは，その機能に応じて異なる形状で表示されます。その例を図3－8と3－9に示します。

**図 3－8　読み出し専用セル**


図3－8に示したような，読み出し専用セルの内容を，ユーザが変更することはできません。

**図 3－9　ビット・フィールド・セル**

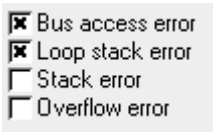



図3－9に示したセルには，エディット・フィールドの選択リストから選択できる小さい範囲の編集値しか表示できません。

**注意**　(☒, ☐)や(0, 1)などのビット・フィールド・セルは，サイズ調整や結合はできません。ビット・フィールドの表示形式は，Data メニューの Format... コマンドで変更できます。ビット・フィールド・セルは，次の形式で表示されます。

- ステータス表示形式：ビット・フィールドに×マークが付いている(☒)は，セルの右側の表示内容が肯定されたことを意味します。ビット・フィールドに何もない(☐)は，表示内容が否定されたことを意味します。
- 値表示形式：　1(1)はセット・ビットを示し，ゼロ(0)は非セット・ビットを示します。

ステータス表示形式と値表示形式の対応は，個々のビットによって異なります。これは，セット／非セットに関係なく，ビットはアクティブになることができるからです。

保守／廃止

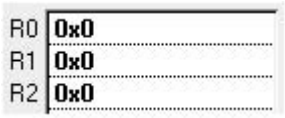
### 3.2.4　セルの選択状態

セルは選択することができ，図3－10から図3－12に示すように3つの状態があります。1つのウインドウ内で複数のセルを選択できますが，アンカを持つセルは1つだけです。アンカとは1つのセルを囲む枠のことで，そのセルが選択されていることを表します。

**(1) 通常の状態**

どのセルにもアンカがなく，選択されていません。

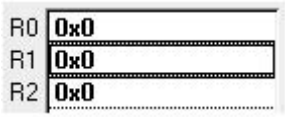
**図 3－10　通常の状態**




**(2) アンカ選択状態**

セルR1が選択されています。このセルをエディット・バーで編集できます。

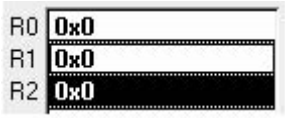
**図 3－11　アンカ選択状態**




**(3) アンカ分離選択状態**

セルR1にアンカがあります。セルR2が選択された状態が反転色表示で示されています。

**図 3－12　アンカ分離選択状態**




**注意　アンカの付いたセル以外は編集できません。同時に選択されているすべてのセルが，エディット・バーのボタン(ゼロ，インクリメント，デクリメント)に反応します。**

### 3.2.5　アンカの移動

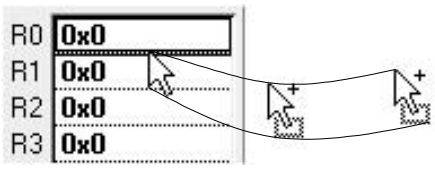
次のいずれかの方法でアンカを移動できます。

- 移動先のセルをクリックします。
- Tab キーを使用します。Tab キーを押すと，アンカが次のセルに進みます。Shift＋Tab キーを押すと，アンカが前のセルに戻ります。
- 矢印キーを使用します。矢印キーを押すと，そのキーの方向にアンカがセル 1 つ分だけ移動します。ただし，Scroll Lock を無効にしておく必要があります。Scroll Lock の詳細については，3.1.4　ステータス・バーを参照してください。

## 3.3　データ値のドラッグ&ドロップ

HSM77016 のメモリ・ウインドウとレジスタ・ウインドウのセル内容（データ値）は，マウスだけでほかのセルにコピーできます。セル内容をコピーするには，まず，アンカを希望のセルに配置するかセル・ブロックを選択します。次に，カーソルの先端を選択フレームに正確に合わせ，マウス・ボタンをクリックします。

図 3－13　データ値のドラッグ




カーソルがデータ・ウインドウを越えると，カーソル形状が次のように変わります。


：データをドロップできない領域にカーソルがきたとき


：データをドロップできるセルにカーソルがきたとき

**注意　セルにデータをドロップできても，データ形式がそのセルに一致していない場合は，ドロップ後にメッセージが表示されます。**

## 3.4　サウンドのサポート

複数のシミュレータ・イベントをサウンドに割り当てることができます。
シミュレータ・イベントをサウンドに割り当てる方法は，次のとおりです。

（1）Windows のコントロール・パネルを開き，サウンド・アイコンをダブルクリックしてサウンド・プロパティを表示します。
（2）イベント・リストで，サウンドに割り当てる HSM77016 のイベントをクリックします。
（3）選択したシミュレータ・イベントが起こった場合に Windows に再生させるサウンドを，ネーム・リストで選択します。希望のサウンドがリストにない場合は，ブラウズをクリックします。

## 3.5　ヘルプ・システム

Windows のヘルプによって，必要な情報を素早く知ることができます。ヘルプ情報には次の 2 種類があります。

（1）HSM77016 ヘルプ

このヘルプには，HSM77016 アプリケーションに関する次の情報があります。

- コマンド
- キーボード
- HSM77016 ファイル
- HSM77016 のウインドウ
- 参照情報
- ダイアログ・インデクス

（2）オンライン・マニュアル

このヘルプには，DSP に関する次の情報があります。

- レジスタ・セット
- アドレス・モード
- データ形式
- 命令セット
- 割り込み
- 内蔵ペリフェラル
- メモリ・インタフェース
- AC/DC ターゲット仕様

Help メニュー，Help キー，またはダイアログ・ボックスの Help ボタンでヘルプを表示できます。API（アプリケーション・プログラム・インタフェース）ヘルプは，HSM77016 のセットアップ・セッション中に指定したスタート・メニュー・フォルダで表示できます。

保守／廃止

### 3.5.1　Help キー

（1）F1

Help キー（F1）を押すと，Help アプリケーション・ウインドウが表示されます。Help ウインドウに最初に表示される情報は，その状況に応じて異なります（対応できない場合もあります）。

（2）Shift+F1

Shift＋F1 を押すと，マウス・カーソルの形状が状況に応じたヘルプ・カーソルに変わります。HSM77016 ユーザ・インタフェースのどの部分のヘルプでも，興味のあるオブジェクトをクリックするだけで表示できます。

（3）Ctrl＋F1

Ctrl＋F1 を押すと，マウス・カーソルの形状が状況に応じたヘルプ・カーソルに変わります。希望の命令をクリックすれば，Instruction Memory ウインドウのμ PD77016 命令セットアップに関する状況に応じたヘルプを表示できます。また，希望のレジスタまたはビット・フィールドをクリックすれば，CPU Register ウインドウまたは Peripheral Register ウインドウのμ PD77016 CPU とペリフェラル・レジスタに関する，状況に応じたヘルプを表示できます。

（4）Ctrl＋Shift＋F1

Ctrl＋Shift＋F1 を押すと，マウス・カーソルの形状が状況に応じたヘルプ・カーソルに変わります。希望の命令をクリックすれば，Instruction Memory ウインドウのμ PD77016 命令セットアップに関する状況に応じたヘルプを表示できます。また，希望のレジスタまたはビット・フィールドをクリックすれば，CPU Register ウインドウまたは Peripheral Register ウインドウのμ PD77016 CPU とペリフェラル・レジスタに関する，状況に応じたヘルプを表示できます。

### 3.5.2　ヘルプ・イン・ダイアログ・ボックス

ダイアログ・ボックスを使用しているときは，Help ボタンをクリックするか Help キー（F1）を押せば，現在のダイアログ・ボックスに関するヘルプをいつでも表示できます。

Help キーの詳細については，付録 A　HSM77016 のキーを参照してください。

ダイアログ・ボックス項目の定義を表示する方法は，次の 2 つです。

（1）タイトル・バーからの場合

1. ダイアログ・ボックスのタイトル・バーのクエスチョン・マーク（?）をクリックします。ポインタの形状が変わります。
2. ポインタを希望のダイアログ・ボックス項目に移動し，クリックします。クリックした領域に関する情報を含むポップアップ・ウインドウが表示されます。
3. ポップアップ・ウインドウの内側をクリックして閉じます。

（2）ダイアログ・ボックスからの場合

1. 右マウス・ボタンでダイアログ・ボックス項目をクリックします。What'sThis？メニューが表示されます。
2. 左マウス・ボタンで What's This？メニューをクリックします。クリックした領域に関する情報を含むポップアップ・ウインドウが表示されます。
3. ポップアップ・ウインドウの内側をクリックして閉じます。

# 第4章　ファイル形式

この章では，HSM77016 のファイル形式について説明します。

## 4.1　データ・ファイル

データ・ファイルには，HSM77016 がサポートしている形式のデータ値が含まれます。データ値は，書式情報を持たないテキストで構成されています。そのため，ユーザは，これらの形式をサポートするテキスト・エディタなどを利用できます。

アドレス情報やそれに類似の情報は含まれません。データ値は，スペースまたはカンマ（,）で区切られます。データ・ファイルの中にコメントを書き込む場合は，コメントの前にセミコロン（;）を付けてください。データ・ファイルは，一般的なアプリケーション・プログラムとのインタフェースになります。

File メニューの Import コマンドまたは Export コマンドで，データ・ファイルを入力または出力（インポートまたはエクスポート）できます。

## 4.2　16 進ファイル

HSM77016 によって，16 進形式ファイルをインストラクション・メモリ，X データ・メモリ，Y データ・メモリのいずれかのメモリ空間に入力できます。次のファイル拡張子がサポートされています。

- HEX：標準 16 進形式のデータを含む共通 16 進ファイル
- HXI ：インストラクション・メモリ用の標準 16 進形式のデータを含むインストラクション・メモリ 16 進ファイル
- HDX：X データ・メモリ用の標準 16 進形式のデータを含むファイル
- HDY：Y データ・メモリ用の標準 16 進形式のデータを含むファイル

**備考**　HXI，HDX，HDY 拡張子の付いたファイルは，通常，WB77016 で生成されます。

## 4.3　リンク・ファイル

リンク形式ファイルは，DSP ツール同士のインタフェースとなるμPD77016 実行可能ファイルです。これらのファイルは，HSM77016 で入力または出力（インポートまたはエクスポート）できます。ファイルには次のデータが含まれます。

- 選択されたセグメントのメモリ内容
- 選択されたセグメントのセグメント／シンボル・テーブル
- ソース・ディバグ情報

保守／廃止

## 4.4　ログ・ファイル

ログ・ファイルには，ログ・セッションの内容が格納されます。ログ・セッションには，シミュレーション動作，シミュレーション・メッセージ，およびユーザが選択したデータが記録されます。

ログ・ファイルは，Log Viewer ツールで管理または表示できます。ロギングはいつでもできますが，シミュレーション・セッション中にはログ・ファイルを指定できません。ログ・ファイルを指定しない場合，HSM77016 アプリケーションを閉じるときに，ログ・セッション・データが拒否されます。

## 4.5　レポート・ファイル

プロファイリング・セッション中に Statistic ウィンドウに収集されたデータを，レポート・ファイルに保存できます。これは書式情報のないテキスト形式のファイルで，次の情報が格納されます。

- ファイル・ヘッダ：レポートを生成したプログラムの名前とバージョン，レポート・ファイルのパスと名前，レポートの作成日時
- プログラム統計情報：レポートが関係するリンク・ファイルのパスと名前
- プロファイル統計情報：マーカ，タイム・カウント，ヒット・カウント，データのソート順序，Statistic ウィンドウに収集されたデータを含むテーブル
- タイプ凡例：データ・テーブルのマーカ・タイプ・シンボルの定義

**図 4−1　レポート・ファイル例**

```
---------------------------------------------------------------------
ATAIR µPD77016 High-Speed Simulator  Version 2.1 Beta 4
D:¥DSP¥FILES¥REPCALL1.TXT created Thursday, 5. August 1997 16:12:08
---------------------------------------------------------------------

Program Statistics
------------------
  File name: D:¥DSP¥FILES¥CALLTST0.LNK

Profile Statistics
------------------
  Total markers: 7
  Total hits: 18
  Total execution time: 5.94µs
  Covered execution time: 8.07µs (136%)
  Sorted by Address. (ascending)

T Marker       Hit         Exec      Time Per
y Address    Count   %      Time   %     Count     Notes
-------------------------------------------------------------
L 0x0200         0   0.0     0ns   0.0     0ns
- 0x0206       n/a   n/a     n/a   n/a     n/a
+ 0x0208       n/a   n/a     n/a   n/a     n/a
> 0x020D         2  11.1    120ns   1.5    60ns
< 0x020E         4  22.2     n/a   n/a     n/a
> 0x020F         6  33.3    540ns   6.7    90ns
< 0x0212         6  33.3     n/a   n/a     n/a

Type Legend
-----------
  L  Label Marker
  >  Function Entry Marker
  <  Function Exit Marker (Execution Time measurement not
     applicable)
  +  Start Marker (Start profiling)
  -  Stop Marker (Stop profiling)
```

保守／廃止

## 4.6　セッション・イメージ・ファイル

セッション・イメージ・ファイルは，シミュレータ・セッションの共通作業ファイルです。このファイルには，次のようなシミュレーション・セッション状態が格納されます。

- メモリ値
- レジスタ値
- 端子とポートの状態
- 時間とカウント測定変数 CYCLE，STEP，TIME，TIME_RESOLUTION の値
- シミュレーション・モデル

セッション・イメージ・ファイルには，アクティブなリンク・ファイルが記録されます。セッション・イメージ・ファイルの書き込み以降にリンク・ファイルに変更があれば，セッション・イメージ・ファイルのロード時にリンク・ファイルがチェックされます。リンク・ファイルが最新であれば，シンボルとディバグの情報がリンク・ファイルからロードされます。リンク・ファイルが違う場合は，新しいファイルをロードするかどうかダイアログ・ボックスが質問します。新しいファイルをロードしないと，シンボルとディバグの情報を入手できません。

セッション・イメージ・ファイルは，File メニューのコマンド New，Open...，Save，Save as...で処理できます。

## 4.7　設定ファイル

設定ファイルには，個々の HSM77016 の構成情報が格納，ロードされます。このファイルには，ユーザが選択した HSM77016 ユーザ・インタフェースの詳細な構成情報（データのウインドウ位置，サイズ，色など）と，シミュレーション・セッションの設定内容（ブレークポイント，ウォッチなど）が含まれます。構成設定内容は，Tool メニューの Options...コマンドで変更，ロード，格納，表示できます。

## 4.8　ソース・ファイル

ソース・ファイルは，μPD77016 アセンブラ言語の構文に応じたソース・テキストを含むテキスト・ファイルです。ソース・ファイルは，ソース・レベル・シミュレーションで HSM77016 が使用します。リンク・ファイルのディバグ情報で，リンク・ファイルの作成に使用されたソース・ファイルを指定します。

関連するソース情報のあるアドレスにシミュレーションが到達すると，Module ウィンドウに対応するソース・ファイルが表示されます。

さらに，Memory メニューの Show Source コマンドを選択すれば，インストラクション・メモリの内容とともにソース・コードが表示されます。ただし，表示できるのは Instruction Memory ウィンドウがアクティブの場合だけです。

**注意**　HSM77016 は，WB77016 で出力されたリロケータブル・ファイル（拡張子 REL）を処理しません。そのため，アセンブラ・ファイルを HSM77016 に入力するには，アセンブルしてリンクする必要があります。
リンク・ファイルに未解決な外部シンボルが含まれている場合，HSM77016 はそのファイルのロードを拒否します。リンクが不十分なプログラム・ファイルはロードできません。ファイル・ロード終了時にエラーが発生した場合，通常，HSM77016 にはすでにロードされたファイルの一部がセットされているので，ロード・コマンド実行前の状態に戻すことはできません。

保守／廃止

## 4.9　タイミング・ファイル

タイミング・ファイルは，所定の時間に実行された動作やイベントによって生じた動作に関する情報を含むテキスト・ファイルです。DSP のシミュレーションまたはその他のタイミング・ファイルに同期する，短期間のシミュレーションに関する情報が含まれます。

タイミング・ファイルの特徴は，次のとおりです。

- 複数のタイミング・ファイルを同時にアクティブにできます。アクティブにできるファイル数は，システムの空きメモリとシミュレーション速度によって制限されます。アクティブなタイミング・ファイルの数が増えると，空きメモリとシミュレーション速度が減少します。
- 入出力に複数のデータ・ファイルを使用できます。どんなタイミング・ファイルでも，データを複数のデータ・ファイルから入出力できます。データ・ファイルとタイミング・ファイルの間には固定した関係はありません。
- プロセッサ・シミュレーションと同期しても同期しなくても実行できます。
- イベント駆動アーキテクチャによって，ほとんどすべてのイベントまたは状態でタイミング・ファイルに同期できます。
- ほとんどすべてのシミュレーション状態を，様々な形式のデータ・ファイルに出力できます。
- IF-THEN-ELSE 文，REPT 文，DO 文およびタイミング・ファイル変数などの高度構造化言語を使用できます。
- 明示ブレークポイントと非明示ブレークポイントを指定できます。

保守／廃止

★ # 第5章　データ・ウインドウ

## 5.1　メモリ・ウインドウ

HSM77016 のメモリ・ウインドウは，それぞれのμPD77016 ファミリのメモリ領域に対応しています。Instruction Memory ウインドウには命令コードが表示され，X-Data Memory ウインドウと Y-Data Memory ウインドウにはデータが表示されます。μPD77116 に対するシミュレーション・セッションの場合，X-DMA Data Memory ウインドウと Y-DMA Data Memory ウインドウを使用できます。これらのウインドウには，μPD77116 DMA のメモリ・コントローラがアクセスするメモリ空間が " 外部メモリ空間 " として表示されます。

DSP のメモリ空間の詳細については，μPD7701x ファミリ，またはμPD77111 ファミリ　ユーザーズ・マニュアル　アーキテクチャ編を参照してください。

メモリ・ウインドウには次のデータが表示されます。

- アドレス情報
- メモリ・データ情報
- ソース・レベル・データ
- セグメント情報
- シンボル情報
- アドレス・ポインタ情報
- ブレークポイント情報
- プロファイリング・マーカ情報

### 5.1.1　メモリ内容の表示形式

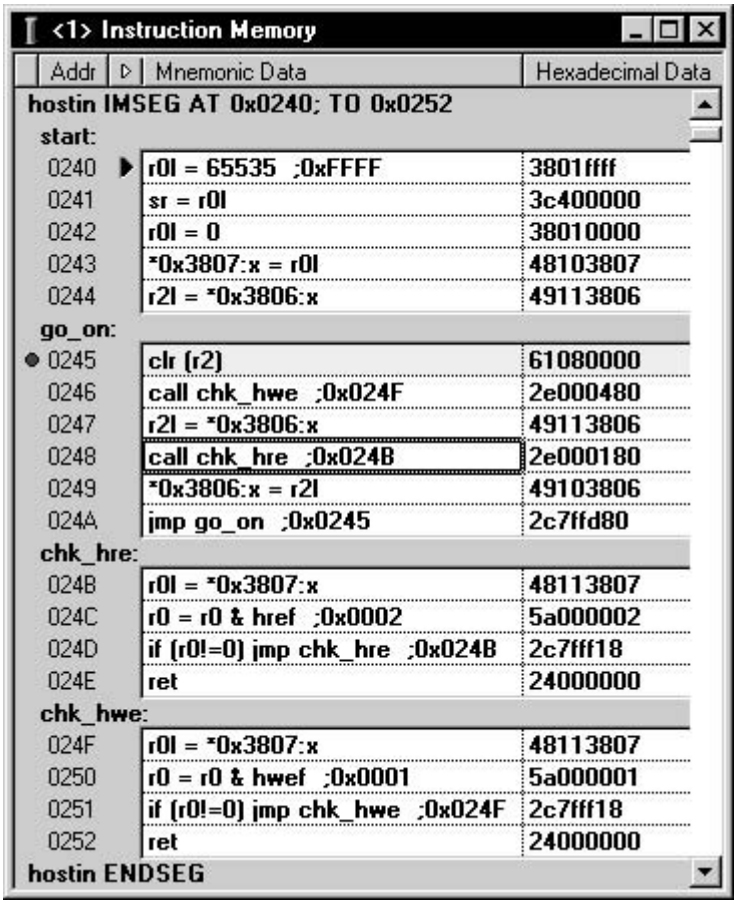
メモリ内容は，メモリ・アドレスによって示されるアドレス行（セル）に表示されます。データ列の数はユーザが選択できますが，少なくとも 1 つのデータ列が必要です。各データ列はそれぞれ異なるデータ表示形式に設定できます。現在のデータ形式が列見出しに表示されます。

表示形式は，ウインドウ別の Memory メニューの Format...コマンドで変更できます。変更内容は選択されているすべての列に反映されます。

Format...コマンドの詳細については，6.4.1　Format...を参照してください。

図 5－1 に Instruction Memory ウインドウの例を示します。

図 5−1　Instruction Memory ウインドウの例




## 5.1.2 ソース・レベル・データ

ソース・レベル・データは，Instruction Memory ウインドウの命令コードと組み合わせて表示することができます。表示されるソース・データは，リンク・ファイル・ディバグ情報から引き出されます。ソース・データ・ブロックは，関連するソース・ファイルの名前が表示されたヘッダ行で強調しています。ソース行には，行番号とそのソース行に含まれるソース・コードが表示されます。関連するソース行を持たない命令メモリの内容は，ソース・データ・ブロックの最後に収集され，“No File”ヘッダ行で強調しています。

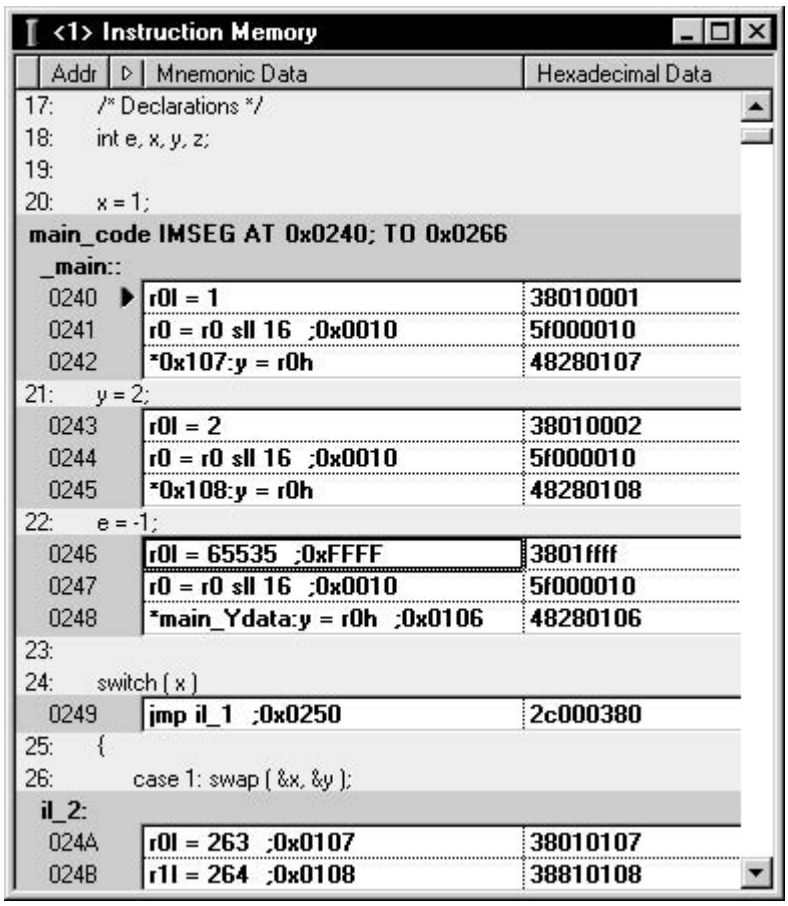
ソース・レベルのブレークポイントまたはプロファイリング・マーカは，関連する命令メモリ・コードの最初のアドレスに設定されます。

Memory メニューの Show Source コマンドでソース・レベル・データの表示を有効にできます。ソース表示色は，Tools メニューの Options...コマンドで変更できます。

図 5−2 に，ソース・レベル・データが表示されている Instruction Memory ウインドウを示します。

保守／廃止

図 5－2　ソース・レベル・データが表示されている Instruction Memory ウインドウ




### 5.1.3　セグメントの表示

セグメント行はセグメントの開始または終了を示します。セグメント行は，セグメントの開始アドレスの直前，およびそのセグメント行で示されるセグメントに属するシンボル行の前に表示されます。

### 5.1.4　レーベルの表示

レーベルとはアドレスを示す名前です。メモリ・ウインドウでは，アドレスと一致するデータ行の上の特別な行にレーベルが表示されます。HSM77016 では，特定のアドレス 1 つに対して複数のレーベルを定義できます。この場合，各レーベルは別々の行に表示されます。

レーベル表示のオンとオフは，Memory メニューの Show Symbol コマンドで切り替えることができます。

**(1) パブリック・レーベルとローカル・レーベル**

パブリック・レーベルは，末尾にコロンが 2 つ付きます。**例**　start::

ローカル・レーベルは，末尾にコロンが 1 つ付きます。**例**　lp:または wait:

### 5.1.5　存在しないメモリ領域

現在のシミュレーション・モデルに含まれていないメモリ・ロケーションには値がありません。アドレス行には，値の代わりに“ (no memory) ”と表示されます。これらのメモリ領域の範囲はシミュレーションされないため，このようなメモリ・ロケーションに値を割り当てることはできません。これらのアドレス行を編集しようとすると，HSM77016 からのメッセージが表示されます。

### 5.1.6　メモリ内容の編集

メモリ値を編集するには，対応するセルにアンカがなければなりません。編集は HSM77016 のエディット・バーで行います。セルの編集方法については，3.2　**データ・ウインドウ**を参照してください。

**注意　インライン・アセンブラを使用するには，命令を含むセルの表示形式がニモニック形式でなければなりません。ニモニック形式の複数の命令フィールドは，スペースで区切られなければなりません。**μPD77016 **アセンブラの構文規則については，**WB77016 **ユーザーズ・マニュアル　言語編を参照してください。**

**（1）**X-Data Memory **ウインドウと** Y-Data Memory **ウインドウの変数**

X-Data Memory **ウインドウと** Y-Data Memory **ウインドウのデータ値として入力できるのは，次のものです。**

- **数値**
- HSM77016 **変数。データ値として** HSM77016 **変数を指定するには，**11.7　**変数で説明する構文規則に従います。**
- HSM77016 **演算子で組み合わされた数値および** HSM77016 **変数。**HSM77016 **が認識できる演算子については，**11.6　**演算子を参照してください。**

**注意　現在の** HSM77016 **の状態によっては，変数が未定義値として評価されることがあります。**

### 5.1.7　アドレス・ポインタ

**アドレス・ポインタは，**Addr **列の右列に表示されます。アドレス・ポインタ式は，**Memory **メニューの** Pointer... **コマンドで定義します。アドレス・ポインタ式は，指定されたポインタ式がある位置のアドレスを示します。**

**（1）ポインタのドラッグ**

**アドレス・ポインタは希望のアドレスに移動できます。ポインタを移動するには，ポインタをクリックしてドラッグします。図** 5－3 **のように，カーソル形状がポインタ・ドラッグ・カーソルに変わります。ポインタ・ドラッグ領域は，**Addr **列の右列です。**

**図** 5－3　**ポインタ・ドラッグ・カーソル**

保守／廃止

### 5.1.8　実行ブレークポイントの設定

Instruction Memory ウインドウ内で実行ブレークポイントを設定したり削除できます。Addr 列と Data 列の間の領域でカーソル形状が図 5−4 のブレークポイント・カーソルに変わったら，クリックして設定や削除を行います。

**図 5−4　ブレークポイント・カーソル**

この設定や削除は，Memory メニューの Toggle Breakpoint コマンドを選択しても，ツール・バーの Toggle Breakpoint ボタンをクリックしても行えます。

**注意**　Instruction Memory ウインドウのブレークポイント群は，Module ウインドウの関連するソース行に表示されます。ソース・モジュールのブレークポイントが削除されると，関連する命令コード・ブロックのすべてのブレークポイントも削除されます。
ソース行のブレークポイント群は，関連する命令メモリ・コードの最初のアドレスに自動設定されます。
ブレークポイントを含むアドレス行の色は，Tools メニューの Colors...コマンドで変更できます。

### 5.1.9　プロファイリング

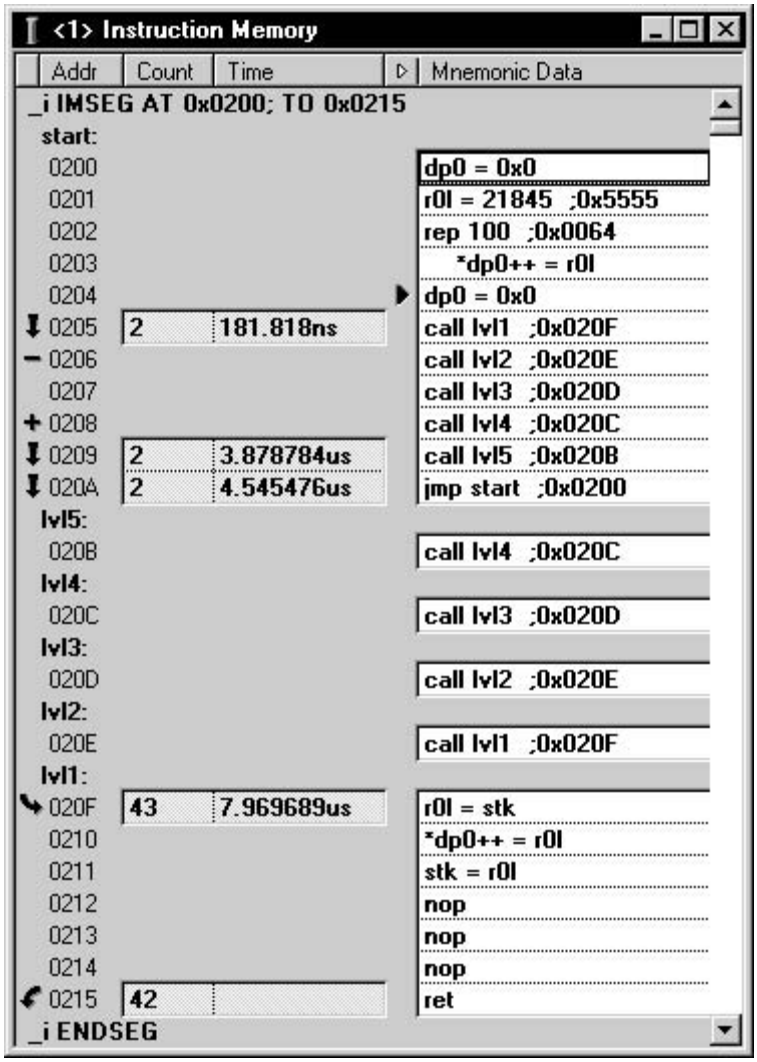
Instruction Memory ウインドウのプロファイリングには次の処理が含まれます。

- プロファイリング・マーカの設定と削除
- プロファイリング・データの検査

図 5−5 に，プロファイリング・マーカが含まれている Instruction Memory ウインドウを示します。

保守／廃止

**図 5−5　プロファイリング・マーカが含まれている Instruction Memory ウインドウ**




マーカ・カラムは，アドレス・カラムとポインタ・カラムの間に挿入されます。マーカ・カラムのセルは表示専用であり，結合することはできません。プロファイリング・データの右側に次のマーカ・シンボルが表示されます。

- ：Label マーカを示します
- ：Entry マーカを示します
- ：Exit マーカを示します
- ：Start マーカを示します
- ：Stop マーカを示します

**注意**　1 つのアドレスには 1 つのマーカしか設定できません。ブレークポイントとプロファイリング・マーカを同じアドレスに設定することはできません。ブレークポイントが設定されているアドレスにマーカが設定された場合，実行ブレークポイントをプロファイリング・マーカに置き換えるようダイアログ・ボックスにメッセージが表示されます。

**(1) プロファイリング・マーカの設定と解除**

Instruction Memory ウインドウでプロファイリング・マーカを設定する方法は，次のいずれかです。

- カーソルを目的のアドレスの右に移動します。カーソル形状がマーカ・カーソル（図 5−6 を参照）に変わってからクリックすると，プロファイリング・マーカが設定されます。マーカのタイプを変更するときや設定／解除を切り替えるときには，マーカ・カラムを次々にクリックしていきます。

- 目的の Marker Toggle コマンドを Memory メニューから選択するか，ツール・バーから Toggle Marker ボタンを選択します。マーカのタイプを変更するときや設定／解除を切り替えるときには，マーカ・ボタンを次々にクリックしていきます。

**図 5−6　マーカ・カーソル**

**(2) プロファイリングの始動**

Stop マーカ以外のプロファイリング・マーカの位置に命令実行が到達すると，プロファイリングが始動します。プログラム実行中にマーカを通過した回数が Count カラムに表示されます。Time カラムには，次のマーカに達するまでの実行時間が表示されます。

## 5.2　レジスタ・ウインドウ

CPU Register ウインドウと Peripheral Register ウインドウには，シミュレートされたμPD77016 ファミリ・プロセッサのレジスタ情報が表示されます。

### 5.2.1　CPU Register ウインドウ

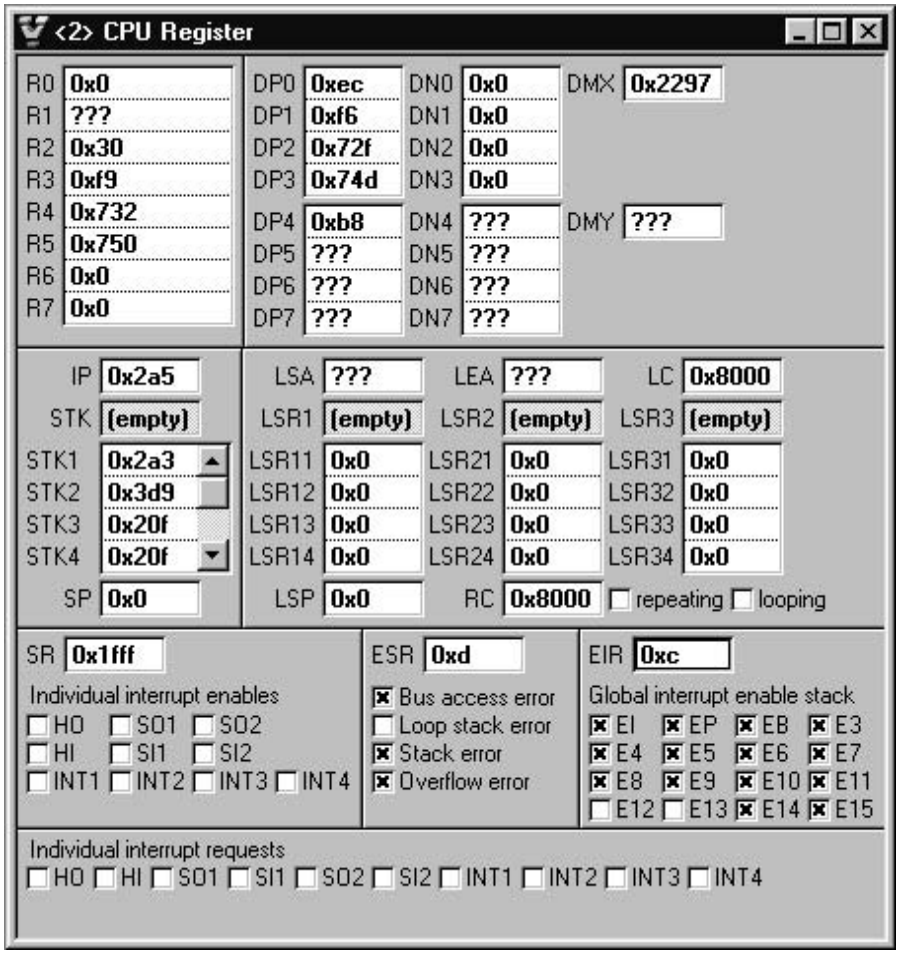
CPU Register ウインドウには，現在のシミュレーション・モデルのすべてのレジスタ情報が表示されます。このウインドウの各ブロックには，次のレジスタ情報が含まれます。

- 汎用レジスタ
- X-Data Memory および Y-Data Memory のレジスタ，ポインタ・レジスタ，インデクス・レジスタ，モジュロ・レジスタ
- インストラクション・ポインタ，スタック・レジスタ，スタック・ポインタ・レジスタ
- ループ・レジスタ，ループ・スタック・レジスタ，ループ・スタック・ポインタ・レジスタ，リピート・レジスタ
- ステータス・レジスタ
- エラー・ステータス・レジスタ
- 割り込み許可フラグ・スタック・レジスタ

図 5−7 は CPU Register ウインドウです。

保守／廃止

図 5ー7　CPU Register ウインドウ




### (1) CPU Register ウインドウのレジスタとフラグ

- R0-R7

  8 本の 40 ビット汎用レジスタです。オペランドの入出力およびメモリとのデータのロード／ストアを行います。

- DP0-DP7

  X メモリおよび Y メモリのデータ・ポインタです。間接アドレシング用の 16 ビット・レジスタです。

- DN0-DN7

  X メモリおよび Y メモリのインデクス・レジスタです。データ・ポインタ修正用の 16 ビット・レジスタです。

- DMX, DMY

  モジュロ・レジスタです。リング・バッファ操作でデータ・ポインタを修正する場合にリング・カウント範囲を指定するレジスタです。

- IP, STK1-STK15, SP

  IP は，インストラクション・メモリ・アドレスを示す 16 ビットのインストラクション・ポインタです。STK1-STK15 は，インストラクションのプッシュ／ポップ用の 16 ビット×15 レベルのレジスタです。SP は，スタックの現在の先頭を示す 16 ビットのスタック・ポインタ・レジスタです。STK フィールドに，スタック・ポインタ・レジスタ（スタックの先頭）で示されるスタック・レジスタの内容が表示されます。

- LSA, LEA, LC, looping

  LSA は，ループ範囲の開始アドレスを示す 16 ビットのループ開始アドレス・レジスタです。LEA は，ループ範囲の終了アドレスを示す 16 ビットのループ終了アドレス・レジスタです。LC は，ループ回数を示す 16 ビットのループ・カウンタ・レジスタです。looping は，シミュレーションのループ・モードを示す 1 ビットです。

- LSR11-LSR34

  （3×16）ビット×4 レベルのループ・スタック・レジスタです。LSA，LEA，および LC のプッシュ／ポップ用です。

- LSP

  16 ビットのループ・スタック・ポインタ・レジスタです。ループ・スタック・レジスタのレベルを示します。LSR1，LSR2，および LSR3 フィールドに，ループ・スタック・ポインタ・レジスタで示されるループ・スタック・レジスタの内容が表示されます。

- RC, repeating

  RC は，リピート回数を示す 16 ビットのリピート・カウンタ・レジスタです。repeating は，HSM77016 のリピート・モードを示す 1 ビットです。

- SR

  16 ビットのステータス・レジスタです。各割り込みの許可／禁止を示します。

- ESR

  16 ビットのエラー・ステータス・レジスタです。CPU のエラー・ステータスを示します。

- EIR

  16 ビットの割り込み許可フラグ・スタック・レジスタです。割り込み許可フラグのスタック用です。

- Individual interrupt requests

  DSP の 4 つの外部割り込みと 6 つの内部割り込みの割り込み要求を示すフラグです。

保守／廃止

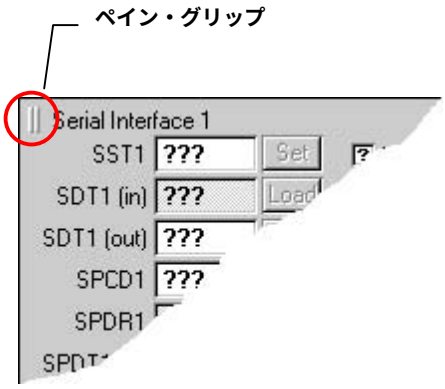
## 5.2.2　Peripheral Register ウインドウ

このウインドウには，シミュレーション・モデル・プロセッサのポート・レジスタ，ポート・フラグ，およびウエイト・サイクル・レジスタが表示されます。表示される項目は，現在使用されている DSP のデバイス・モデルによって異なります。

### (1) Peripheral Register ウインドウのブロック

Peripheral Register ウインドウは複数のブロックに分かれています。各ブロックには，特定の周辺デバイスのレジスタが表示されます。ブロックのペイン・グリップ（図 5－8 を参照）を左クリックしたまま目的の場所に移動すれば，ブロックの位置を変更できます。ペイン・グリップがあるのは，μPD77116 の場合だけです。各ブロックは別々に開いたり閉じることができます。ブロックの空白部分をクリックすると，コンテキスト・メニューが表示されます。このメニューで，ブロックを開くか閉じるかを選択できます。チェック・マークはブロックを表示することを示しています。

**図 5－8　ペイン・グリップ**



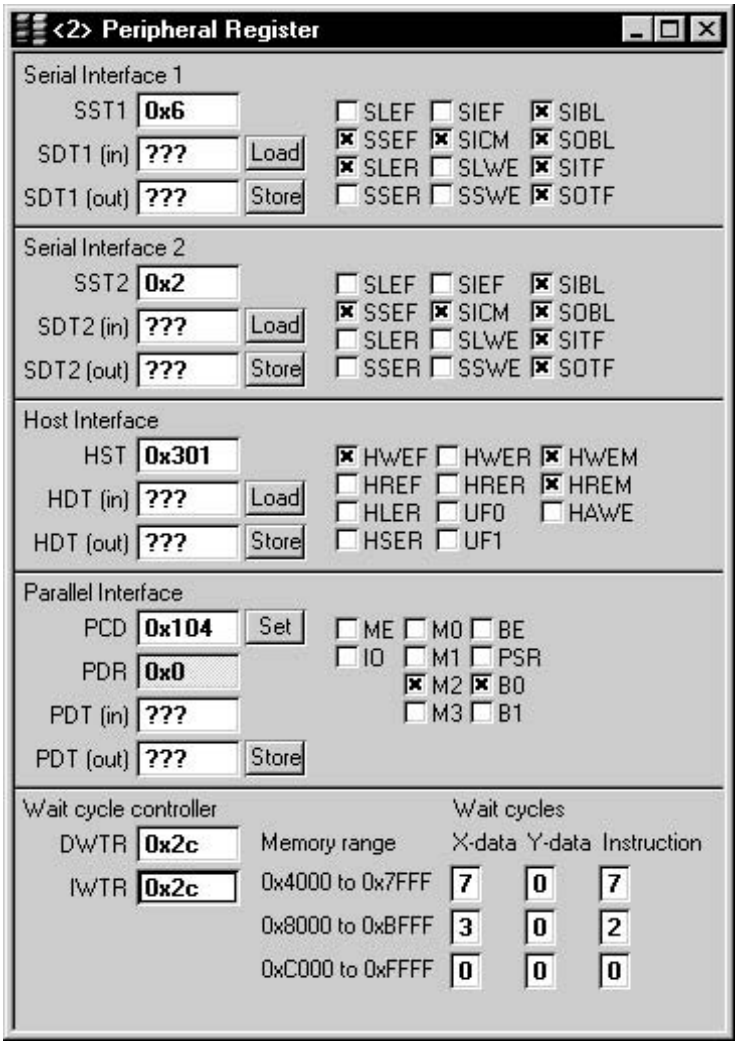

Peripheral Register ウインドウを閉じると，ブロック開閉設定が自動的に保存されます。ディフォールトのブロック開閉設定で Peripheral Register ウインドウを開くには，Shift キーを押したまま Window メニューの Peripheral Register コマンドを実行する必要があります。

保守／廃止

(2) μPD77016 に対するシミュレーションでの Peripheral Register ウインドウ

μPD77016 に対するシミュレーション・セッションでは，Peripheral Register ウインドウは図 5－9 のようになります。

図 5－9　μPD77016 に対するシミュレーションでの Peripheral Register ウインドウ




シミュレーションでの Peripheral Register ウインドウのブロックを次に示します。

保守／廃止

(a) Serial Interface 1

- SST1：シリアル入出力モードを設定する 16 ビットのシリアル・ステータス・レジスタです。そのステータスが示されます。
- SDT1 (in) ：16 ビットのシリアル・データ入力レジスタです。
- SDT1 (out) ：16 ビットのシリアル・データ出力レジスタです。

(b) Serial Interface 2

- SST2：16 ビットのシリアル・ステータス・レジスタです。シリアル入出力モードの設定用です。そのステータスが示されます。
- SDT2 (in)：16 ビットのシリアル・データ入力レジスタです。
- SDT2 (out)：16 ビットのシリアル・データ出力データです。

(c) Host Interface

- HST：16 ビットのホスト・ステータス・レジスタです。DSP とホスト CPU の間のステータス連絡用です。
- HDT (in)：16 ビットのホスト・データ入力レジスタです。
- HDT (out)：16 ビットのホスト・データ出力レジスタです。

(d) Parallel Interface

- PCD：16 ビットのポート・コマンド・レジスタです。汎用入出力ポートの設定（入力／出力ピン）およびビット操作用です。
- PDR： 16 ビットのポート・ディレクション・レジスタです。最後の読み出しアクセス時の PCD レジスタの値を含みます。
- PDT (in)：16 ビットのポート・データ・レジスタです。汎用入力ポートからの入力データの読み出し用です。
- PDT (out)：16 ビットのポート・データ・レジスタです。汎用出力ポートからの入力データの読み出し用です。

(e) Wait cycle controller

- DWTR：16 ビットのウエイト・サイクル・レジスタです。外部メモリ・アクセスで挿入されるウエイト回数を指定します。このレジスタ情報の表示は，選択したメモリ・モデルによって大きく違ってきます。
- IWTR：16 ビットのインストラクション・ウエイト・サイクル・レジスタです。外部インストラクション・メモリ・アクセスでウエイト回数を指定します。このレジスタ情報の表示は，選択したメモリ・モデルによって大きく違ってきます。

(f) Wait cycles

ウエイト・サイクルは，指定されたメモリ範囲の X データ，Y データ，およびインストラクション・メモリに設定できます。設定可能なウエイト・サイクルは，0（ウエイトなし），1，3，7 のいずれかです。設定可能なすべてのウエイト・サイクルが，エディット・バーの選択リストに示されます。ここで示した値以外は入力できません。

保守／廃止

（g）Load ボタン，Store ボタン，Set ボタン

Load ボタンを押すと，データ入力レジスタ（SDT1，SDT2，HDT）は読み出し専用になります。これらのレジスタが読み取り専用の場合は，レジスタの内容を変更できません。

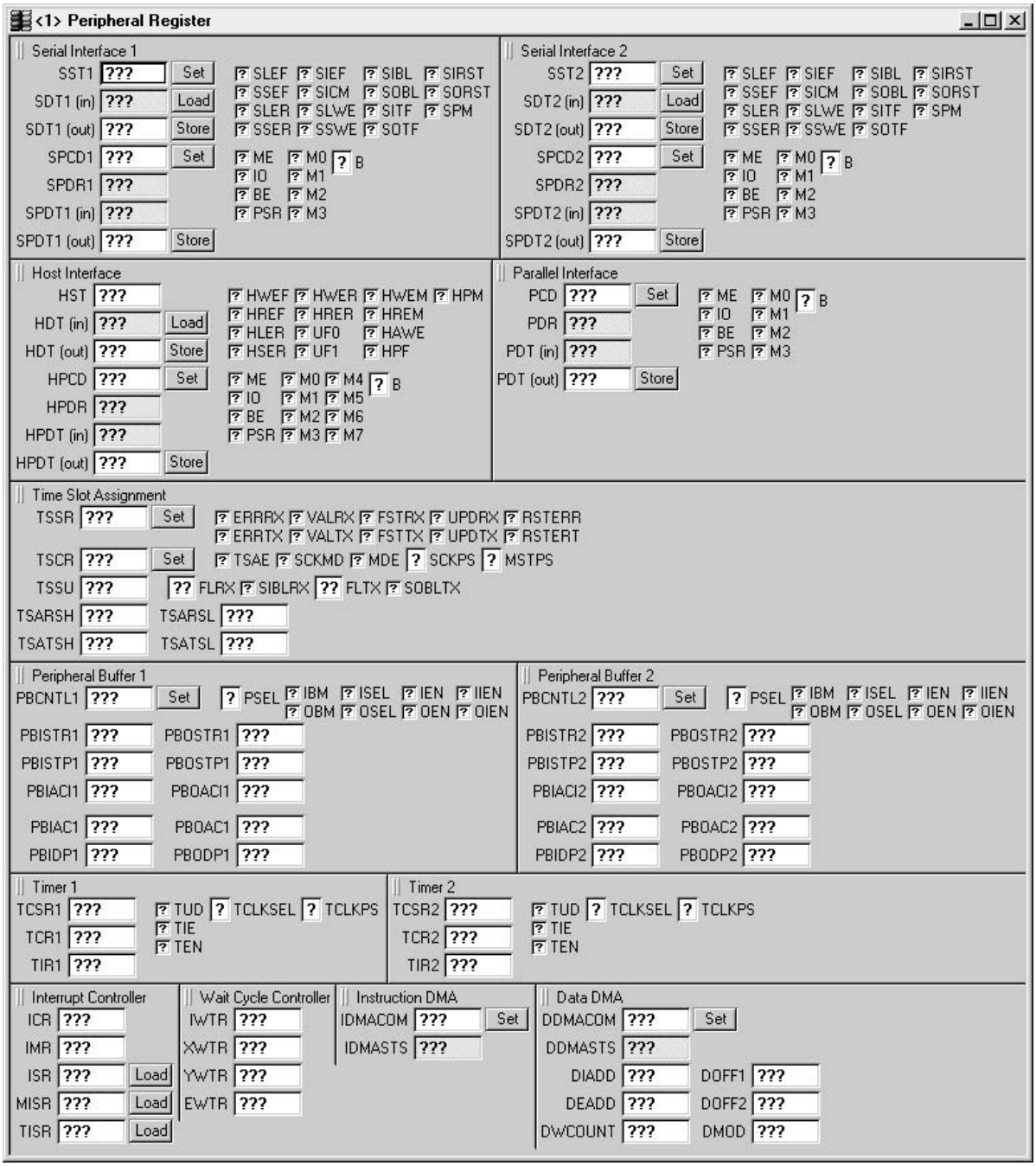
Store ボタンを押すと，データ出力レジスタ（SDT1，SDT2，HDT）は書き込み専用になります。これらのレジスタが書き込み専用の場合だけ，レジスタの内容を変更できます。

Set ボタンを押すと，現在のセル内容が PCD レジスタに書き込まれます。

（3）μPD77116 に対するシミュレーションでの Peripheral Register ウインドウ

μPD77116 に対するシミュレーション・セッションでは，Peripheral Register ウインドウは図 5－10 のようになります。

図 5－10　μPD77116 に対するシミュレーションでの Peripheral Register ウインドウ



μPD77116 に対するシミュレーションでの Peripheral Register ウインドウのブロックを次に示します。

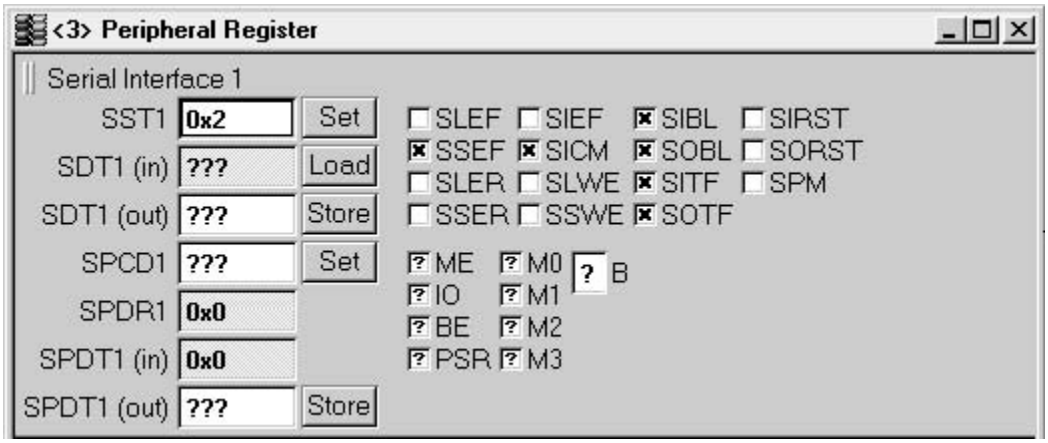
（a）Serial Interface1 と Serial Interface2 ブロック

図 5－11 に示した Serial Interface1 と Serial Interface2 には，次のレジスタ情報が表示されます。

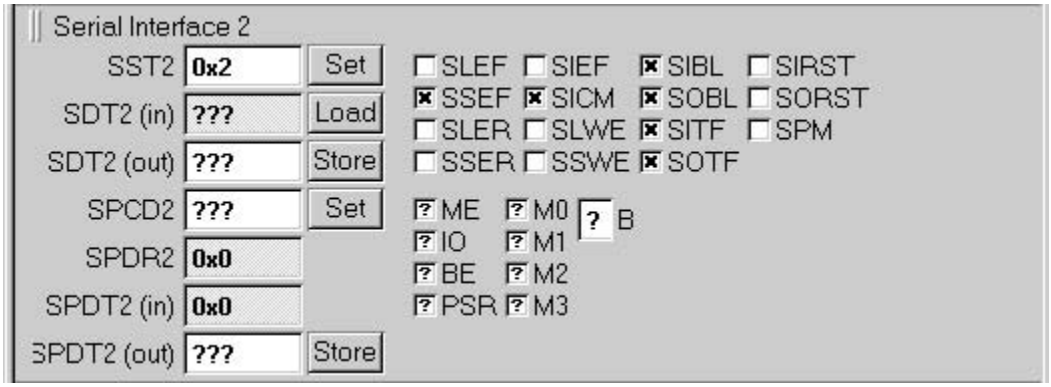
- SST1, SST2：シリアル・ステータス・レジスタです。シリアル入出力モードの設定用です。そのステータスが示されます。
- SDT1 (in), SDT2(in)：シリアル・データ入力レジスタです。
- SDT1 (out), SDT2 (out)：シリアル・データ出力レジスタです。
- SPCD1, SPCD2：シリアル・ポート・コマンド・データ・レジスタです。
- SPDR1, SPDR2：シリアル・ポート・データ・レジスタです。
- SPDT1 (in), SPDT2 (in)：シリアル・ポート・データ入力レジスタです。
- SPDT1 (out), SPDT2 (out)：シリアル・ポート・データ出力レジスタです。

図 5－11　Serial Interface ブロック

（a）Serial Interface 1




（b）Serial Interface 2

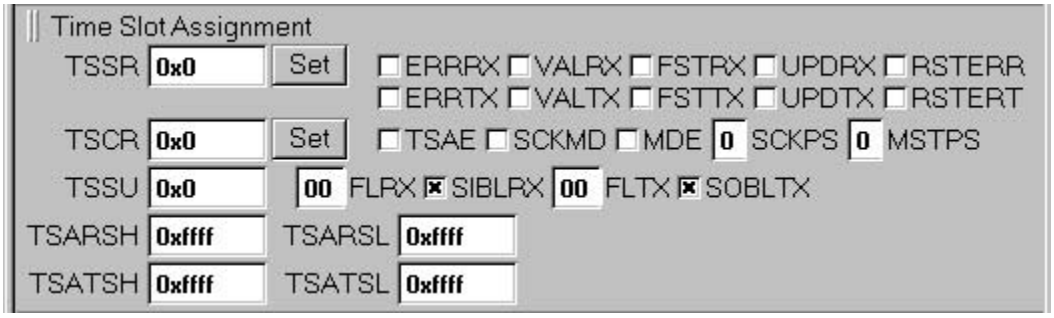


（b）Timer Slot Assignment ブロック

図 5－12 に示した Timer Slot Assignment ブロックには，次のレジスタ情報が表示されます。

- TSSR：TSA ステータス・レジスタです。
- TSCR：TSA コントロール・レジスタです。
- TSSU：TSA セットアップ・レジスタです。
- TSARSH：TSA 受信スロット上位ワード・レジスタです。
- TSATSH：TSA 送信スロット上位ワード・レジスタです。
- TSARSL：TSA 受信スロット下位ワード・レジスタです。
- TSATSL：TSA 送信スロット下位ワード・レジスタです。

図 5－12　Timer Slot Assignment ブロック

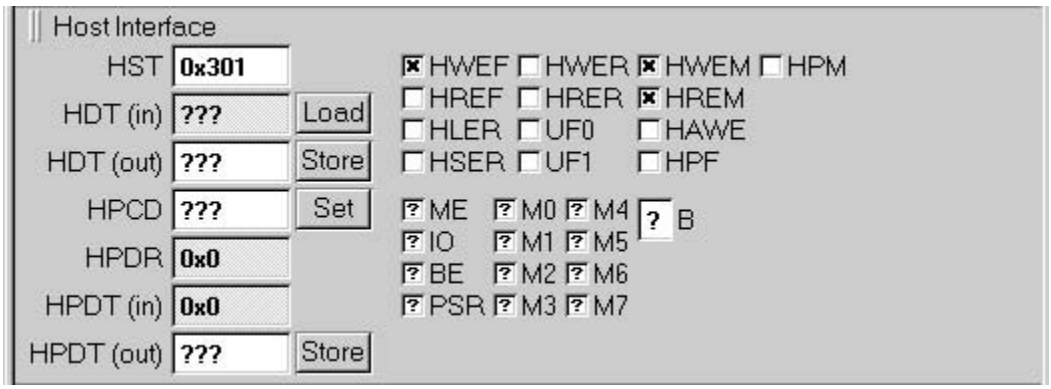



（c）Host Interface ブロック

図 5－13 に示した Host Interface ブロックには，次のレジスタ情報が表示されます。

- HST：ホスト・ステータス・レジスタです。
- HDT (in)：ホスト・データ入力レジスタです。
- HDT (out)：ホスト・データ出力レジスタです。
- HPCD：ホスト・ポート・コマンド・データ・レジスタです。
- HPDR：ホスト・ポート・データ・レジスタです。
- HPDT (in)：ホスト・ポート入力レジスタです。
- HPDT (out)：ホスト・ポート出力レジスタです。

図 5－13　Host Interface ブロック



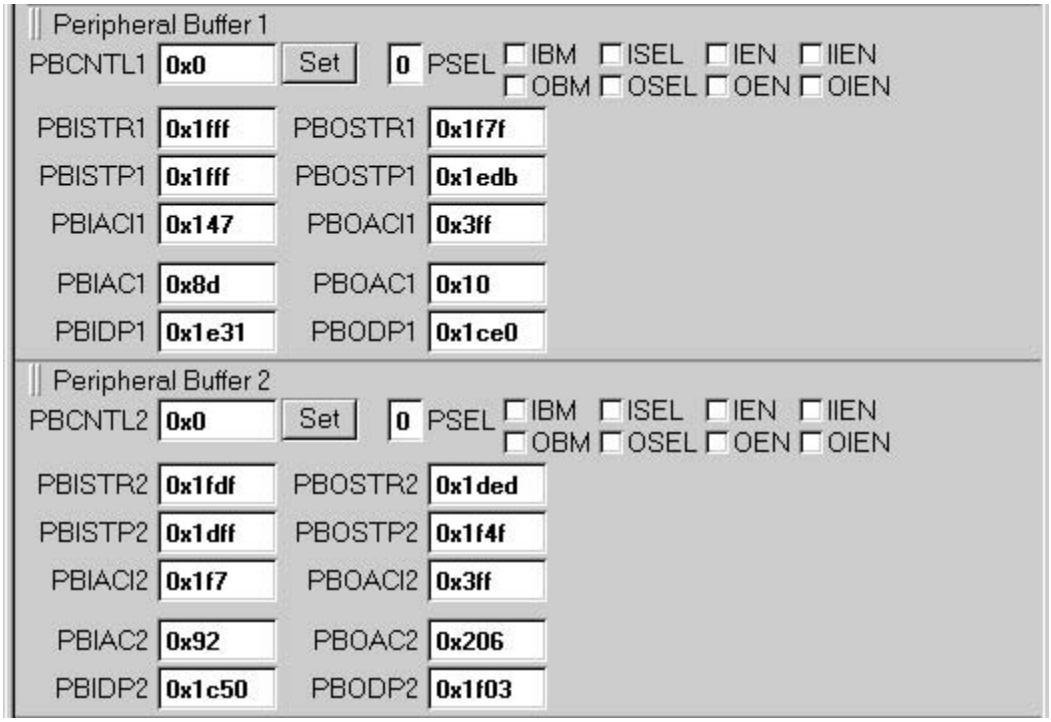
保守／廃止

（d）Peripheral Buffer1 と Peripheral Buffer2 ブロック

図 5－14 に示した Peripheral Buffer ブロックには，次のレジスタ情報が表示されます。

- PBCNTL1, PBCNTL2：ペリフェラル・バッファ・コントロール・レジスタです。
- PBISTR 1, PBISTR2：ペリフェラル・バッファ入力開始アドレス・レジスタです。
- PBISTP1, PBISTP2：ペリフェラル・バッファ入力終了アドレス・レジスタです。
- PBIACI1, PBIACI2：ペリフェラル・バッファ入力アクセス・カウンタ初期値レジスタです。
- PBIAC1, PBIAC2：ペリフェラル・バッファ入力アクセス・カウンタ・レジスタです。
- PBIDP1, PBIDP 2：ペリフェラル・バッファ入力データ・ポインタ・レジスタです。
- PBOSTR1, PBOSTR 2：ペリフェラル・バッファ出力開始アドレス・レジスタです。
- PBOSTP1, PBOSTP 2：ペリフェラル・バッファ出力終了アドレス・レジスタです。
- PBOACI1, PBOACI 2：ペリフェラル・バッファ出力アクセス・カウンタ初期値レジスタです。
- PBOAC1, PBOAC 2：ペリフェラル・バッファ出力アクセス・カウンタ値レジスタです。
- PBODP1, PBODP 2：ペリフェラル・バッファ出力データ・ポインタ・レジスタです。

図 5－14　Peripheral Buffer ブロック



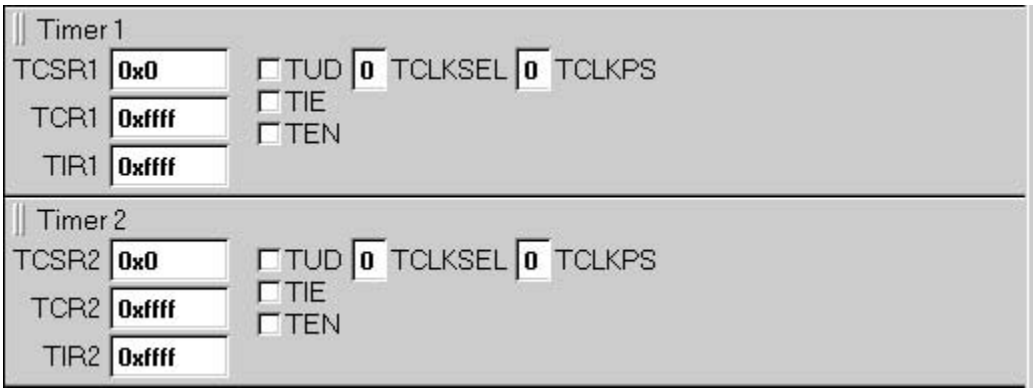
保守／廃止

**(e)** Timer1 **と** Timer2 **ブロック**

**図** 5－15 **に示した** Timer **ブロックには，次のレジスタ情報が表示されます。**

- TCSR1, TCSR2：**タイマ・コントロール・ステータス・レジスタです。**
- TCR1, TCR 2：**タイマ・カウンタ・レジスタです。**
- TIR1, TIR2：**タイマ・イニシャル・レジスタです。**

**図** 5－15　Timer **ブロック**

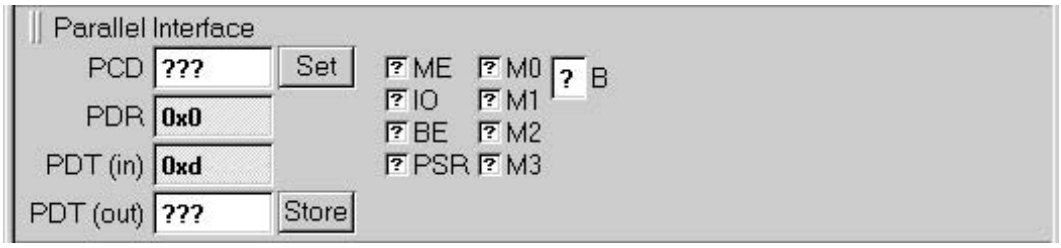



**(f)** Parallel Interface **ブロック**

**図** 5－16 **に示した** Parallel Interface **ブロックには，次のレジスタ情報が表示されます。**

- PCD：**ポート・コマンド・データ・レジスタです。**
- PDR：**ポート・データ・レジスタです。**
- PDT (in) ：**ポート・データ入力レジスタです。**
- PDT (out) ：**ポート・データ出力レジスタです。**

**図** 5－16　Parallel Interface **ブロック**



（g）Interrupt Controller ブロック

図 5－17 に示した Interrupt Controller ブロックには，次のレジスタ情報が表示されます。

- ICR：割り込みコントロール・レジスタです。
- IMR：割り込みマスク・レジスタです。
- ISR：割り込みステータス・レジスタです。
- MISR：メモリ割り込みステータス・レジスタです。
- TISR：タイマ割り込みステータス・レジスタです。

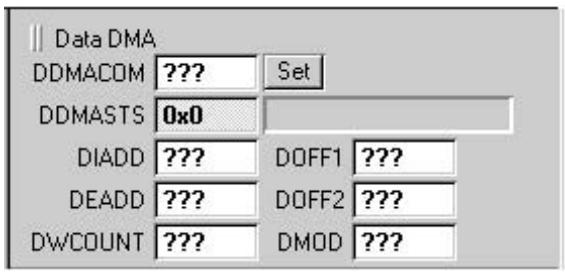
図 5－17　Interrupt Controller ブロック


（h）Data DMA ブロック

図 5－18 に示した Data DMA ブロックには，次のレジスタ情報が表示されます。

- DDMACOM：データ DMA コマンド・レジスタです。
- DDMASTS：データ DMA ステータス・レジスタです。
- Progress Indicator：このフィールドには，DMA 処理の進行状況が表示されます。
- DIADD：データ DMA 内部開始アドレス・レジスタです。
- DEADD：データ DMA 外部開始アドレス・レジスタです。
- DWCOUNT：データ DMA ワード・カウント・レジスタです。
- DOFF1, DOFF2：データ DMA 外部オフセット・アドレス・レジスタです。
- DMOD：データ DMA 外部モジュロ・アドレス・レジスタです。

図 5－18　Data DMA ブロック



保守／廃止

(i) Instruction DMA ブロック

図 5－19 に示した Instruction DMA ブロックには，次のレジスタ情報が表示されます。

- IDMACOM：インストラクション DMA コマンド・レジスタです。
- IDMASTS：インストラクション DMA ステータス・レジスタです。
- Progress Indicator：このフィールドには，DMA 処理の進行状況が表示されます。

図 5－19　Instruction DMA ブロック


(j) PLL ブロック

図 5－20 に示した PLL ブロックには，次のレジスタ情報が表示されます。

- CLKCNTL：クロック・コントロール・レジスタです。

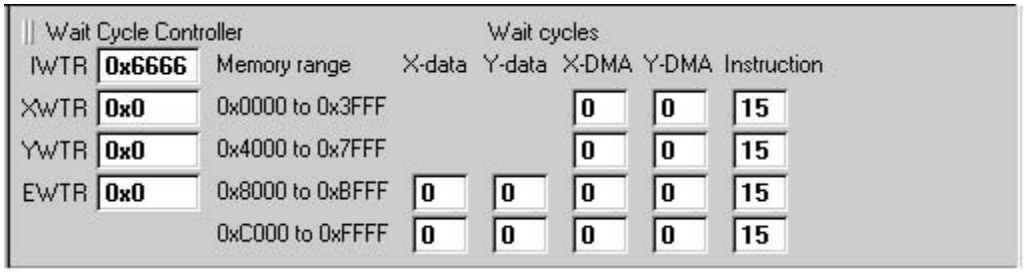
図 5－20　PLL ブロック


(k) Wait Cycle Controller ブロック

図 5－21 に示した Wait Cycle Controller ブロックには，次のレジスタ情報が表示されます。

- IWTR：インストラクション・メモリ・ウエイト・サイクル・コントロール・レジスタです。
- XWTR：X-DMA メモリ・ウエイト・サイクル・コントロール・レジスタです。
- YWTR：Y-DMA メモリ・ウエイト・サイクル・コントロール・レジスタです。
- EWTR：外部直接アクセス・データ・メモリ・ウエイト・サイクル・コントロール・レジスタです。
- Memory range：メモリ範囲が表示されます。
- Wait cycles：ウエイト・サイクル回数が表形式で表示されます。

図 5－21　Wait Cycle Controller ブロック



保守／廃止

## 5.3　Breakpoint ウインドウ

Breakpoint ウインドウには，定義されているすべてのブレークポイントが表示されます。

図 5－22 に Breakpoint ウインドウを示します。

図 5－22　Breakpoint ウインドウ

<1> Breakpoints

| Addr | Condition | Action | Group | Enable | Pass | Notes |
|---|---|---|---|---|---|---|
| Global | Error bits set | Break | 1 | Yes | 1 | |
| Global | Read access "0x200:x" @0x0200:X | Log "r1"<br>Break | 2 | Yes | 1 | |
| Global | Elapsed time "200ns", in 200ns | Break | 3 | Yes | 1 | |
| Global | Expression true "r0==2" | Disable group 3<br>Break | 4 | Yes | 1 | |
| Global | Always | Break | 6 | Yes | 1 | Cannot |
| 0x0408 | Always | Break | 5 | Yes | 1 | |

### 5.3.1　Breakpoint ウインドウのカラム

- Addr

  Addr カラムには，ブレークポイントが設定されているアドレスが表示されます。次のアドレス情報があります。

  - インストラクション・メモリ内のアドレス

    指定のアドレスが実行されると，ブレークポイント条件がチェックされます。

  - Global

    ブレークポイント条件は，実行アドレスとは関係なくチェックされます。たとえば，データ・メモリ・アクセスのブレークポイントなどです。

  - タイミング・ファイル行

    タイミング・ファイルがブレークポイント行に到達すると，ブレークポイント条件がチェックされます。タイミング・ファイル・ブレークポイントは明示的でも（Breakpoint ウインドウまたは Timing File ウインドウで設定）非明示的でも（タイミング・ファイルの Break 命令で設定）どちらでもかまいません。非明示的なタイミング・ファイル・ブレークポイントは，Breakpoint ウインドウから削除できません。非明示的ブレークポイントの背景色は，ウインドウ色より明るくなります。非明示的ブレークポイントは，対応するタイミング・ファイルが閉じられたときに削除されます。タイミング・ファイル・ブレークポイントには，指定条件や指定動作がありません。

保守／廃止

• Condition

ブレークポイントの起動時に満足されなければならない条件を指定します。ブレークポイントが起動した場合，条件により次のようになります。

パス・カウント = 1：ブレークが実行されます。
パス・カウント > 1：パス・カウンタが 1 減分され，ブレークは実行されません。

**注意　1 つのブレークポイントに複数のブレーク条件を指定できますが，その場合はすべての条件が満足されなければなりません。**

• Action

パス・カウント = 1 によりブレークポイントが起動したときの開始動作を指定します。

**注意　1 つのブレークポイントに複数のブレーク動作を指定できますが，その場合，ブレークポイント条件が真になるときにすべての動作が実行されます。**

• Group

ブレークポイントが属すグループを指定します。グループ番号は，Breakpoint メニューの Add Action | Enable Group...コマンドまたは Add Action | Disable Group...コマンドでブレークポイントを有効または無効にする場合に使用されます。

• Enable

次のブレークポイント・ステータスが表示されます。

Yes：ブレークポイントがアクティブです。
No：ブレークポイントが非アクティブです。どの条件にも反応せず，パス・カウンタも減分されません。

• Pass

ブレーク動作の実行前に，アドレスとブレーク条件が何回満たされなければならないかを指定します。

• Notes

指定されたブレークポイントに関するさらに詳しい情報が表示されます。Notes は確認専用のカラムです。

保守／廃止

### 5.3.2　ブレークポイント

ブレークポイントで，プログラム実行を一時的に停止するタイミングを指定します。停止時点で，レジスタの値を調べたり，ブレーク動作を実行したりできます。ブレークポイントとして次のものを指定できます。

- インストラクション・メモリ内のアドレス
  指定のアドレスが実行されると，ブレークポイント条件がチェックされます。この条件が真になると，シミュレーションが中断されます。
- Global
  ブレークポイント条件は,実行アドレスと関係なくチェックされます。この条件が真になると,シミュレーションが中断されます。
- 明示的または非明示的なタイミング・ファイル・ブレークポイント
  タイミング・ファイル実行が指定のブレークポイントに到達すると，ブレークポイント条件がチェックされます。この条件が真になると，タイミング・ファイル実行が中断されます。

### 5.3.3　ブレークポイントの追加

ブレークポイントの追加方法は次のとおりです。

- Breakpoint メニューコマンドから希望のブレークポイントのタイプを選択します。
- Memory メニューまたは Module メニューの Toggle Breakpoint コマンドを選択します。このコマンドは，Instruction Memory ウインドウでの現在のアンカ位置または Module ウインドウでのカーソル位置に実行ブレークポイントを設定します。
  Module ウインドウのソース行でブレークポイントを設定した場合，ブレークポイントは関連するインストラクション・メモリ・コードの最初のアドレスに自動的に設定されます。
  インストラクション・メモリ・ブロックでブレークポイントを設定した場合，ブレークポイントはこのブロックに関係付けられたソース行に自動的に実行されます。
- Breakpoint ウインドウがアクティブなときに Insert キーを押すと，実行ブレークポイントが生成され，アドレス値の入力が要求されます。これが，明示的タイミング・ファイル・ブレークポイントです。
- Timing File ウインドウでブレークポイントの有効／無効を切り替えるか，タイミング・ファイル・コードの Break 命令を指定します。これが，非明示的タイミング・ファイル・ブレークポイントです。

### 5.3.4　ブレークポイントの削除

ブレークポイントを削除するには，削除対象のブレークポイント行にアンカを移動しなければなりません。Edit メニューから Delete コマンドを選択するか，キーボードの Delete キーを押します。削除するかどうかを確認するダイアログ・ボックスが表示されます。

**注意　アンカがブレークポイントの動作または条件の位置にあり，複数の動作または条件のリストがある場合，それらの項目のすべてが削除されます。ブレークポイントの動作または条件がすべて削除された場合は，ブレークポイントすべてが削除されます。**
**タイミング・ファイルの Break コマンドで設定した非明示的タイミング・ファイル・ブレークポイントは，Breakpoint ウインドウから削除できません。**

### 5.3.5　ブレークポイントの保存

Breakpoint ウインドウで定義したブレークポイントは，Tools メニューの Options...コマンドで保存または復元できます。

## 5.4　Watch ウインドウ

Watch ウインドウには，監視する必要がある重要な値を持つ変数と式のリストが表示されます。グラフィック・タイミング出力が Watch ウインドウに統合されています。Watch ウインドウの Show Signals コマンドでこの出力をオンにすると，Watch ウインドウが Watch ブロックと Diagram ブロックに分割されます。

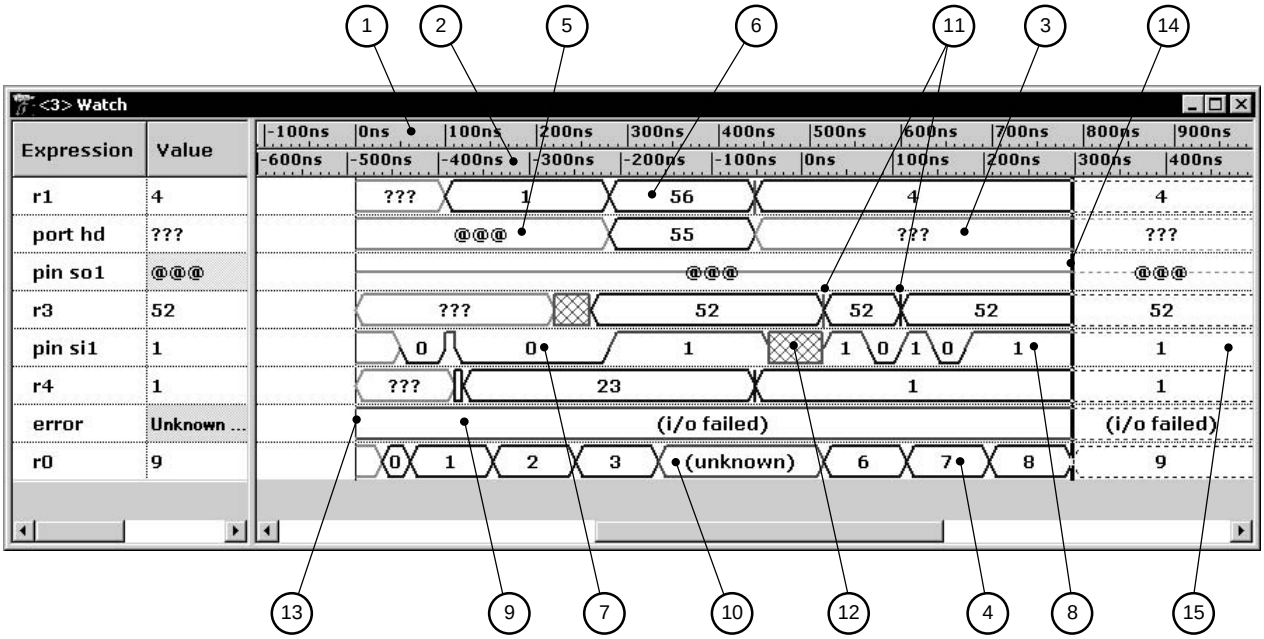
左側の Watch ブロックには，Expression と Value の 2 つのカラムがあります。Expression カラムは，式の追加，削除，修正に使用します。Value カラムは，式の値の表示と修正に使用します。

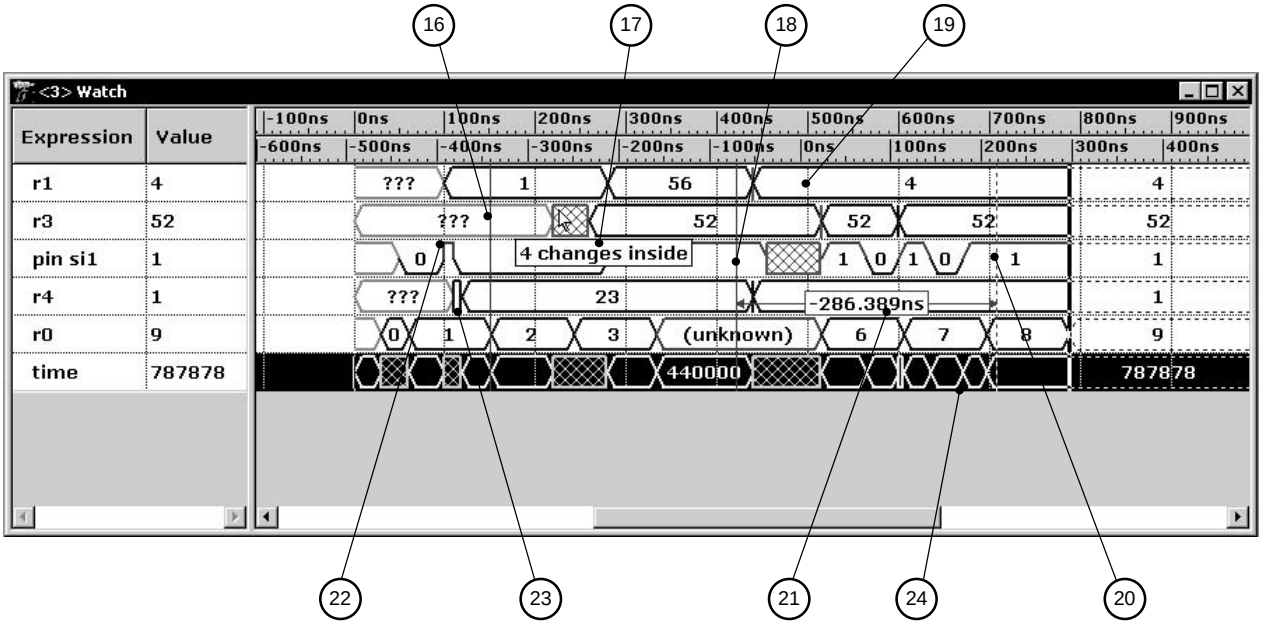
Diagram ブロックには，シグナルの経過状態が表示されます。Watch ブロックに追加されたそれぞれの式に対応するシグナル図が，Diagram ブロックに表示されます。シグナル図には，現在のシグナル値の他に，シミュレーション中に生じたシグナル状態の変化も示されます。Watch メニューでは，グラフィック・タイミング出力に関連するコマンドを利用できます。

図 5－23 に Watch ウインドウを示します。

保守／廃止

**図 5ー23　Watch ウインドウ**




1：**絶対タイム・スケール**
2：**相対タイム・スケール**
3：**未定義整数シグナル**
4：**定義整数シグナル**
5：**ハイ・インピーダンス整数シグナル**
6：**ハイ・インピーダンス** 2 **進シグナル**
7：2 **進シグナル，ロウ・レベル**
8：2 **進シグナル，ハイ・レベル**
9：**未知シンボリック名**
10：**未記録シグナル**
11：**ゼロ・タイム変更**
12：**エイリアス・シグナル**
13：**記録開始**
14：**実際の時間**
15：**将来の項目**
16：**タイム・マーカ**
17：**シグナル・ツール・チップ**
18：**タイム・カーソル**
19：**グリッド線**
20：**測定開始時間マーカ**
21：**測定ツール**
22：**短い** 2 **進シグナル**
23：**短い整数シグナル**
24：**選択シグナル行**

保守／廃止

### 5.4.1　Watch ブロックのカラム

（1）Expression

Expression カラムには，ユーザが指定した式とその評価値が表示されます。各式のシグナル状態は，Diagram ブロックに表示されます。シグナルの追加は，Expression ブロックで行う必要があります。追加したシグナルは，Diagram ブロックに表示されます。有効な watch 式は次のとおりです。

- シンボル名とレーベル名：シンボル名とレーベル名は大文字と小文字の区別があり，ソース・テキストに書き込まれているとおりに入力しなければなりません。レーベル名は，次の構文で参照されます。

  *symbolname.labelname*

  シンボルまたはレーベルが評価されると，Value カラムにシンボル／レーベル開始アドレスが表示されます。
- HSM77016 変数（レジスタ，ポートなど）
- タイミング・ファイル変数
- 数値
- レーベル名とシンボル名，HSM77016 演算子の組み合わされた（タイミング・ファイル）変数と数値

**備考**　HSM77016 変数，演算子，数値形式構文については，第 11 章　レファレンスを参照してください。

（2）Value

Value カラムには式の現在の値が表示されます。値が読み出し専用でない場合は，値の編集や変更ができます。Expression カラムの式が読み出し専用の場合は，関連している Value カラムの値も読み出し専用となり，ユーザが変更することはできません。各値にそれぞれ別の表示形式を選択できます。

### 5.4.2　Diagram ブロックの構成要素

Diagram ブロックには，各式のシグナル状態および以前と現在のステータスのシグナル値が表示されます。

さらに，以前のシグナル状態変化（縦棒の右）とそれに対応するシグナルの現在値（縦棒の左）も表示されます。ただし，HSM77016 の起動前後には現在のシグナル値しか表示されません。シグナル状態の履歴は，μPD77016 命令（step，trace，animate，run）の実行時だけ生成されます。step，trace，または animate の実行中に，Diagram ブロックの表示が更新されます。run コマンドでシミュレーションを開始した場合，このブロックの表示はシミュレーションの終了後に更新されます。ユーザ定義の表示色をシグナル図に指定できます。式の表示形式は，対応するシグナルに適用されます。

シグナルが小さすぎて，発生した変化すべてを表示できない場合は，Diagram ブロックに網かけ領域が表示されます。変化数情報は，ツール・チップから読み取ることができます。

step または trace 以外で複数回レジスタの値を変更する場合は，ゼロ・タイム変更が複数になります。この動作は，Watch ウインドウの Diagram ブロックで網かけ領域と同じ色の行に示されます。

記録される領域は，バッファ・サイズ調整によって変わります。Option ダイアログ・ボックスの Recording タブで，バッファ・サイズ設定を調整できます。詳細については，図 6－35 Recording タブ（Options ダイアログ・ボックス）を参照してください。

記録される領域は，縦棒で囲まれています。Diagram ブロックのスクロール領域は，この縦棒で定義されます。Watch ウインドウのビューをこの領域の外側に移動することはできません。zoom コマンドを使用しても移動できません。

保守／廃止

**注意**　Windows メニューの watch コマンドで Watch ウインドウをオンにすると，表示されている式が連続して記録されます。Watch ウインドウをオフに切り替えると，式の連続記録が停止します。
Diagram ブロックが閉じている場合に命令を実行すると，記録されていない領域が表示されます。

### 5.4.3　タイム・スケール

シグナル期間とシグナル距離を表示するために，Diagram ブロックのタイム・スケールでは，シグナルの光表示とタイミング動作が連動するようになっています。そのため，それぞれ別々にオンとオフを切り替えることができる 2 つのタイム・スケールが実装されています。

ズーム倍率とズーム・サイズの設定によって，Diagram ブロックに問題なく収まるように両方のタイム・スケールを調整できます。

絶対タイム・スケールには，HSM77016 の時間変数が表示されます。このスケールは移動できません。両方のタイム・スケールを表示した場合，絶対タイム・スケールが常に上に表示されます。

相対タイム・スケールを左クリックして目的の位置に移動すれば，相対タイム・スケールをスクロールできます。ゼロ・ポイントになるタイム・スケール・ノッチをダブルクリックすれば，タイム・スケールのゼロ・ポイントをどの位置にでも設定できます。マウス・ボタンを押したままにすると，選択したポイントをどの位置にでも移動できます。

### 5.4.4　シグナルのタイプ

**（1）整数シグナル**

整数シグナルは，ポート，レジスタ，整数タイミング・ファイル変数および整数式です。これらのシグナルには次の状態があります。

- 定義値
- 不定義値
- ハイ・インピーダンス

**（2）2 進シグナル**

2 進シグナルは，端子，2 進タイミング・ファイル変数および 2 進式です。これらのシグナルには次の状態があります。

- ハイ・レベル
- ロウ・レベル
- ハイ・インピーダンス
- 不定義値

### 5.4.5　Watch 式の入力

Watch 式を入力する方法は次のいずれかです。

- Watch メニューの Add Watch コマンドを実行します。
- キーボードの Insert キーを押します。
- Evaluate and Modify Variable ダイアログ・ボックスの Add to Watch Window ボタンを押します。
- Symbol Table ダイアログ・ボックスの Add to Watch Window ボタンを押します。
- クリップ・ボード

**注意**　Watch 式を追加するには，Watch ブロックがアクティブになっていなければなりません。

## 5.5　Log ウインドウ

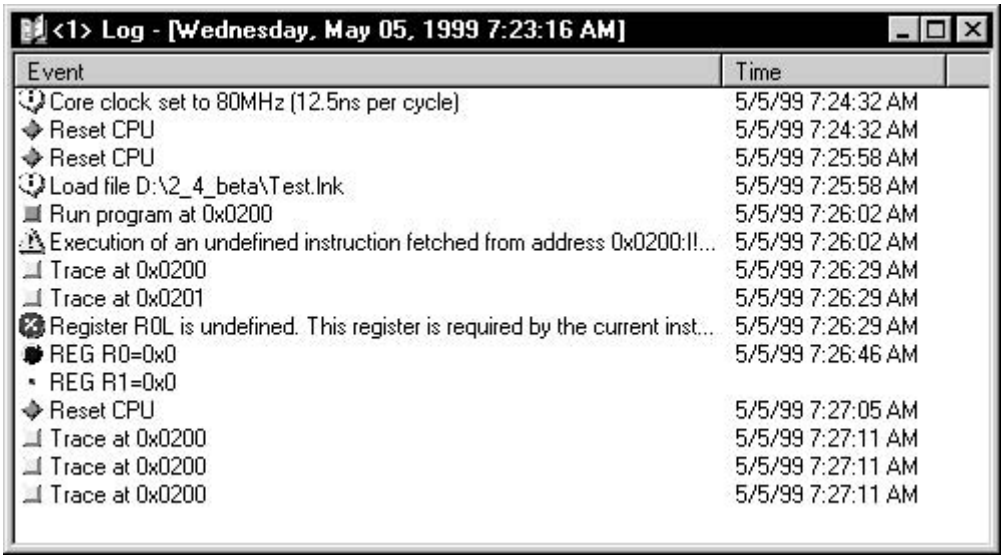
Log ウインドウには，ログ・セッションで要約されたシミュレータ・イベントが記録されます。ユーザが特定のセッション名を指定しないかぎり，ディフォールトのセッション名が使用されます。ログ・セッションはログ・ファイルに保存されるか，テキスト・ファイルにエクスポートされます。ログ・ファイルを指定しないと，HSM77016 アプリケーションの終了時にログ・セッション・データが拒否されます。指定したログ・ファイルの名前は，現在のログ・セッションの名前（[　] で囲まれます）とともに Log ウインドウのタイトル・バーに表示されます。

ログは常に有効になっており，ユーザが無効にすることはできません。次のイベント情報がログ・データとして自動的に生成されます。

- シミュレーション動作
- シミュレーション・メッセージ
- タイミング・ファイル・イベント
- ブレークポイント通過
- ログ・バッファに転送される，ユーザが選択した HSM77016 のメモリ・ウインドウとレジスタ・ウインドウのデータ
- シミュレーション中に表示されたエラー・メッセージ（すべてログ・バッファに書き込まれます）

図 5－24 に Log ウインドウの例を示します。

図 5ー24　Log ウインドウ




Log ウインドウには，データ・カラムとカラム・ヘッダがあります。Event カラムには記録されたログ・イベントが表示され，Time カラムにはログ・イベントが発生した日付と時刻が表示されます。

Log ウインドウの内容は，Log メニューの Clear Session コマンドで削除できます。Log ウインドウの内容をテキスト・ファイルに保存するには，File メニューの Export...コマンドを選択します。Log ウインドウのテキスト表示色は，Tools メニューの Options...コマンドで変更できます。

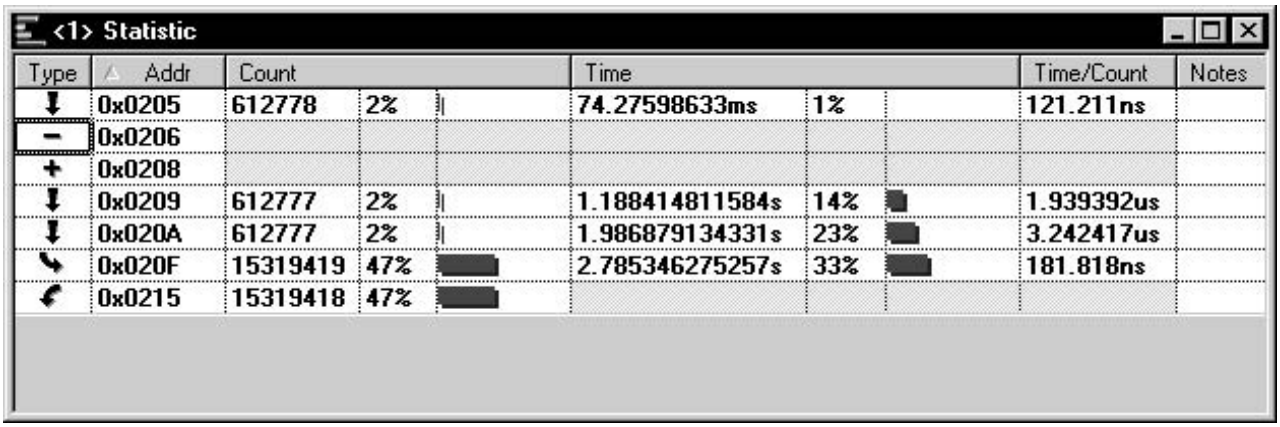
保守／廃止

## 5.6　Statistic ウインドウ

Statistic ウインドウには，プロファイリング・セッションで収集されたデータの概要が表示されます。

図 5－25 に Statistic ウインドウの例を示します。

図 5－25　Statistic ウインドウ




### 5.6.1　Statistic ウインドウのカラム

（1）Type

このカラムには，マーカ・タイプのシンボルが表示されます。

（2）Addr

このカラムには，マーカが設定されているアドレス，またはソース・モジュールの名前と行番号が表示されます。

（3）Count

このカラムには，マーカ上をプログラムが通過した回数と，全マーカについての全通過回数に対する比率がパーセントで表示されます。パーセント値が 1 より小さく 0 より大きい場合は，<1%と表示されます。Start マーカと Stop マーカについては，これらのマーカは通過数の計数対象になっていないことを示すために明るい色で表示されます。

（4）Time

このカラムには，マーカ間の実行時間と，全実行時間に対する比率がパーセントで表示されます。パーセント値が 1 より小さく 0 より大きい場合は，<1%と表示されます。Start マーカと Stop マーカについては，これらのマーカが実行時間の計数対象になっていないことを示すために明るい色で表示されます。

（5）Time/Count

このカラムには，通過回数 1 回当たりの実行時間が表示されます。Exit マーカ，Start マーカ，Stop マーカについては，これらのマーカが実行時間の計数対象になっておらず，通過回数 1 回当たりの実行時間を計算できないため，明るい色で表示されます。

**注意**　Notes **カラムはフィードバック情報を表示するためだけのものです。このカラムには，指定されたマーカに関する補足情報が表示されます。**

### 5.6.2　Statistic ウインドウのデータ配置

Statistic **ウインドウには，必要に応じて次のデータ単位で表示することができます。**

- Addr **の値**
- Count **の値**
- Time **の値**
- Time/Count **の値**

**カラムのデータを並べ替えるには，カラム・ヘッダをクリックします。同じカラムを続けてクリックすると，昇順と降順（これらはカラム・ヘッダ内の上向き矢印と下向き矢印で表示されます）が交互に切り替わります。**Statistic **メニューで対応するソート・コマンドを選択する方法でも並べ替えができます。**

### 5.6.3　マーカの設定と削除

Statistic **ウインドウにマーカを設定するには，**Statistic **メニューの設定コマンドを選択します。**

Statistic **ウインドウのマーカを削除する場合は，アンカがそのマーカの置かれている行のどこかになければなりません。**Edit **メニューの** delete **コマンドを選択するか，キーボードの** Delete **キーを押して削除します。その際，削除するかどうかを確認するダイアログ・ボックスが表示されます。**

## 5.7　Text ウインドウ

Text **ウインドウはフィードバック情報を表示するためだけものです。表示されたデータは変更できませんが，選択してクリップ・ボードにコピーすることができます。**Text **ウインドウには，**View **ウインドウ，**Module **ウインドウと** Timing File **ウインドウがあります。これらすべてのウインドウに，行番号を表示する機能と指定行へ移動する機能があります。移動するには，ヘッダ行の** Line **カラムをクリックします。**

**ほかの表示設定の場合と同様に，**Tools **メニューの** Options...**コマンドで** Text **ウインドウの表示色を変更できます。**

### 5.7.1　View ウインドウ

**一般に，**View **ウインドウはプログラムのソース・ファイルやデータ・ファイルを確認するために使用します。このウインドウにはどんなテキスト・ファイルでもロードできます。**

保守／廃止

## 5.7.2　Module ウインドウ

Module ウインドウは，ソース・レベル・シミュレーション時にソース・ファイルを表示するためだけに使用します。ソース・レベル・シミュレーション時に，次の動作を実行できます。

- アドレス・ポインタの監視

  Line（番号）カラムの右のカラムにアドレス・ポインタが表示されます。アドレス・ポインタ式は，Module メニューの Pointer...コマンドで定義されます。アドレス・ポインタは，指定されたポインタ式が配置されているアドレスを示します。アドレス・ポインタは，どのアドレスにも移動できます。移動するには，ポインタをクリックしてドラッグします。カーソル形状がポインタ・ドラッグ・カーソルに変わります。ポインタ・ドラッグ領域は行番号カラムの右のカラムです。

- ブレークポイントの切り替え

  行番号が表示されているカラムをクリックすれば，実行ブレークポイントの設定と削除ができます。カーソルがこのカラムに入ると，カーソル形状がブレークポイント・カーソルに変わります。Module メニューの Toggle Breakpoint コマンドまたはツール・バーの Toggle Breakpoint ボタンでブレークポイントを切り替えることもできます。

  ブレークポイントをソース行に設定すると，該当する命令コードの最初のアドレスにブレークポイントが自動的に設定されます。ソース・ブレークポイントを削除すると，該当する命令コードにあるブレークポイントもすべて削除されます。

### (1) プロファイリング

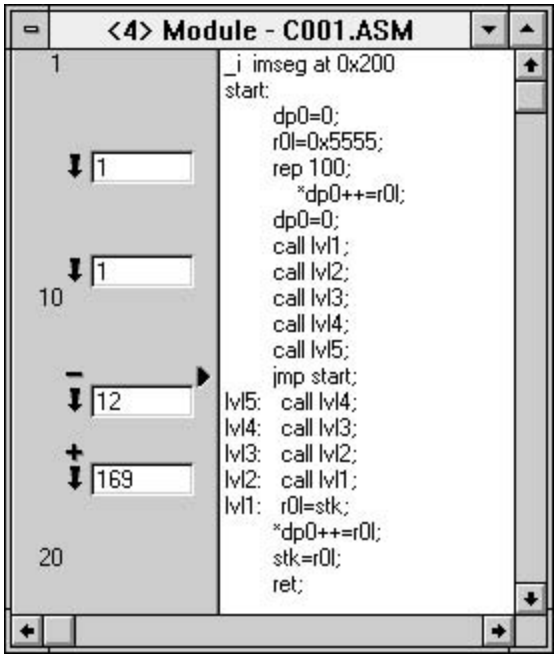
Module ウインドウのプロファイリングには，次の手順が含まれています。

- プロファイリング・マーカの設定と削除
- プロファイリング・データの検査

図 5－26 に，プロファイリング・マーカが含まれている Module ウインドウを示します。

保守／廃止

図 5－26　プロファイリング・マーカが含まれている Module ウインドウ




### (2) プロファイリング・マーカの設定と解除

Module ウインドウでプロファイリング・マーカを設定する方法は次のとおりです。

(a) カーソルを目的の行番号の右に移動します。カーソル形状がマーカ・カーソルに変わってからクリックすると，プロファイリング・マーカが設定されます。マーカのタイプを変更するときや設定／解除を切り替えるときには，マーカ・カラムを次々にクリックしていきます。

(b) 目的の Marker Toggle コマンドを Module メニューから選択するか，ツール・バーから Toggle Marker ボタンを選択します。マーカのタイプを変更するときや設定／解除を切り替えるときには，マーカ・カラムを次々にクリックしていきます。

マーカ・カラムは行番号とポインタ・カラムの間に挿入されます。マーカ・カラムのセルは表示専用であり，つなぎ合わせることはできません。
左端のマーカ・カラムには，次のマーカ・シンボルが表示されます。

- ：Label マーカ
- ：Entry マーカ
- ：Exit マーカ
- ：Start マーカ
- ：Stop マーカ

**注意**　1 つのアドレスに設定できるマーカは 1 つだけです。ブレークポイントとプロファイリング・マーカを同じアドレスに設定することはできません。ブレークポイントが設定されているアドレスにマーカが設定された場合，実行ブレークポイントをプロファイリング・マーカに置き換えるようダイアログ・ボックスにメッセージが表示されます。

**（3）プロファイリングの開始**

**実行が** Start **マーカまたは** Label **マーカに達すると，プロファイリングが開始されます。マーカの** Count **カラムには，プログラムの実行中にマーカを通過した回数が表示されます。**

## 5.8　Timing File **ウインドウ**

Timing File **ウインドウは，タイミング・ファイルを表示するためだけのものです。このウインドウのタイトル・バーには，ロードされたタイミング・ファイルとそのステータスが表示されます。タイミング・ファイルのステータスには，次のものがあります。**

- Break: BREAK **命令によってタイミング・ファイルがシミュレーションを停止しました。**
- End: END **命令によってタイミング・ファイルの実行が停止しました。**
- Error: **エラー条件によってタイミング・ファイルの実行が停止しました。**
- Ready: **タイミング・ファイルの実行中です。**Timing File **ウインドウの行番号カラムの右の実行ポインタは，現在の実行行を示します。**
- Suspend: Timing File **メニューの** Suspend **コマンドによってタイミング・ファイルの実行が中断されました。**
- Wait: WAIT **命令によって，タイミング・ファイルがイベントを待っています。**

### 5.8.1　**タイミング・ファイルの実行ポインタ**

Module **ウインドウと同様に，**Timing File **ウインドウには実行ポインタが表示されます。ユーザがこのポインタを変更することはできません。**

### 5.8.2　**タイミング・ファイルのブレークポイント**

**タイミング・ファイルのブレークポイントは，非明示的と明示的の両方が可能です。非明示的ブレークポイントは，タイミング・ファイルの** BREAK **命令で設定します。明示的ブレークポイントは，**Timing File **ウインドウの** Line **カラムの目的の行をクリックして設定します。明示的ブレークポイントは，**BREAK **命令以外のタイミング・ファイル命令を含む任意の行に設定できます。タイミング・ファイルのブレークポイントは，**Breakpoint **ウインドウに一覧表示されます。シミュレーション・ブレークポイントと同様に，タイミング・ファイルのブレークポイントではどのような条件と動作でも指定できます。ただし，非明示的ブレークポイントは** Breakpoint **ウインドウから削除することはできません。対応するタイミング・ファイルが閉じられると，非明示的ブレークポイントが削除されます。タイミング・ファイルの詳細については，**4.9　**タイミング・ファイル****を参照してください。**

〔メ　　モ〕

保守／廃止

★ # 第6章　メニューとメニューコマンド

## 6.1　File メニュー

File メニューには，ファイル（セッション・イメージ・ファイル，リンク・ファイル，16 進ファイル，データ・ファイル，タイミング・ファイル，テキスト・ファイル）をロードまたは保存するコマンドと終了するコマンドが含まれています。さらに，このメニューのファイル・リストによって，最近使用されたファイルに素早くアクセスすることができます。

### 6.1.1　New

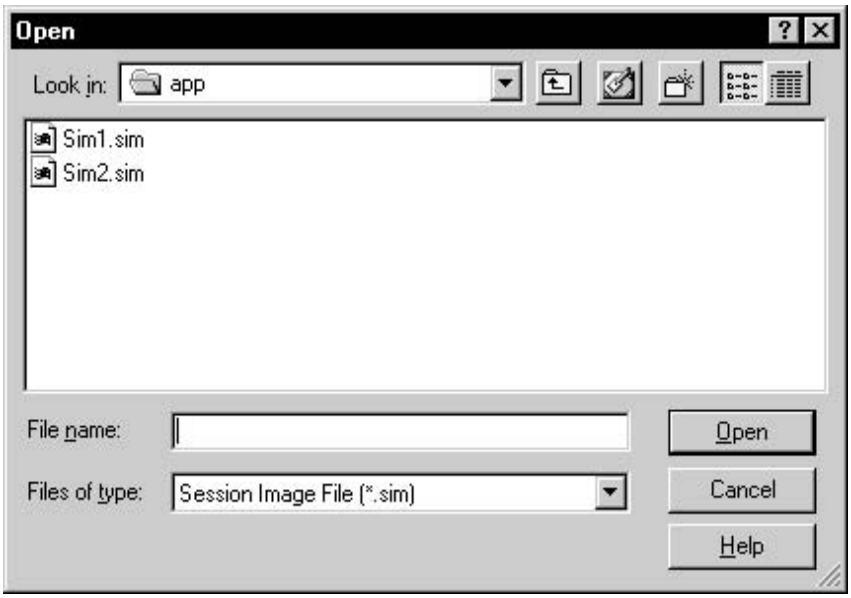
New コマンドは，現在のセッション・イメージ・ファイルやリンク・ファイルを削除することにより，新しいセッション・イメージ・ファイルを開き，HSM77016 のファイルをリンクします。Memory ウインドウと Register ウインドウの内容は定義されていません。

### 6.1.2　Open...

Open...コマンドを選択すると，Open ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，セッション・イメージ・ファイル，リンク・ファイル，タイミング・ファイルをロードしたり，特定の Text ウインドウに任意のテキスト・ファイル（アセンブラ・ソース・ファイルなど）を表示するのに使用します。

図 6－1 は Open ダイアログ・ボックスです。

図 6－1　Open ダイアログ・ボックス




リンク・ファイルのロード時に対象の DSP が自動的にリセットされます。

ロードしたリンク・ファイルにディバグ情報が含まれている場合は，インストラクション・ポインタの現在の位置を示すファイルが，ソース・レベル・シミュレーション用の Module ウインドウに表示されます。ソース・ファイルが表示されていない場合は，手動でソース・ファイルをブラウズする Browse ダイアログ・ボックスが表示

されます。

ファイルを開くには，File name フィールドにファイル名とパスを入力するか，ディレクトリまたはドライブからファイルを選択します。Files of type リストには，利用できるファイル拡張子が表示されます。必要なファイル拡張子に設定してください。アクティブなリンク・ファイルまたはセッション・イメージ・ファイルの名前が，HSM77016 のタイトル・バーに表示されます。

### 6.1.3　Save

Save コマンドは，HSM77016 のタイトル・バーに表示されている名前で現在のセッション・イメージ・ファイルを保存します。セッション・イメージ・ファイルがロードされていない場合，このコマンドは無効です。

### 6.1.4　Save As...

Save As...を選択すると，標準のファイル選択ダイアログ・ボックスが表示されます。現在のセッション・イメージ・ファイル（セッション状態）を新しいファイル名で保存します。

### 6.1.5　Close

Close コマンドは，現在のリンク・ファイルを閉じて，すべてのセグメントとシンボルを削除します。Module ウインドウにソース・ファイルが表示されている場合，Module ウインドウが View ウインドウに変わります。ただし，Module ウインドウは開いたままです。

### 6.1.6　Import...

Import...コマンドを選択すると，標準のファイル・オープン・ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，リンク・ファイル（LNK），16 進ファイル（HEX，HXI，HDX，HDY），データ・ファイル（DAT）などのファイルからデータをインポートするために使用します。ファイルを開くには，File name フィールドにファイル名とパスを入力するか，ディレクトリまたはドライブからファイルを選択します。Files of type リストには，利用できるインポート・ファイル拡張子が表示されます。必要なファイル拡張子に設定してください。

**（1）リンク・ファイルのインポート**

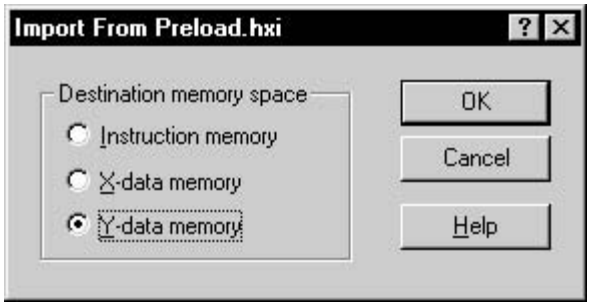
リンク・ファイルに含まれているプログラムとデータ情報がロードされます。ほかのリンク・ファイルが Open...コマンドでロードされている場合，シンボル情報は破棄されます。そうでなければ，シンボル情報もロードされ，ファイルは現在のアクティブ・ファイルに設定されます。

**（2）16 進ファイルのインポート**

16 進ファイルに含まれるデータは，インストラクション・メモリ，X データ・メモリ，Y データ・メモリのいずれかのメモリ空間にインポートできます。16 進ファイルをロードすることをユーザが確認すると，16 進ファイル・インポート用のダイアログ・ボックスが表示されます。このダイアログ・ボックスで，メモリ空間を選択します。

図 6－2 は 16 進ファイル・インポート用ダイアログ・ボックスです。

図 6－2　16 進ファイル・インポート用ダイアログ・ボックス



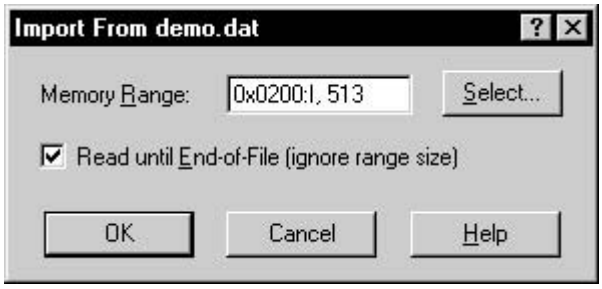

拡張子 HXI のファイルはインストラクション・メモリに，拡張子 HDX のファイルは X データ・メモリに，拡張子 HDY のファイルは Y データ・メモリにそれぞれディフォールトでロードされます。

(3) データ・ファイルのインポート

データ・ファイルはデータ値だけを含むファイルです。メモリ空間とメモリ範囲は，データ・インポート用のダイアログ・ボックスで指定しなければなりません。

図 6－3 はデータ・インポート用ダイアログ・ボックスです。

図 6－3　データ・インポート用ダイアログ・ボックス




メモリ空間とメモリ範囲は，直接入力することも，Select...ボタンをクリックして表示されるダイアログ・ボックスを利用して入力することもできます。Select ダイアログ・ボックスでは，リストからメモリ空間を選択し，データのロード先であるメモリ範囲の開始アドレスと終了アドレスを入力します。

メモリ範囲を直接入力する場合の入力構文については，11.4　**メモリ範囲の入力**を参照してください。

現在アクティブなメモリ・ウインドウのアドレス行が選択されている場合，対応するメモリの種類と範囲がインポートの対象として示されます。

ファイル全体をロードする場合，Read until End-of-File チェック・ボックスにチェック・マークを付けます。指定の開始アドレスからファイルがロードされますが，指定の終了アドレスは無視されます。

インポートされたデータの数値形式は，データが対象メモリ・ウインドウにあれば変更できます。

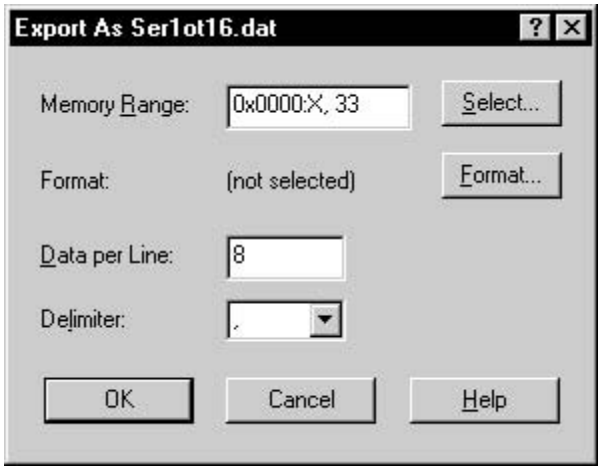
### 6.1.7　Export...

Export...コマンドを選択すると，標準のファイル選択ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，データをデータ・ファイル形式にエクスポートするために使用されます。Export As...ダイアログ・ボックスでエクスポートするファイルの種類と名前を選択してその内容を確認したあと，データ・エクスポート用のダイアログ・ボックスが表示されます。

図 6－4 はデータ・エクスポート用ダイアログ・ボックスです。

保守／廃止

**図 6－4　データ・エクスポート用ダイアログ・ボックス**




### (1) データ・エクスポート用ダイアログ・ボックスの構成要素

- Memory Range

  Import Data **ダイアログ・ボックスの場合と同様に，メモリ範囲は直接入力することも，**Select...**ボタンをクリックして表示されるダイアログ・ボックスを利用して入力することもできます。メモリ範囲を直接入力する場合の入力構文については，**11.4　**メモリ範囲の入力を参照してください。現在アクティブなメモリ・ウインドウのメモリ行が選択されている場合，それと一致するメモリの種類，範囲，および値がエクスポート対象として示されます。**

- Format

  Format...**ボタンをクリックすると，メモリ値の出力形式を指定するための** Number Format **ダイアログ・ボックスが表示されます。このダイアログ・ボックスについては，**6.4.1　Format...**を参照してください。**

- Data per Line

  **データ・ファイル** 1 **行当たりのデータ値の個数を指定できます。**1 **を入力すると，**1 **行ごとに** 1 **つの値が出力されます。**1 **行に複数の値を出力する場合，それぞれの値をユーザ定義の区切り文字で仕切ることができます。**

- Delimiter

  **区切り文字としてスペースとタブ（タブは¥t と表記）が示されますが，別の区切り文字を指定することもできます。**

- OK

  **ダイアログ・ボックスを閉じ，タイトル・バーで指定されている動作を実行します。**

- Cancel

  **ダイアログ・ボックスを閉じ，タイトル・バーで指定されている動作を取り消します。**

（2）Export Log Data

Export As...ダイアログ・ボックスは，現在のログ・セッションのデータをテキスト・ファイルにエクスポートするために使用します。ログ・データには，シミュレーション動作，シミュレーション・メッセージ，ブレークポイント通過，および HSM77016 のメモリ・ウインドウまたはレジスタ・ウインドウのデータが含まれます。一般にログ・データは，Log Viewer ツールで管理できるログ・ファイルに保存されます。

### 6.1.8　Exit

次のいずれかの動作で終了します。

データ・ウインドウの表示設定やブレークポイントなどは，Tools メニューの Options...コマンドで保存できます。

- File メニューから Exit コマンドを選択します。
- HSM77016 のメイン・ウインドウの閉じるボタンをダブルクリックします。
- メイン・ウインドウのシステム・メニューから Close コマンドを選択します。

### 6.1.9　ファイルのリスト

File メニューの下部にあるファイル・リストを使用すれば，最近使用したファイルに素早くアクセスできます。つまり，ダイアログ・ボックスでファイルの場所を指定しなくても，リストからファイル名を選択するだけで，ファイルをすぐに開くことができます。このリストには，最大 4 個のファイル名が表示され，最近開いたファイルから順に 1，2，3，4 とファイル名の先頭に番号が付けられます。このリストは常に更新され，最も最近開いたファイルが一番上に表示されます。

## 6.2　Edit メニュー

データ・ウインドウのセルの編集中だけ，Edit メニューの Undo，Cut，Copy，Paste コマンドを使用できます。

### 6.2.1　Undo

Undo コマンドは，ユーザによる最後のデータ変更処理を取り消します。取り消しできない処理の場合，このコマンドは薄く表示されます。取り消し動作の簡単な説明が，Undo コマンドの右に表示されます。

### 6.2.2　Cut

Cut コマンドは，選択されたデータをクリップ・ボードに入れます。元の位置のデータは削除されます。

### 6.2.3　Copy

Copy コマンドは，選択されたデータをクリップ・ボードにコピーします。

### 6.2.4　Paste

Paste コマンドは，クリップ・ボードのデータをカーソル位置に挿入するか，現在選択されているデータと置き換えます

### 6.2.5　Delete

Delete コマンドは，次の対象を削除します。

- メモリ・ウインドウとレジスタ・ウインドウ内の選択されたデータ
- Breakpoint ウインドウ内の選択されたブレークポイント
- Watch ウインドウ内の選択されたウォッチ

**注意**　Delete コマンドの取り消し（undo 処理）はできません。また，Delete コマンドでデータをクリップ・ボードにコピーすることもできません。

### 6.2.6　Dump to log

Dump to log コマンドは，現在選択されているログ・データを Log ウインドウに表示します。HSM77016 の任意のメモリ・ウインドウ項目またはレジスタ・ウインドウ項目をログ・データにすることができます。

Log ウインドウの内容に対して次の操作ができます。

- Log Viewer ツールで管理できるログ・ファイルに保存します。
- File メニューの Export...コマンドでテキスト・ファイルに保存します。
- Log メニューの Clear Session コマンドでクリアします。

## 6.3　View メニュー

View メニューには，メモリ・ウインドウ，レジスタ・ウインドウ，またはテキスト・ウインドウ内の位置を移動するコマンド，変数とシンボル・テーブルを調べるコマンド，ツール・バーとステータス・バーの表示のオンとオフを切り替えるコマンドが含まれています。

### 6.3.1　Goto...

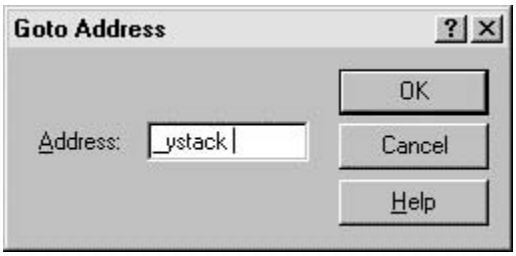
メモリ・ウインドウ，レジスタ・ウインドウ，またはテキスト・ウインドウがアクティブな場合だけ，このコマンドを使用できます。

**（1）アドレスへの移動**

Instruction Memory ウインドウ，X-Data Memory ウインドウ，または Y-Data Memory ウインドウがアクティブな場合は，Goto...コマンドを選択すると Goto Address ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，セルのアンカを指定のアドレスに移動するために使用します。

図 6－5 は Goto Address ダイアログ・ボックスです。

**図 6－5　Goto Address ダイアログ・ボックス**



保守／廃止

(2) Goto Address **ダイアログ・ボックスの構成要素**

- Address

  **アドレス値は，**10 **進，**2 **進，**8 **進，**16 **進のいずれかの数値形式で入力できます。数値形式の構文が誤っている場合や，指定されたアドレスがメモリの範囲外にある場合は，エラー・メッセージが表示されます。**

- OK

  **ダイアログ・ボックスを閉じ，指定されたアドレスにアンカを移動します。数値形式の構文が誤っている場合や，指定されたアドレスがメモリの範囲外にある場合は，エラー・メッセージが表示されます。**

- Cancel

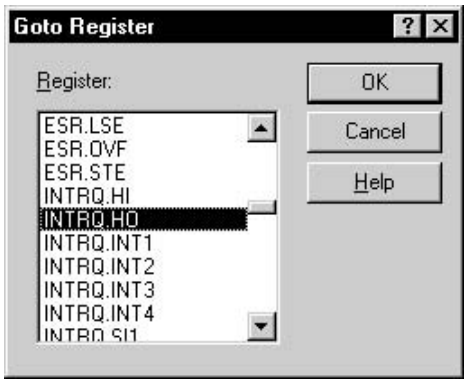
  **ダイアログ・ボックスを閉じ，タイトル・バーに示された操作を取り消します。**

### (3) レジスタへの移動

CPU Register **ウインドウまたは** Peripheral Register **ウインドウがアクティブな場合は，**Goto...**コマンドを選択すると** Goto Register **ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，アクティブ・ウインドウの指定のレジスタにアンカを移動するために使用します。**

**図** 6－6 **は** Goto Register **ダイアログ・ボックスです。**

**図** 6－6　Goto Register **ダイアログ・ボックス**




(4) Goto Register **ダイアログ・ボックスの構成要素**

- Register

  Register **リストには，アクティブなレジスタ・ウインドウに表示されるすべてのレジスタが示されます。このリストから，移動先のレジスタを選択してください。**

- OK

  **ダイアログ・ボックスを閉じ，アクティブなレジスタ・ウインドウの指定されたレジスタにアンカを移動します。**

- Cancel

  **レジスタに移動せずにダイアログ・ボックスを閉じます。**

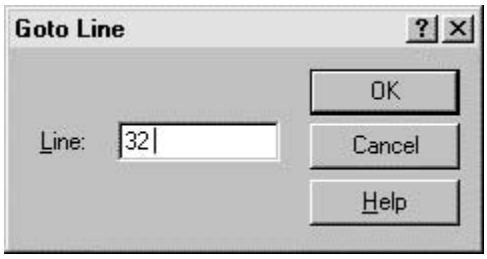
保守／廃止

**（5）行への移動**

**テキスト・ウインドウ（**View**，**Module**，**Timing File**）がアクティブな場合は，**Goto...**コマンドを選択すると** Goto Line **ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，カーソルをテキスト・ウインドウの指定の行に移動するために使用します。**

**図** 6－7 **は** Goto Line **ダイアログ・ボックスです。**

**図** 6－7　Goto Line **ダイアログ・ボックス**




**（6）**Goto Line **ダイアログ・ボックスの構成要素**

- Line
  **行番号の入力フィールドです。入力した行番号がウインドウの実際の行番号を越えている場合は，エラー・メッセージが表示されます。**

- OK
  **ダイアログ・ボックスを閉じ，アクティブ・ウインドウの指定された行の先頭にカーソルを移動します。**

- Cancel
  **行に移動せずにダイアログ・ボックスを閉じます。**

### 6.3.2　Follow

Follow **コマンドには，アンカ位置の値によって指定された，**Instruction Memory **ウインドウ，**X-Data Memory **ウインドウ，または** Y-Data Memory **ウインドウのアドレスへジャンプするための** 3 **種類のサブコマンドがあります。**CPU Register **ウインドウ，**X-Data Memory **ウインドウ，または** Y-Data Memory **ウインドウにアンカがある場合だけ，このコマンドを使用できます。**

### 6.3.3　Variable...

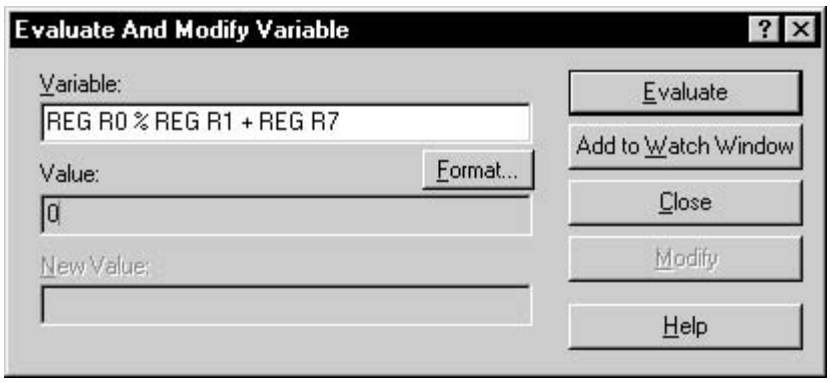
Variable...**コマンドを選択すると，**Evaluate and Modify Variable **ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，式の値を調べたり変更するために使用します。式として有効なのは次のものです。**

- C **言語ソースの変数または式**
- HSM77016 **変数**
- **タイミング・ファイル変数**
- **数値**
- **レーベル**
- HSM77016 **で許容されている演算子で組み合わされた数値と変数**

保守／廃止

図 6－8 は Evaluate And Modify Variable ダイアログ・ボックスです。

図 6－8　Evaluate And Modify Variable ダイアログ・ボックス




（1）Evaluate And Modify Variable ダイアログ・ボックスの構成要素

- Variable
  変更または確認する式を入力します。

- Value
  評価された（現在の）式の値が表示されます。

- New Value
  新しい式の値を入力します。このフィールドを使用できるのは，式の値を変更できる場合だけです。

- Format
  Value フィールドの表示形式を変更または確認するための Number Format ダイアログ・ボックスを表示します。このボタンを使用できるのは，Value フィールドに値が表示されている場合だけです。

- Evaluate
  式を評価し，Value フィールドに評価結果を表示します。式が無効な場合は，Value フィールドにエラー・メッセージが表示されます。

- Add to Watch Window
  Watch ウインドウに変数（式）を追加します。

- Close
  ダイアログ・ボックスを閉じます。

- Modify
  新しい式の値を設定します。このボタンを使用できるのは，式の値を変更できる場合だけです。

**注意**　New Value と Modify を使用できるのは，式を変更できる場合だけです。

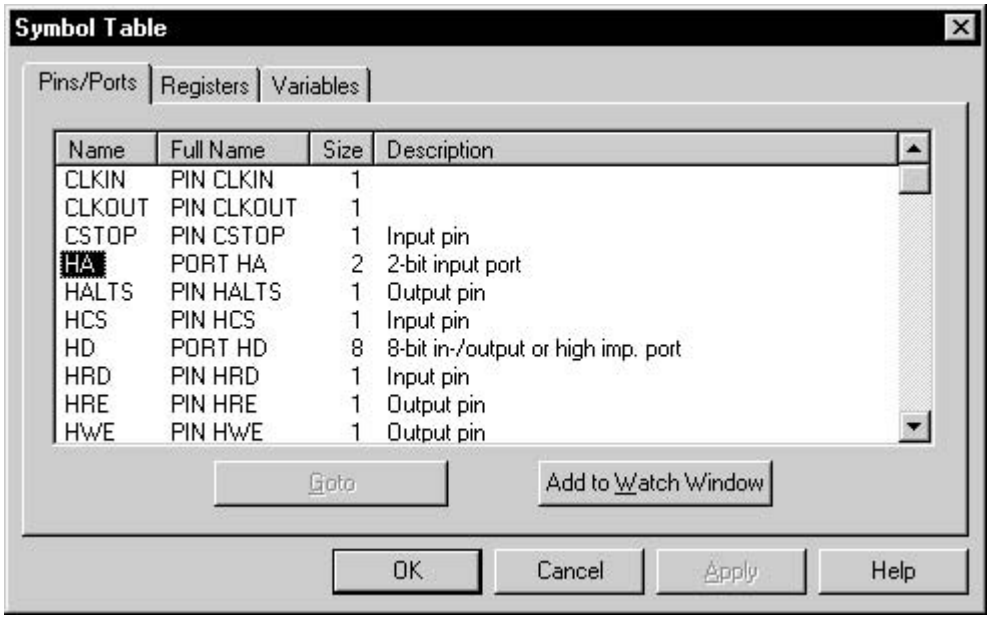
保守／廃止

### 6.3.4　Symbol Table...

Symbol Table...**コマンドを選択すると，タブのあるダイアログ・ボックスが表示されます。このダイアログ・ボックスは，現在ロードされているシンボル・テーブルを調べるために使用します。シンボル・テーブルのカテゴリはタブによって区分けされています。それぞれのタブには，対応するカテゴリのシンボルを含むリストが表示されます。タブの列ヘッダをクリックすれば，その列の表示リスト項目を昇順または降順に並べ替えることができます。論理セグメントおよびシンボルは通常，**WB77016 **のアセンブラで作成され，リンク・ファイルによってインポートされます。**

**図** 6－9 **は** Symbol Table **ダイアログ・ボックスです。**

**図** 6－9　Symbol Table **ダイアログ・ボックス**




**（1）**Pins/Ports **タブ，**Registers **タブ，**Variables **タブ**

Pins/Ports **タブと** Registers **タブには，対応する** DSP **アーキテクチャ要素が表示されます。**Variables **タブには，**HSM77016 **の時間とカウント測定変数（**CYCLE，STEP，TIME**）と，現在ロードされているソース・ファイルで宣言されている変数（ローカルおよびグローバルのタイミング・ファイル変数など）が表示されます。**

- Name

**参照構文に関係なく，変数名が表示されます。**Variable **タブには，時間とカウント測定変数が常に表示されます。**

- CYCLE：**シミュレートされたクロック・サイクル数を示します。**
- STEP：**実行済みの命令の数を示します。**
- TIME： **実行時間をナノ秒（**ns**）で示します。**
- TIME_RESOLUTION：**シミュレーション時間を** 1 **秒ごとに分解します。**

- Full Name

HSM77016 **内の任意の式で変数が参照されるときに書き込む必要のある変数名が表示されます。**

- Size

  対応する変数のサイズがビットで表示されます。

- Description

  補足情報が表示されます。

- Goto

  CPU Register ウインドウまたは Peripheral Register ウインドウの選択されたレジスタまたはビット・フィールドに移動します。レジスタ・セルまたはビット・フィールドにアンカが置かれます。選択されたレジスタまたはビット・フィールドが存在しない場合，対応するレジスタ・ウインドウが開きます。

- Add to Watch Window

  このボタンは，リストで現在選択されている項目を Watch ウインドウに追加します。

（2）Segments タブ

このタブには，メモリ保持コードまたはデータの連続部分であるセグメント（通常，WB77016 で作成されます）が表示されます。

- Name

  現在ロードされているリンク・ファイルで定義されているセグメントが表示されます。

- Area

  セグメントが配置されたメモリ領域が表示されます。

- Begin

  セグメント開始アドレスが表示されます。

- End

  セグメント終了アドレスが表示されます。

（3）Symbols タブ

このタブには，定義されているすべてのシンボルのプロパティが表示されます。

- Name

  現在ロードされているリンク・ファイルで定義されているシンボルが表示されます。

- Segment

  シンボルを含むセグメントの名前が表示されます。

- Value

  シンボル値が表示されます。

保守／廃止

### （4）C-Functions タブの構成要素

C-Functions タブには，現在ロードされている C ソース・ファイルで宣言されている関数が表示されます。このタブを使用できるのは，C ディバグ情報を含むリンク・ファイルがロードされている場合だけです。

- Name
  C 関数の名前が表示されます。

- Definition
  パラメータの種類を含むパラメータと関数タイプが表示されます。

- Location
  関数がロードされている C ソース・ファイルの名前とパスが表示されます。

- Goto
  Goto ボタンは，現在選択されている関数に適用されます。関数が定義されているソース・ファイルを Module ウインドウにロードし，関数定義が記述されている行の先頭にカーソルを移動します。

### （5）C-Variables タブの構成要素

C-Variables タブには，現在ロードされている C ソース・ファイルで宣言された変数または識別子が表示されます。このタブを使用できるのは，C ディバグ情報を含むリンク・ファイルがロードされている場合だけです。

- Name
  変数または識別子の名前が表示されます。

- Definition
  タイプ情報とメモリ位置を含む変数または識別子の名前が表示されます。

- Description
  対応する変数が位置しているメモリ・アドレスと変数タイプに関する情報が表示されます。

- Scope
  対応する変数が有効な範囲が表示されます。

- Add to Watch Window
  このボタンで，リストで現在選択されている項目を Watch ウインドウに追加します。このボタンは識別子や構造体のメンバには適用できないので，注意してください。

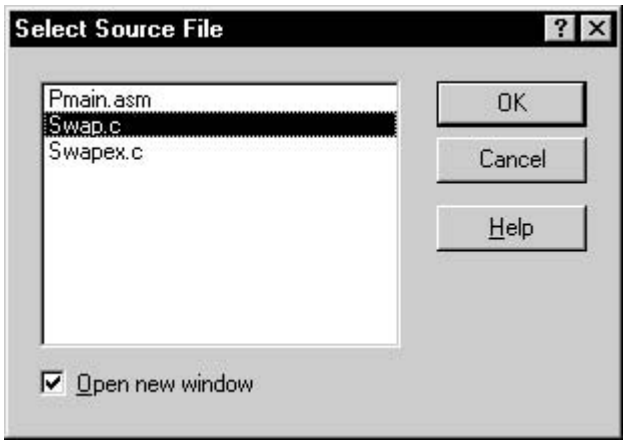
保守／廃止

### 6.3.5　Module...

Module...コマンドを使用できるのは，ディバグ情報を含んでいるリンク・ファイルがロードされた場合だけです。ディバグ情報には，ロードされたリンク・ファイルの作成に使用されたすべてのソース・ファイルのリストが含まれます。このコマンドを選択すると，Select Source File ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，ソース・レベルのシミュレーション用のソース・ファイルを選択するために使用します。選択したファイルは，Module ウインドウに表示されます。

図 6－10 は Select Source File ダイアログ・ボックスです。

図 6－10　Select Source File ダイアログ・ボックス




（1）Select Source File ダイアログ・ボックスの構成要素

- File List
  Module ウインドウで開くファイルを選択します。

- Open new window
  このボックスにチェック・マークを付けると，選択したファイルが新しい Module ウインドウに開きます。

- OK
  ダイアログ・ボックスを閉じ，選択したソース・ファイルを開きます。

- Cancel
  ソース・ファイルを開かないでダイアログ・ボックスを閉じます。

### 6.3.6　Toolbar

Toolbar コマンドは，ツール・バー表示のオンとオフを切り替えます。表示がオンの場合，このコマンドの左にチェック・マーク（√）が付きます。ツール・バーのボタンとフィールドの詳細については，3.1.3 **ツール・バー**を参照してください。

### 6.3.7　Status Bar

Status Bar コマンドは，ステータス・バー表示のオンとオフを切り替えます。表示がオンの場合，このコマンドの左にチェック・マーク（√）が付きます。ステータス・バーの詳細については，3.1.4 **ステータス・バー**を参照してください。

## 6.4　Data メニュー

### 6.4.1　Format...

HSM77016 は，数値を次の各形式で表示することができます。

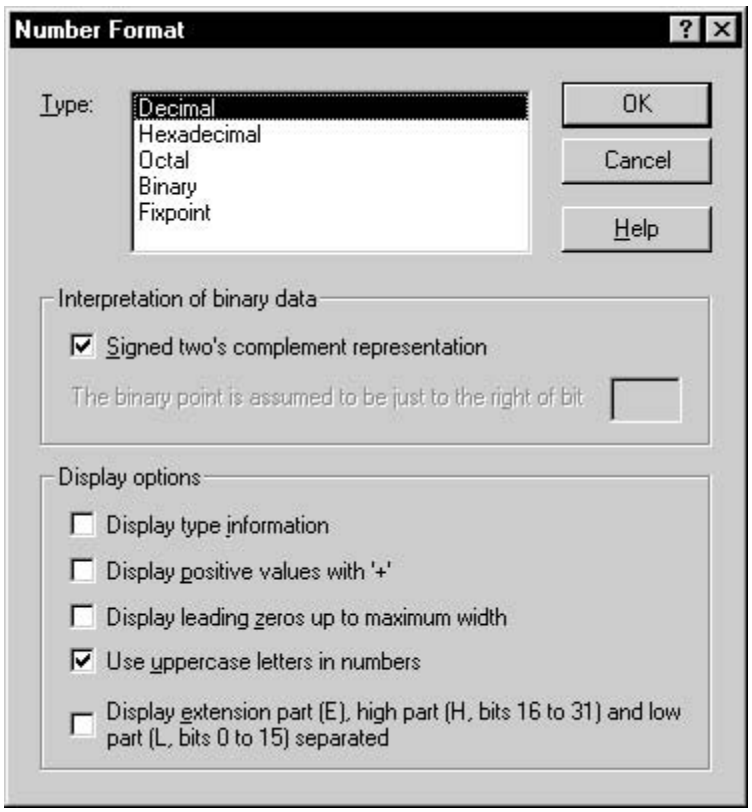
- 2 進
- 10 進
- 16 進
- 8 進
- 固定小数点
- ニモニック
- C 言語の式
- ステータス
- 値

**注意　上記の形式がすべてのデータ・セルで使用できるとはかぎりません。**Number Format **ダイアログ・ボックスの** Type **リストには，アンカが位置しているデータ列で使用できる表示形式だけが示されます。**Type **選択ボックスの中で希望の形式をクリックして表示形式を選択します。**

**図** 6－11 **は** Number Format **ダイアログ・ボックスです。**

保守／廃止

**図** 6－11　Number Format **ダイアログ・ボックス**




**（**1**）**Number Format **ダイアログ・ボックスの構成要素**

- Type
  **使用できる数値表示形式のリストです。このリストから表示形式を選択します。**

- Signed two's complement representation
  **数値を符号付きの** 2 **の補数で表示します。最上位桁が符号ビットです。**

- The binary point is assumed to be just to the right of bit
  **ピリオドの位置を指定するための入力ボックスです。たとえば，**31 **を入力すると，ビット** 31 **と** 32 **の間がピリオドになります。入力できる値は，**0 **から** 39 **の整数です。**

- Display type information
  **値とともに基数も表示します。たとえば，**16 **進数では** 0x **が表示されます。**

- Display positive values with '+'
  **正数に“** + **”を表示します。この選択ボックスにチェック・マークを付けても，符号付き数値として表示されるわけではありません。符号付き数値として表示するには，**Signed two's complement representation **にチェック・マークを付ける必要があります。**

- Display leading zero's up to maximum width
  **先行ビットが** 0 **の場合も** 0 **を表示します。**

- Use uppercase letters in numbers
  **文字を大文字で表示します。このボックスにチェック・マークを付けない場合，文字は小文字で表示されます。**
- Display extension part, high part and low part separated
  **40 ビット値を** 8，16，16 **ビットに分けて表示します。この選択が有効なのは，**40 **ビット値の場合だけです。**
  **数値表示形式の詳細については，**11.5　**数値形式の構文を参照してください。**

**注意　数値の表示形式はツール・バーのボタンでも設定できます。詳細については，**3.1.3　**ツール・バーを参照してください。**

### 6.4.2　Add Column

Add Column **コマンドを選択すると，アンカが位置しているデータ列の左端に分割用カーソル（↤┼↦）が表示されます。新しい列を作成してセル範囲にドラッグすることができます。新しい列は，左端にあった列の表示形式属性をすべて引き継ぎます。**

### 6.4.3　Size Column

Size Column **コマンドを選択すると，アンカが位置しているデータ列の右端にサイズ調整用カーソル（↔）が表示されます。このカーソルをドラッグすれば，セル範囲のサイズを変更できます。**

### 6.4.4　Zero, Increment, Decrement

**これら** 3 **つのコマンドは，選択されているすべてのセルに適用されます。**Zero **コマンドは，セルの内容をゼロにリセットします。**Increment **コマンドはセルの内容を** 1 **増分し，**Decrement **コマンドは** 1 **減分します。以上のコマンドの機能は，コンテキスト・メニューを右クリックするか編集バーにある同じ名前のボタンでも実行できます。詳細については，**3.1.2　**エディット・バーを参照してください。**

## 6.5　Run メニュー

**この節では，仮想** DSP **内のプログラム実行とタイミング・ファイル実行のシミュレーションの際に利用できるオプションについて説明します。**

**タイミング・ファイル（ウインドウ）がアクティブでない場合のシミュレーション実行用コマンドは，命令コードとソース・コードに関連しています。これらのコマンドは，大きく分けて次の** 2 **種類になります。**

- **ステップ実行**
  **レジスタやメモリ領域の表示は，各シミュレーション・ステップ後に更新されます。ユーザは，線形プログラム，サブルーチン，割り込みルーチンが実行される場合でも，**DSP **実行を正確に追跡できます。**

- **ブロック実行**
  **シミュレーションは，次のようなイベントが発生するまで表示を更新せずにフル・スピードで続きます。イベントとしては，ブレークポイントへの到達，サブルーチンの完了（ネストしているすべてのサブルーチンを含む），**RETURN **命令の実行，プログラム・カウンタが入力カーソルに達した場合，ユーザのキー入力などがあります。**

保守／廃止

タイミング・ファイル実行コマンドは，現在アクティブなタイミング・ファイル（ウインドウ）に関連しています。タイミング・ファイルは 1 行ずつ実行されます。実行中に WAIT xxx コマンドが選択されると，タイミング・ファイル処理は，WAIT 条件が真になるまで現在のステップにとどまります。タイミング・ファイル実行コマンドは，Run メニューの Run，Break，Trace，Animate-Trace です。

### 6.5.1　Run

Run コマンドで，シミュレーションを開始します。非明示的ブレークポイントは無視されますが，明示的ブレークポイントはアクティブです。一度シミュレーションが始まると，このコマンドはグレー表示されます。シミュレーションは，Break コマンドが選択されるか，ブレークポイントに達するまで続きます。シミュレーション中，表示は更新されません。ロードされたタイミング・ファイルがシミュレーションと並行して実行されます。

**（1）タイミング・ファイルの実行**

タイミング・ファイル・ウインドウがアクティブである場合，Run コマンドでタイミング・ファイルの実行が始まります。一度実行が始まると，このコマンドはグレー表示されます。タイミング・ファイル・ブレークポイントに達するか，WAIT コマンドが選択されるまで，タイミング・ファイルは 1 行ずつ実行されます。Break コマンドを選択すれば，タイミング・ファイルの実行を停止できます。シミュレーションは，タイミング・ファイルの実行に関係なく実行できます。

### 6.5.2　Break

Break コマンドで，シミュレーションを中断します。Break コマンドを選択すると，シミュレーションが停止し，ブレーク発生箇所の状態を反映するように表示が更新されます。中断したシミュレーションを再開するには，Run コマンドを選択します。

**（1）タイミング・ファイルの実行**

タイミング・ファイル・ウィンドウがアクティブである場合，Break コマンドはタイミング・ファイルの実行を中断させます。中断したタイミング・ファイルの実行を再開するには，Run コマンドを選択します。

### 6.5.3　Instruction Trace

Instruction Trace コマンドは，1 命令分を実行します。割り込みルーチン内部の実行の途中経過も表示します。

### 6.5.4　Trace

Trace コマンドは，1 ステップをシミュレーションします。サブルーチン内部の実行の途中経過も表示します。しかし，割り込みルーチン内部の実行の途中経過は表示しません。

**（1）タイミング・ファイルの実行**

タイミング・ファイル・ウインドウがアクティブである場合，Trace コマンドはタイミング・ファイルを 1 行ずつ実行します。シミュレーションは，タイミング・ファイルの実行に関係なく実行できます。

### 6.5.5　Step

Step コマンドは，1 ステップをシミュレートします。サブルーチン内部の実行の途中経過は表示しません。

現在の命令がサブルーチン・コール命令（CALL）の場合，サブルーチン全体が実行されます。

現在の命令が LOOP または REPEAT 対象命令の最後の命令の場合，LOOP または REPEAT 命令を完了し，ループまたはリピートのあとの次の命令で実行が停止します。

保守／廃止

### 6.5.6　Animate | Instruction Trace

**アニメート・インストラクション・トレース・モードでは，命令のトレースを実行します。画面を更新し，一定の遅延の間ウエイトして，自動的に次の命令から続行します。**

### 6.5.7　Animate | Trace

**アニメート・トレース・モードでは，トレース・モードで実行します。画面を更新し，一定の遅延の間ウエイトして，自動的に次の命令から続行します。**

### 6.5.8　Animate | Step

**アニメート・ステップ・モードでは，ステップ・モードで実行します。画面を更新し，一定の遅延の間ウエイトして，自動的に次の命令から続行します。**

### 6.5.9　Until Return

Until Return **コマンドは，現在のサブルーチンから** RETURN **が実行されるまでシミュレーションを実行します。**

### 6.5.10　To Cursor

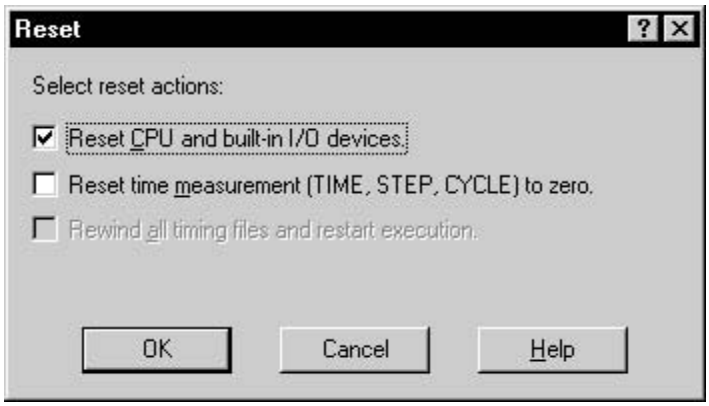
To Cursor **コマンドは，インストラクション・メモリ内の現在のカーソル・アドレスに達するまでシミュレーションを実行します。このコマンドを使用できるのは，**Instruction Memory **ウインドウまたは** Module **ウインドウがアクティブな場合だけです。それ以外の場合は使用できず，グレー表示されます。**

### 6.5.11　Reset...

Reset...**コマンドで，**Reset **ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，リセット動作を選択します。**

**図** 6－12 **は** Reset **ダイアログ・ボックスです。**

**図** 6－12　Reset **ダイアログ・ボックス**




**（**1**）**Reset **ダイアログ・ボックスの構成要素**

- Reset CPU and built-in I/O devices.
  **このコマンドにチェック・マークを付けると，リセット・シグナルがシミュレーションしているプロセッサに送信されます。このコマンドの効果は，短いリセット・パルスを物理チップに送信する場合に似ています。**

- Reset time measurement (TIME, STEP, CYCLE) to zero.
  このコマンドにチェック・マークを付けると，HSM77016 の時間とカウント測定変数 TIME，STEP，CYCLE がゼロにリセットされます。

- Rewind all timing files and restart execution.
  このコマンドにチェック・マークを付けると，現在ロードされているタイミング・ファイルがすべて始めに戻されます。このコマンドは，タイミング・ファイルの状態に関係なく適用されます。このコマンドは，タイミング・ファイルを削除したあとで再度開く処理と同じです。このコマンドが有効なのは，複数のタイミング・ファイルがロードされている場合だけです。

- Rewind
  このコマンドにチェック・マークを付けると，現在アクティブなタイミング・ファイル・ウインドウ（ファイル名がコマンドの右に表示されます）が始めに戻されます。このコマンドは，タイミング・ファイルの状態に関係なく適用されます。このコマンドは，タイミング・ファイルを削除したあとで再度開く処理と同じです。“ Rewind all timing files and restart execution. ” にチェック・マークが付いている場合，このコマンドは無効です。Rewind コマンドは，タイミング・ファイル・ウインドウがアクティブな場合だけ表示されます。

- OK
  ダイアログ・ボックスを閉じ，選択されたオプションでリセット動作を実行します。

- Cancel
  リセット動作を実行しないでダイアログ・ボックスを閉じます。

### 6.5.12　Back Trace

Back Trace コマンドは，最後にシミュレーションしたプログラムを，後戻りしながら表示します。これは，Undo コマンドと同じ機能です。ただし，表示は前方シミュレーションと同様です。たとえば，後戻りによって復元されたレジスタ値は，変更された値として表示されます。

シミュレーションが前方に実行されていなければ，このコマンドを使用できません。

シミュレーションによっては，このコマンド用の内部バッファがオーバフローして，コマンドを使用できなくなることがあります。

保守／廃止

# 6.6　Memory メニュー

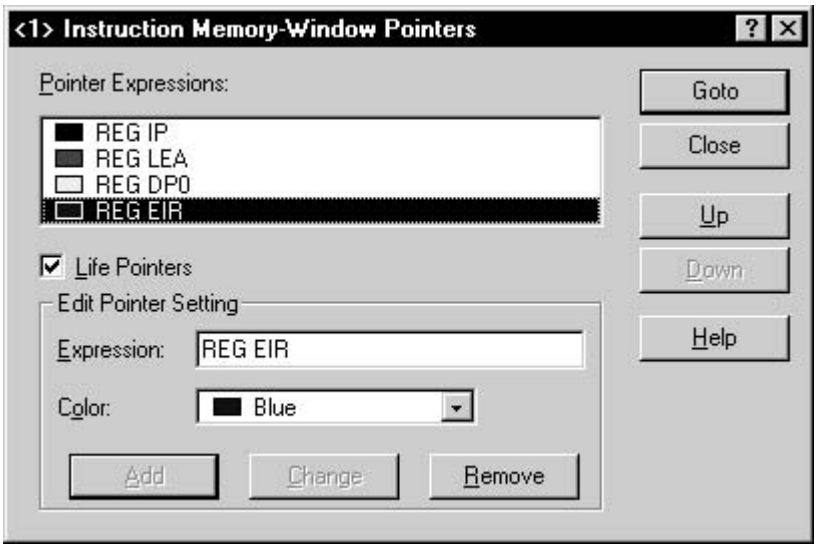
Memory メニューの各コマンドは，現在アクティブな Instruction Memory ウインドウ，X-Data Memory ウインドウ，または Y-Data Memory ウインドウに適用されます。したがって，Memory メニューの項目が有効なのは，メモリ・ウインドウがアクティブな場合にかぎられます。

## 6.6.1　Pointer...

Pointer...を選択すると，Window Pointers ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，プログラムの実行中に式をウォッチするためのポインタをユーザが独自に定義できます。ポインタは，メモリ・ウインドウのアドレス列とデータ列の間に（▶）で表示されます。ポインタ式を区別するために，各ポインタにそれぞれ異なる色を割り当てることができます。Window Pointers ダイアログ・ボックスのタイトル・バーには，ポインタが適用されるメモリ・ウインドウの名前が表示されます。

図 6－13 は Window Pointers ダイアログ・ボックスです。

図 6－13　Window Pointers ダイアログ・ボックス




（1）Window Pointers ダイアログ・ボックスの構成要素

- Pointer Expressions

  このリスト・ボックスには，すべてのユーザ定義ポインタとそれに割り当てられている色が表示されます。ポインタを選択するには，カーソルをその行に移動してクリックします。

- Life Pointers

  このボックスにチェック・マークを付けると，シミュレーションが続いている間，指定されたポインタが表示されるよう，表示がスクロールします。

- Expression

  ポインタ式を入力または編集するための入力フィールドです。

- Color
  このドロップダウン・リストには，利用できるすべてのポインタ色と，追加の色を作成するための Custom Color...項目が含まれています。

- Add
  Expression 入力フィールド内の新しいポインタ式を，現在選択されている色でポインタ・リストに追加します。

- Change
  選択したポインタの色や式を変更します。

- Remove
  選択したポインタをリストから削除します。

- Goto
  ダイアログ・ボックスを閉じて，現在指定されているポインタ式の位置を含むメモリ・ウインドウ範囲を表示します。

- Close
  ダイアログ・ボックスを閉じます。

- Up/Down
  選択したポインタをポインタ・リスト内で上下に移動します。リスト内の位置が上であるほど，Life Pointers が有効な場合にそのポインタが表示されやすくなります。

### 6.6.2　Show Header

Show Header コマンドは，ヘッダ行表示のオンとオフを切り替えます。ヘッダ行を表示しないと，画面のスペースを節約できます。このコマンドの左にチェック・マーク（√）が付いている場合，ヘッダ行が表示されます。

### 6.6.3　Show Symbols

Show Symbols コマンドは，シンボル表示のオンとオフを切り替えます。このコマンドの左にチェック・マーク（√）が付いている場合，シンボルが表示されます。

表示されるソース・データは，リンク・ファイルのディバグ情報から得られます。ソース・データ範囲は，関連するソース・ファイルの名前が表示されたヘッダ行で強調されます。ソース行には，行番号とその行のソース・コードが表示されます。ソース行の表示色は，Tools メニューの Options...コマンドで変更できます。

関連するソース・コードを持たない命令コードは，ソース・データ範囲の最後に収集され，“ No File ”ヘッダ行で強調されます。ディバグ情報をまったく含まないファイルにシミュレーションが入ると，シミュレーションの取り消しまたは続行を選択するよう，ユーザにメッセージが表示されます。

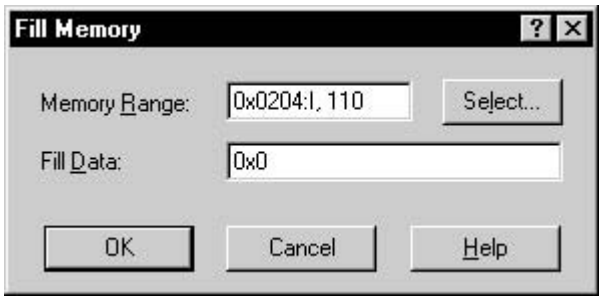
保守／廃止

### 6.6.4　Fill...

Fill...コマンドを選択すると，Fill Memory ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，ある値で特定のメモリ範囲（インストラクション・メモリ，X データ・メモリ，Y データ・メモリ）を初期化または書き込むために使用します。

図 6－14 は Fill Memory ダイアログ・ボックスです。

図 6－14　Fill Memory ダイアログ・ボックス




（1）Fill Memory ダイアログ・ボックスの構成要素

- Memory Range

  Select Range ダイアログ・ボックスで選択したメモリ範囲とメモリ・ボックスの開始アドレスが表示されます。右端の値は，選択されたメモリ・ワード数を示しています。Select...ボタンをクリックしないで，メモリ範囲を直接入力することもできます。入力構文の詳細については，11.4　**メモリ範囲の入力**を参照してください。

- Fill Data

  メモリに書き込む値を入力するためのフィールドです。数値，アドレス，レーベル，レジスタ名を入力できます。

- OK

  指定のデータでメモリの書き込みを開始します。ステータス・バーに，書き込み処理の進行状況が示され，終了すると書き込まれたワード数が表示されます。

- Cancel

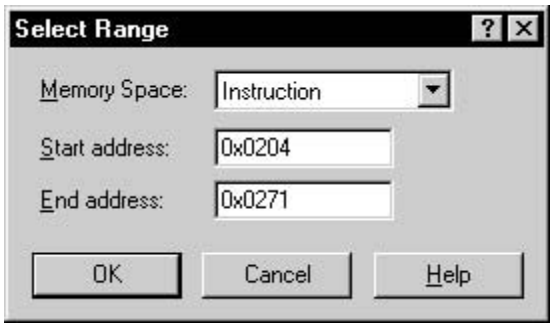
  メモリの書き込みを行わずにダイアログ・ボックスを閉じます。

保守／廃止

(2) Select...**ボタン**

Select...**ボタンをクリックすると，図** 6－15 **に示した** Select Range **ダイアログ・ボックスが表示されます。**

**図** 6－15　Select Range **ダイアログ・ボックス**




(3) Select Range **ダイアログ・ボックスの構成要素**

- Memory Space

  **検索するメモリ空間を選択します。インストラクション・メモリ，**X **データ・メモリ，**Y **データ・メモリのいずれかを選択できます。**

- Start address/End address

  **書き込み対象となるメモリ・ブロックの開始アドレスと終了アドレスです。終了アドレス（メモリ範囲に含まれます）は，開始アドレスよりも大きく，また両アドレスは選択されたメモリ・タイプに有効なアドレス範囲内にある必要があります。アドレスの数値形式は，**10 **進，**2 **進，**8 **進，**16 **進形式のいずれかです。**

- OK

  **入力内容を確認してダイアログ・ボックスを閉じます。**

- Cancel

  **入力内容を拒否してダイアログ・ボックスを閉じます。**

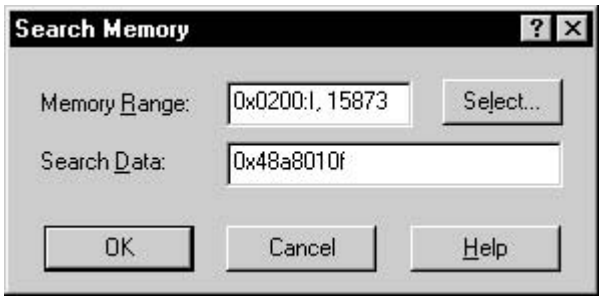
保守／廃止

### 6.6.5　Search...

Search...コマンドを選択すると，Search Memory ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，指定のメモリ範囲からある値を検索するために使用します。値が検索されると，検索を続けるかどうかの問い合わせが表示されます。

図 6－16 は Search Memory ダイアログ・ボックスです。

図 6－16　Search Memory ダイアログ・ボックス




（1）Search Memory ダイアログ・ボックスの構成要素

- Memory Range

  Select Range ダイアログ・ボックスで選択したメモリ範囲とメモリ・ブロックの開始アドレスが表示されます。右端の値は，選択されたメモリ・ワード数を示しています。Select...ボタンをクリックしないで，メモリ範囲を直接入力することもできます。入力構文の詳細については，11.4　**メモリ範囲の入力**を参照してください。

- Select...

  このボタンをクリックすると，Select Range ダイアログ・ボックスが表示されます。このダイアログ・ボックスのオプションとコントロールについては，6.6.1　Pointer...の説明を参照してください。

- Search Data

  検索データとして式を入力します。式には，数値，アドレス，レーベル，レジスタ名を含むことができます。

- OK

  検索を開始します。ステータス・バーに検索状況が示され，これ以上一致するものがないことや，検索の終了が通知されます。検索データが見つからない場合や，入力された式が認識されない場合はメッセージが表示されます。検索データが見つかると，検索式の値と一致する指定されたメモリ空間内の最初の場所にアンカが移動します。次または直前の一致を検索するための Search ダイアログ・ボックスが表示されます。

- Cancel

  検索を中止してダイアログ・ボックスを閉じます。

保守／廃止

### 6.6.6　Toggle Breakpoint

このコマンドは，現在のアンカ位置での実行ブレークポイントの有効／無効を切り替えます。

アンカ位置に実行ブレークポイントがない場合，このアドレスに実行ブレークポイントが設定され有効になります。

アンカ位置に実行ブレークポイントがある場合，ブレークポイントの状態が有効なら無効に，無効なら有効になります。

### 6.6.7　Toggle Marker

このコマンドは，現在アンカがある位置のプロファイリング・マーカの設定を切り替えます。アンカまたはカーソル位置にプロファイリング・マーカがない場合，このアドレスにプロファイリング・マーカが設定され有効になります。アンカまたはカーソル位置にプロファイリング・マーカがすでにある場合，その状態が有効なら無効に，無効なら有効になります。

### 6.6.8　Toggle Entry Marker

このコマンドは，現在アンカがある位置の Entry マーカの設定を切り替えます。アンカまたはカーソル位置に Entry マーカがない場合，このアドレスに Entry マーカが設定され有効になります。アンカまたはカーソル位置に Entry マーカがすでにある場合，その状態が有効なら無効に，無効なら有効になります。

### 6.6.9　Toggle Exit Marker

このコマンドは，現在アンカがある位置の Exit マーカの設定を切り替えます。アンカまたはカーソル位置に Exit マーカがない場合，このアドレスに Exit マーカが設定され有効になります。アンカまたはカーソル位置に Exit マーカがすでにある場合，その状態が有効なら無効に，無効なら有効になります。

### 6.6.10　Toggle Start Marker

このコマンドは，現在アンカがある位置の Start マーカの設定を切り替えます。アンカまたはカーソル位置に Start マーカがない場合，このアドレスに Start マーカが設定され有効になります。アンカまたはカーソル位置に Start マーカがすでにある場合，その状態が有効なら無効に，無効なら有効になります。

### 6.6.11　Toggle Stop Marker

このコマンドは，現在アンカがある位置の Stop マーカの設定を切り替えます。アンカまたはカーソル位置に Stop マーカがない場合，このアドレスに Stop マーカが設定され有効になります。アンカまたはカーソル位置に Stop マーカがすでにある場合，その状態が有効なら無効に，無効なら有効になります。

### 6.6.12　Follow Jmp / Call

Follow Jmp/Call コマンドが有効なのは，アンカが JMP 命令または CALL 命令のアドレス行にある場合だけです。このコマンドを選択すると，JMP 命令または CALL 命令のディスティネーション・アドレスとして指定されているアドレス行にアンカが移動します。

保守／廃止

# 6.7　Breakpoint メニュー

Breakpoint メニューの各コマンドは，Breakpoint ウインドウだけに適用されます。したがって，Breakpoint メニューの項目が使用できるのは，Breakpoint ウインドウがアクティブな場合だけです。

## 6.7.1　Show Header

Show Header コマンドは，Breakpoint ウインドウのヘッダ行表示のオンとオフを切り替えます。このコマンドの左にチェック・マーク（√）が付いている場合，ヘッダ行が表示されます。

## 6.7.2　Set At...

Set At...コマンドは，インストラクション・メモリ・アドレス，ソース行，またはタイミング・ファイル行に結び付いたブレークポイントの設定行を Breakpoint ウインドウに追加します。新しいブレークポイントの Address フィールドにアンカが移動し，インストラクション・メモリ・アドレス値，ソース行番号，またはタイミング・ファイルの行番号を入力できるようになります。

**注意　インストラクション・メモリ・アドレスがソース・ファイル行に関係している場合，次の構文でブレークポイントを指定できます。**

| |
|---|
| *source_file_name* ! *line_number* |

*source_file_name*：　ファイル拡張子を含むソース・ファイルまたはタイミング・ファイルの名前です。この入力では，大文字と小文字を区別する必要はありません。

*line_number*：対象となる行番号です。

## 6.7.3　Global Expression True…

Global Expression True...コマンドは，指定された式がゼロ以外の値の場合，つまり真の場合に起動するグローバル・ブレークポイントの設定行を Breakpoint ウインドウに追加します。

## 6.7.4　Global Expression Changed...

Global Expression Changed...コマンドは，指定された式に割り当てられた値が変更された場合に起動するグローバル・ブレークポイントの設定行を Breakpoint ウインドウに追加します。

### 6.7.5　Memory Read...

Memory Read...**コマンドは，指定されたアドレスのメモリ・リード・アクセスによって起動するグローバル・ブレークポイントの設定行を** Breakpoint **ウインドウに追加します。**

**メモリ・リード・ブレークポイント条件に従うアドレスの入力構文は，次のとおりです。**

| |
|---|
| *address* : *mem_type[,block_size]* |

*address*：　10 **進，**2 **進，**8 **進，**16 **進のいずれかの数値形式の** X **データ・メモリまたは** Y **データ・メモリのアドレスです。入力では，大文字と小文字を区別する必要はありません。**

*mem_type*：X **データ・メモリの場合は** X**，**Y **データ・メモリの場合は** Y **です。入力では，大文字と小文字を区別する必要はありません。**

*block_size*：**メモリ・ブロックの長さ（メモリ・ワード数）です。ディフォールトでは** 1 **になっています。**

### 6.7.6　Memory Write...

Memory Write...**コマンドは，指定されたアドレスのメモリ・ライト・アクセスによって起動されるグローバル・ブレークポイントの設定行を** Breakpoint **ウインドウに追加します。**

### 6.7.7　Memory Access...

Memory Access...**コマンドは，指定されたアドレスのメモリ・リード・アクセスまたはメモリ・ライト・アクセスによって起動するグローバル・ブレークポイントの設定行を** Breakpoint **ウインドウに追加します。**

**アドレスの入力構文については，**6.7.5　Memory Read...**を参照してください。**

### 6.7.8　Time...

Time...**コマンドは，指定された時間が経過したあとに起動するグローバル・ブレークポイントの設定行を** Breakpoint **ウインドウに追加します。**

### 6.7.9　Steps...

Steps...**コマンドは，指定された数のプロセッサ・ステップが実行されたあとに起動するグローバル・ブレークポイントの設定行を** Breakpoint **ウインドウに追加します。**

### 6.7.10　Error Bits set

Error Bits set **コマンドは，**ESR**（エラー・ステータス・レジスタ）内のいずれかのエラー・ビットがセットされたときに起動するグローバル・ブレークポイントの設定行を** Breakpoint **ウインドウに追加します。**

### 6.7.11　Add Condition | Expression True....

**このコマンドは，現在選択されているブレークポイントに式の真条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，式を入力してください。このブレークポイントが起動するのは，指定された式の評価がゼロ以外の定義値である場合です。**

保守／廃止

### 6.7.12　Add Condition | Expression Changed...

**このコマンドは，現在選択されているブレークポイントに式変更条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，式を入力してください。このブレークポイントが起動するのは，指定された式の値が変更された場合です。**

### 6.7.13　Add Condition | Memory Read...

**このコマンドは，現在選択されているブレークポイントにメモリ・リード条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，アドレスを入力してください。このブレークポイントが起動するのは，指定されたアドレス範囲が読み出された場合です。**

### 6.7.14　Add Condition | Memory Write...

**このコマンドは，現在選択されているブレークポイントにメモリ・ライト条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，アドレスを入力してください。このブレークポイントが起動するのは，指定されたアドレス範囲へのライト・アクセスが発生した場合です。**

### 6.7.15　Add Condition | Memory Access...

**このコマンドは，現在選択されているブレークポイントにメモリ・リードまたはメモリ・ライト条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，アドレスを入力してください。このブレークポイントが起動するのは，指定されたアドレス範囲でリード・アクセスまたはライト・アクセスが発生した場合です。**

### 6.7.16　Add Condition | Time...

**このコマンドは，現在選択されているブレークポイントに時間条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，ナノ秒（**ns**）単位で時間を入力してください。このブレークポイントが起動するのは，指定されたブレークポイントに達し，指定された時間が経過した場合です。**

### 6.7.17　Add Condition | Steps...

**このコマンドは，現在選択されているブレークポイントにステップ条件を追加します。アンカがそのブレークポイントの** Condition **フィールドに移動するので，プロセッサ・ステップ数を入力してください。このブレークポイントが起動するのは，指定されたブレークポイントに達し，指定された数のプロセッサ・ステップが実行された場合です。**

### 6.7.18　Add Condition | Error Bits set

**このコマンドは，現在選択されているブレークポイントにエラー・ビット・セット条件を追加します。このブレークポイントが起動するのは，指定されたアドレスに達し，**ESR**（エラー・ステータス・レジスタ）のエラー・ビットのいずれかがセットされた場合です。**

### 6.7.19　Add Action | Break

**このコマンドは，現在選択されているブレークポイントにブレーク動作を追加します。このブレークポイントが起動すると，実行が中断され，制御が** HSM77016 **に戻ります。**

**注意　この動作は** 1 **回だけ行われ，最後の動作としてリストされます。**

保守／廃止

### 6.7.20　Add Action | Execute...

このコマンドは，現在選択されているブレークポイントに実行動作を追加します。アンカがそのブレークポイントの Action フィールドに移動するので，評価する式を入力してください。このブレークポイントが起動すると，実行が中断され，入力された式（レジスタの値変更などの効果的な影響を生みます）が評価されます。

### 6.7.21　Add Action | Log...

このコマンドは，現在選択されているブレークポイントにログ動作を追加します。アンカがそのブレークポイントの Action フィールドに移動するので，式を入力してください。このブレークポイントが起動すると，実行が中断され，指定された式がログ・バッファに格納されます。

### 6.7.22　Add Action | Enable Group...

このコマンドは，現在選択されているブレークポイントにグループ有効化動作を追加します。アンカがそのブレークポイントの Action フィールドに移動するので，グループ番号を入力してください。このブレークポイントが起動すると，実行が中断され，指定されたブレークポイント・グループが有効になります。

### 6.7.23　Add Action | Disable Group...

このコマンドは，現在選択されているブレークポイントにグループ無効化動作を追加します。アンカがそのブレークポイントの Action フィールドに移動するので，グループ番号を入力してください。このブレークポイントが起動すると，実行が中断され，指定されたブレークポイント・グループが無効になります。

### 6.7.24　Add Action | Update Display

このコマンドは，現在選択されているブレークポイントに表示更新動作を追加します。このブレークポイントが起動するたびに，HSM77016 のデータ・ウインドウの内容が更新されます。この動作は，ブレークポイントに達してもシミュレーションのブレークを行わず（ブレーク動作の指定なし）に，データの変更をウォッチする場合に便利です。

### 6.7.25　Test Action

Test Action コマンドは，ブレークポイントの現在選択されている実行処理，グループ有効化処理，またはグループ無効化処理を行います。このコマンドが有効なのは，ブレークポイントの動作が選択されている場合だけです。

### 6.7.26　Remove All

Remove All コマンドは，現在定義されているすべてのブレークポイントを削除します。Breakpoint ウインドウには，“ No breakpoints defined ” と表示されます。

**注意　すべての定義済みブレークポイントが失われます。**Remove All **コマンドは取り消すことはできませんが，ブレークポイントは** Tools **メニューの** Options... **コマンドで保存，復元できます。**

保守／廃止

# 6.8　Log メニュー

Log メニューの各コマンドは，アクティブな Log ウインドウに適用されます。したがって，Log メニューが有効なのは，Log ウインドウがアクティブな場合だけです。

HSM77016 のログ機能の詳細については，5.5　Log ウインドウを参照してください。

## 6.8.1　New Session

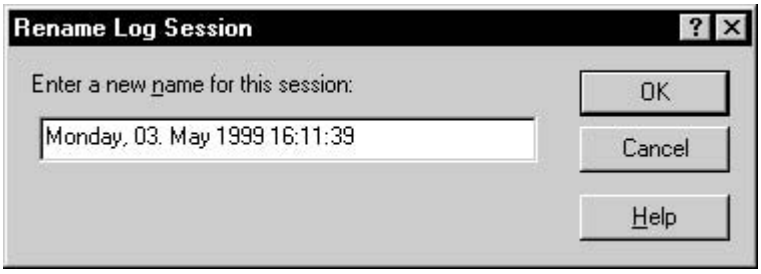
New Session コマンドは，現在のセッション項目を保存し，新しいログ・セッションを開始します。ログ・ファイルが指定されていない場合ファイル選択ダイアログ・ボックスが表示されるので，ログ・ファイル名を入力してください。現在のログ・セッションの名前が，Log ウインドウのタイトル・バーにかっこ（[　]）付きで表示されます。Log ウインドウの内容は，File メニューの Export...コマンドでテキスト・ファイルに保存できます。シミュレーション・セッションのログ・セッションを格納するログ・ファイルを指定するには，Tools メニューから Options...メニューを選択します。

## 6.8.2　Rename Session...

Rename Session...コマンドを選択すると，Rename Log Session ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，新しいログ・セッション名の入力に使用します。ディフォールトのセッション名は現在の日付けと時刻です。ログ・セッション・データを保存するには，File メニューから Export...コマンドを選択します。エミュレーション・セッションのログ・セッションを格納するログ・ファイルを指定するには，Tools メニューから Options...コマンドを選択します。

図 6－17 は Rename Log Session ダイアログ・ボックスです。

図 6－17　Rename Log Session ダイアログ・ボックス




（1）Rename Log Session ダイアログ・ボックスの構成要素

- Enter a new name for this session
  ログ・セッション名を入力します。編集フィールドには，ディフォールトのセッション名が表示されています。

- OK
  セッション名を確認してダイアログ・ボックスを閉じます。

- Cancel
  元のセッション名を保持してダイアログ・ボックスを閉じます。

### 6.8.3　Clear Session

Clear Session コマンドは，Log ウインドウから現在のログ・セッションのすべてのイベント項目を削除します。ログ・セッション名は保持されます。ログ・セッション・データを保存するには，File メニューから Export...コマンドを選択します。シミュレーション・セッションのログ・セッションを格納するログ・ファイルを指定するには，Tools メニューから Options...コマンドを選択します。

## 6.9　Watch メニュー

Watch メニューの各コマンドは，Watch ウインドウだけに適用されます。したがって，Watch メニューの項目が有効なのは，Watch ウインドウがアクティブな場合だけです。

コマンドは，Watch ウインドウの Diagram ブロックの空き領域を右クリックして表示されるコンテキスト・メニューから選択することもできます。

### 6.9.1　Show Header

Show Header コマンドは，ヘッダ行表示のオンとオフを切り替えます。ヘッダ行を表示しなければ，画面のスペースを節約できます。このコマンドの左にチェック・マーク（√）が付いている場合，ヘッダ行が表示されます。

### 6.9.2　Show Signals

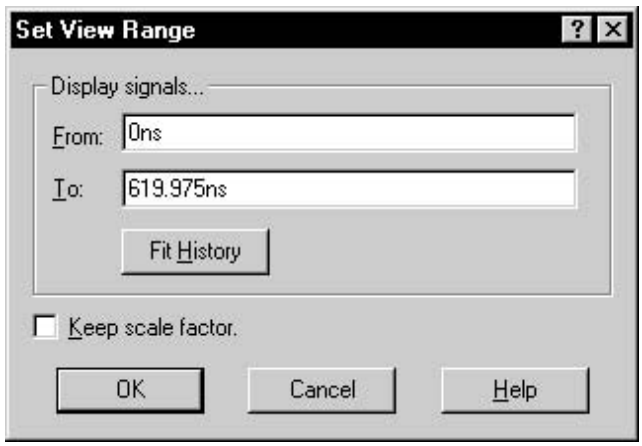
Show Signals コマンドは,Watch ウインドウの Diagram ブロック表示のオンとオフを切り替えます。このブロックには絶対タイム・スケールと相対タイム・スケールが表示されます。このコマンドの左にチェック・マーク（√）が付いている場合，Diagram ブロックが表示されます。

### 6.9.3　Set View Range...

このコマンドは，Set View Range ダイアログ・ボックスを開きます。このダイアログ・ボックスは，Watch ウインドウの Diagram ブロック内に移動したり，表示内容を変更するために使用します。

図 6－18 は Set View Range ダイアログ・ボックスです。

図 6－18　Set View Range ダイアログ・ボックス



保守／廃止

（1）Set View Range **ダイアログ・ボックスの構成要素**

- From
  Diagram **ブロックの左側に表示される希望開始時間を入力します。**

- To
  Diagram **ブロックの右側に表示される終了時間を入力します。ズーム倍率は，開始時間と終了時間の値の差によって決まります。**

- Fit History
  Display **ブロックのシグナル・ヒストリ表示を調整します。**

- Keep scale factor
  **チェック・マークが付いている場合，同じ表示ズーム倍率になるようにビューの終了時間が自動的に再計算されます。チェック・マークが付いていない場合，ビュー時間はそれぞれ個別に変わります。**

- OK
  **入力したビュー時間がそれぞれ一貫しているかどうか確認します。表示できない場合，現在のズーム倍率が保持され，通知なしで実際の時間が表示されるようにビューが移動します。**

- Cancel
  **すべての入力を無視します。**

### 6.9.4　Goto Current Time

**このコマンドは，ウインドウの内容を現在の時間に移動します。ズーム倍率は変更されません。**

### 6.9.5　Zoom In

**このコマンドは，ビューを** 2 **倍に拡大表示（ズーム・イン）します。ビューの中央が縮尺の中心になります。**

### 6.9.6　Zoom Out

**このコマンドは，ビューを** 2 **分の** 1 **に縮小表示（ズーム・アウト）します。ビューの中央が縮尺の中心になります。**

### 6.9.7　Zoom Tool

**このコマンドは，ズーム・インまたはズーム・アウトを可能にするズーム・ツールをアクティブにします。**Shift **キーを押すとズーム・アウトに，**Shift **キーを押さないとズーム・インになります。カーソルを** Signal Display **ブロックに移動すると，カーソル形状が虫眼鏡（＋／－）に変わります。**

Shift **キーを押さないと，ズーム・イン・モードがアクティブになります。この時点で，以後拡大表示する時間範囲を選択できます。一定の時間範囲を選択しない場合は，マウスを左クリックします。この動作で，**2 **倍拡大表示されます。**

Shift **キーを押すと，ズーム・アウト・モードがアクティブになります。この時点で，縮小表示する時間範囲を端数で選択できます。ズーム倍率は，以前に表示された時間範囲を選択した端数で除算した値です。一定の時間範囲を選択しない場合は，マウスを左クリックします。この動作で，**2 **分の** 1 **に縮小表示されます。**

保守／廃止

### 6.9.8　Measure Tool

このコマンドは，マークされたポイントと Signal Display ツール・バー（マウスを左クリックした場合はツール・チップ）の現在のカーソル位置との時間差を表示します。この測定ツールが有効なのは，ズーム・ツールが選択され，タイム・カーソル表示がオンになっている場合です。測定ツールが有効な場合にカーソルが Signal Display ブロックに移動すると，カーソル形状が測定カーソルに変わります。マウスを左クリックすると，現在のカーソル位置に破線が表示されます。マウスの左ボタンを押したままカーソルを任意の位置に移動します。マウス・ボタンを離すと，次の動作が実行されるまで，表示が停止します。

### 6.9.9　Time Marker

このコマンドは，マーカ表示のオンとオフを切り替えます。このコマンドの左にチェック・マークが付いている場合，マーカが表示されます。

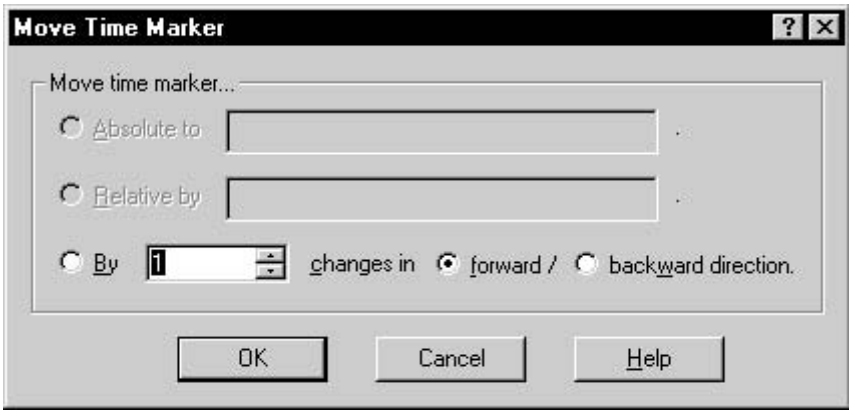
### 6.9.10　Move To...

このコマンドを選択すると，Move Time Marker ダイアログ・ボックスが表示されます。このダイアログ・ボックスを使用して，Watch ウインドウの Diagram ブロックのタイム・マーカをブラウズできます。このコマンドはタイム・マーカが表示されているときのみ有効です。

このコマンドを選択したときにヒストリ領域の外側にタイム・マーカがある場合，タイム・マーカがヒストリ領域の内部に移動します。

図 6－19 は Move Time Marker ダイアログ・ボックスです。

図 6－19　Move Time Marker ダイアログ・ボックス




（1）Move Time Marker ダイアログ・ボックスの構成要素

- Absolute to
  タイム・マーカを絶対時間に移動します。

- Relative by
  タイム・マーカを入力された時間値だけ移動します。

- By ... changes in forward/ backward direction
  （現在のタイム・マーカ位置から）スキップしたいシグナル変更数を入力できます。現在のタイム・マーカ位置の前方でも後方でも移動できます。このオプションを選択できるのは，Move Time Marker ダイアログ・ボックスが開いたあとにシグナルが選択されている場合だけです。

- OK
  **ダイアログ・ボックスを閉じ，選択されたシグナル変更を** Watch **ウインドウに表示します。**

- Cancel
  Watch **ウインドウの現在のタイム・マーカの位置を変更しないで，ダイアログ・ボックスを閉じます。**

### 6.9.11　Move to Next Change

**このコマンドは，選択されたシグナルの次回の変更にマーカを移動します。必要に応じてビューの位置が変わります。**

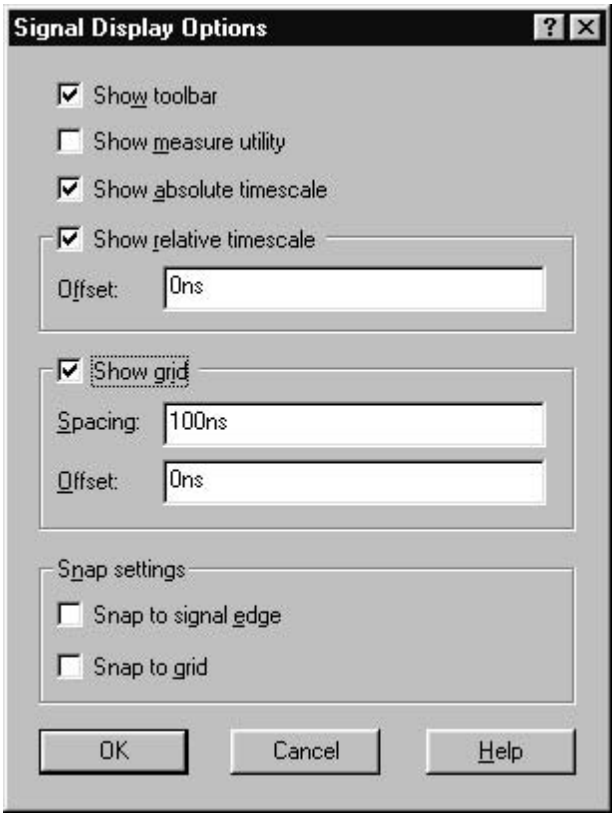
### 6.9.12　Move to Prev Change

**このコマンドは，選択されたシグナルの前回の変更にマーカを移動します。必要に応じてビューの位置が変わります。**

### 6.9.13　Signal Options...

**このコマンドを選択すると，**Diagram **ブロックに** Signal Display Options **ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，様々な設定を行うことができます。図** 6－20 **は** Signal Display Options **ダイアログ・ボックスです。**

**図** 6－20　Signal Display Options **ダイアログ・ボックス**



保守／廃止

（1）Signal Display Options ダイアログ・ボックスの構成要素

- Show toolbar

  Signal Display ツール・バーを表示します。このツール・バーを使用すれば，クリックするだけで頻繁に使用する Watch メニューのコマンドに素早くアクセスできます。このツール・バーのシグナル表示フィールドには，マークされたポイントとカーソルの現在位置の時間差が表示されます。Signal Display Options ダイアログ・ボックスで，ツール・バーの表示のオンとオフを切り替えることができます。

- Show measure utility

  このオプションにチェック・マークが付いていて，ズーム・ツールが選択されていない場合，測定ツールが有効です。この状態でカーソルが Watch ウインドウの Signal Display ブロックに移動すると，測定カーソルに変わります。マウスを左クリックすると，現在のカーソル位置に破線が付きます。マウスのボタンを離すと，次の動作が実行されるまで表示が停止します。測定ツールによって，マークされたポイントと Signal Display ツール・バーまたはツール・チップの現在のカーソル位置との時間差が表示されます。

- Show absolute timescale

  このオプションにチェック・マークが付いている場合，絶対タイム・スケールが表示されます。

- Show relative timescale

  このオプションにチェック・マークが付いている場合，相対タイム・スケールが表示されます。

  Offset：絶対タイム・スケールに時間オフセットを入力できます。

- Show grid

  このオプションにチェック・マークが付いている場合，垂直のグリッド線が表示されます。グリッド線同士の間隔が小さすぎる場合は，1 本しか表示されません。

  Spacing：2 本のグリッド線間の時間値を入力できます。

  Offset：時間オフセットの値を入力できます。

- Snap to signal edge

  このオプションにチェック・マークが付いている場合，相対タイム・スケールを最も近いシグナル・エッジにスナップできます。

- Snap to grid

  このオプションにチェック・マークが付いている場合，相対タイム・スケールを最も近いグリッド線にスナップできます（グリッド線がオンになっている場合）。

- OK

  設定を確認してダイアログ・ボックスを閉じます。

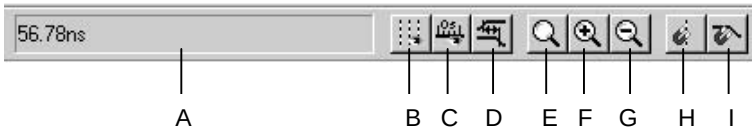
- Cancel

  新しい設定を破棄してダイアログ・ボックスを閉じます。

図 6－21 は Signal Display ツール・バーです。

保守／廃止

図 6－21　Signal Display ツール・バー




A：Time Display フィールド
B：Grid ボタン
C：Relative Time Scale ボタン
D：Measure ボタン
E：Zoom Tool ボタン
F：Zoom In ボタン
G：Zoom Out ボタン
H：Snap to Grid ボタン
I：Snap to Signal Edge ボタン

（2）Signal Display ツール・バーのボタンとフィールド

Timer Display フィールド：マーカとカーソル間の時間を表示します。
Grid ボタン：グリッド線表示のオンとオフを切り替えます。
Relative Time Scale ボタン：相対タイム・スケール表示のオンとオフを切り替えます。
Measure ボタン：測定ツール表示のオンとオフを切り替えます。
Zoom Tool ボタン：ズーム・ツールをアクティブにします。
Zoom In ボタン：ズーム・インします。
Zoom Out ボタン：ズーム・アウトします
Snap to Grid ボタン：最も近いグリッド線にスナップします。
Snap to Signal Edge ボタン：最も近いシグナル・エッジにスナップします。

### 6.9.14　Add Watch

Add Watch コマンドは，Watch ウインドウに新しい行を追加します。アンカが新しい行の Expression 列に移動するので，ウォッチする式を入力してください。また，Insert キーを押しても新しい行を追加できます。

（1）Watch エントリからの削除

現在選択されている行を Watch ウインドウから削除するには，Edit メニューから Delete コマンドを選択するか，Delete キーを押します。

### 6.9.15　Remove All

Remove All コマンドは，Watch ウインドウ内のウォッチ式をすべて削除します。Watch ウインドウの設定は，Tools メニューの Options...で保存できます。

保守／廃止

# 6.10　Statistic メニュー

Statistic メニューには，プロファイリング・マーカを管理し，ユーザの必要に応じて統計データを配置するコマンドが含まれています。プロファイリング・マーカとブレークポイントを同じアドレスに設定することはできません。ブレークポイントが設定されているアドレスにマーカが設定された場合，既存のブレークポイントを指定のプロファイリング・マーカに置き換えるようダイアログ・ボックスにメッセージが表示されます。

## 6.10.1　Show Header

このコマンドは，ヘッダ行の表示のオンとオフを切り替えます。このコマンドの左にチェック・マーク（√）が付いている場合，ヘッダ行が表示されます。

## 6.10.2　Set Marker At...

このコマンドは，インストラクション・メモリ・アドレスへのマーカの設定行を Statistic ウインドウに追加します。アンカが新しいマーカ行の Address フィールドに移動するので，アドレス値を入力してください。

## 6.10.3　Set Entry Marker At...

このコマンドは，インストラクション・メモリ・アドレスへの Entry マーカの設定行を Statistic ウインドウに追加します。アンカが新しいマーカ行の Address フィールドに移動するので，アドレス値を入力してください。

## 6.10.4　Set Exit Marker At...

このコマンドは，インストラクション・メモリ・アドレスへの Exit マーカの設定行を Statistic ウインドウに追加します。アンカが新しいマーカ行の Address フィールドに移動するので，アドレス値を入力してください。

## 6.10.5　Set Start Marker At...

このコマンドは，インストラクション・メモリ・アドレスへの Start マーカの設定行を Statistic ウインドウに追加します。アンカが新しいマーカ行の Address フィールドに移動するので，アドレス値を入力してください。

## 6.10.6　Set Stop Marker At...

このコマンドは，インストラクション・メモリ・アドレスへの Stop マーカの設定行を Statistic ウインドウに追加します。アンカが新しいマーカ行の Address フィールドに移動するので，アドレス値を入力してください。

## 6.10.7　Sort By Type

このコマンドは，Statistic ウインドウ内のデータを，マーカの種類別に並び替えます。このコマンドの左にチェック・マーク（√）が付いている場合，種類別に並び替えをしています。

## 6.10.8　Sort By Address

このコマンドは，Statistic ウインドウ内のデータを，マーカのアドレス値順に並び替えます。このコマンドの左にチェック・マーク（√）が付いている場合，アドレス値順に並び替えをしています。

## 6.10.9　Sort By Count

このコマンドは，Statistic ウインドウ内のデータを，マーカの通過回数順に並び替えます。このコマンドの左にチェック・マーク（√）が付いている場合，マーカの通過回数順に並び替えをしています。

保守／廃止

### 6.10.10　Sort By Time

このコマンドは，Statistic ウインドウ内のデータを，マーカの時間カウント値順に並び替えます。このコマンドの左にチェック・マーク（√）が付いている場合，マーカの時間カウント値順に並び替えをしています。

### 6.10.11　Sort By Time Per Call

このコマンドは，Statistic ウインドウ内のデータを，マーカの Time/Call 値順に並び替えます。このコマンドの左にチェック・マーク（√）が付いている場合，マーカの Time/Call 値順に並び替えをしています。

### 6.10.12　Sort Ascending

このコマンドは，Statistic ウインドウ内のデータを，選択されたソート・タイプに従って昇順に並び替えます。このコマンドの左にチェック・マーク（√）が付いていれば昇順に並び替え，付いていなければ降順に並び替えます。

### 6.10.13　Entry Marker Stack...

このコマンドを選択すると，Active Profile Entry Markers ダイアログ・ボックスが表示されます。このダイアログ・ボックスには，現在スタック上にあるすべての Entry マーカが表示されます。このコマンドを使用するには，スタック上に少なくとも 1 つの Entry マーカがなければなりません。

### 6.10.14　Reset Statistic

このコマンドは，Statistic ウインドウ，Instruction Memory ウインドウ，Module ウインドウ内のマーカの通過回数および時間カウント値をすべてゼロにリセットします。

### 6.10.15　Remove All

このコマンドは，現在定義されているプロファイリング・マーカをすべて削除します。削除されると，Statistic ウインドウに “ No markers defined. ” と表示されます。定義されているすべてのマーカが失われるので，注意してください。このコマンドを一度選択すると，取り消しできません。しかし，Tools メニューの Options...コマンドでマーカを保存または復元できます。

### 6.10.16 Report...

このコマンドを選択すると，標準のファイル選択ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，Statistic ウインドウのデータを保存するレポート・ファイルを作成，選択します。レポート・ファイルの詳細については，4.5　**レポート・ファイル**を参照してください。

保守／廃止

# 6.11　Module メニュー

Module メニューの各コマンドは，Module ウインドウだけに適用されます。したがって，Module メニューの項目が使用できるのは，Module ウインドウがアクティブな場合だけです。

Module ウインドウの次の操作方法は Instruction Memory ウインドウと同じです。

- プロファイリング・マーカの設定と削除
- プロファイリング・データの検査

プロファイラの Module メニューのコマンド（Toggle Marker，Toggle Entry Marker，Toggle Exit Marker，Toggle Start Marker，Toggle Stop Marker）については，6.6.7　Toggle Marker から 6.6.11　Toggle Stop Marker を参照してください。

## 6.11.1　Pointer...

Pointer...コマンドを選択すると，Pointers ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，プログラムの実行中に式をウォッチするポインタを独自に定義するために使用します。ポインタは，Module ウインドウの行番号カラムの右に（▶）で表示されます。ポインタ式を区別するために，各ポインタにそれぞれ異なる色を割り当てることができます。

**注意**　Module ウインドウで行ったポインタの設定変更は，すべての Module ウインドウに適用されます。Pointers ダイアログ・ボックスの Pointer...コマンドの詳細については，6.6.1　Pointer...を参照してください。

## 6.11.2　Toggle Breakpoint

このコマンドは，現在のカーソル位置に実行ブレークポイントの有効／無効を切り替えます。

- カーソル位置に実行ブレークポイントがない場合，このアドレスに実行ブレークポイントが設定され有効になります。
- カーソル位置に実行ブレークポイントがある場合，ブレークポイントの状態が有効なら無効に，無効なら有効になります。

# 6.12　Timing File メニュー

Timing File メニューの各コマンドは,現在アクティブな Timing File ウインドウに表示されているタイミング･ファイルに適用されます。したがって，このメニューが表示されるのは，Timing File ウインドウがアクティブな場合だけです。

保守／廃止

### 6.12.1　Show Header

Show Header **コマンドは，ヘッダ行表示のオンとオフを切り替えます。ヘッダ行を表示しないと，画面のスペースを節約できます。このコマンドの左にチェック・マーク（√）が付いている場合，ヘッダ行が表示されます。**

### 6.12.2　Toggle Breakpoint

Toggle Breakpoint **コマンドは，カーソル位置での明示的タイミング・ファイル・ブレークポイントの有効／無効を切り替えます。**

### 6.12.3　Suspend

Suspend **コマンドは，タイミング・ファイルのサスペンド・モードのオンとオフを切り替えます。サスペンド・モードでは，影響を受けたタイミング・ファイルの実行が停止されます。このコマンドを有効にするには，コマンドの左にチェック・マーク（√）を付けます。**

## 6.13　Tools メニュー

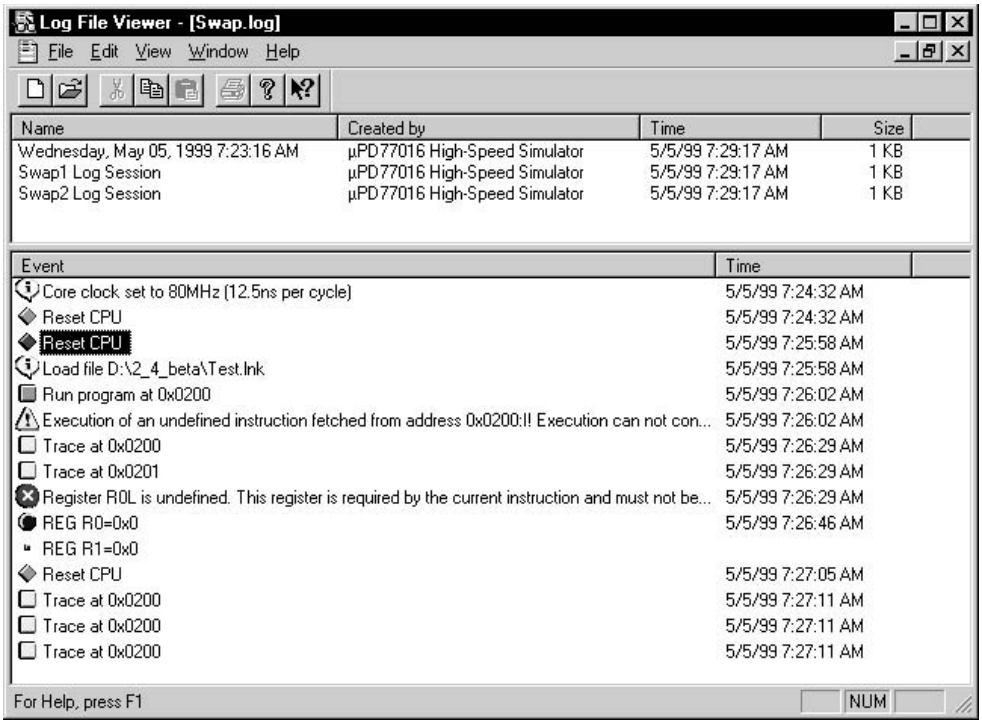
Tools **メニューの各コマンドは，**Log File Viewer **アプリケーションの開始，ユーザ・インタフェース設定の変更，シミュレーション・オプションの設定，シミュレータ・セッション設定の保存，およびシミュレーション・モデルの選択に使用します。**

### 6.13.1　Log Viewer...

Log Viewer...**コマンドを選択すると，**Log File Viewer **ツールが起動します。このツールは，**DSP **ツールで生成したログ・ファイルの検査と管理に使用します。**Log File Viewer **ツールは，**HSM77016 **から独立したスタンドアロン・アプリケーションです。したがって，**Log File Viewer **を終了しても，起動したツールは影響を受けません。**

**図** 6－22 **が** Log File Viewer **ツールです。**

**図** 6－22　Log File Viewer **ツール**



保守／廃止

### 6.13.2　Log File Viewer ツール

Log File Viewer のデータ・ウインドウにログ・ファイルが表示されます。このデータ・ウインドウは，セッション部分とイベント部分に分かれています。セッション部分には，ログ・ファイルに含まれるログ・セッションに関する情報が表示されます。

- Name
  セッション名が表示されます。

- Created by
  ログ・セッションの作成されたアプリケーション名が表示されます。

- Time（セッション部）
  ログ・セッションの作成日時が表示されます。
  Log Viewer データ・ウインドウのイベント部分には，選択されたセッションに関する情報が表示されます。

- Size
  ログ・ファイルのサイズが表示されます。

- Event
  ログ・イベントが表示されます。

- Times（イベント部）
  ログ・イベントが発生した日時が表示されます。

### 6.13.3　Log Viewer の機能

Log Viewer の機能には，ログ・ファイル機能，セッション機能，イベント機能があります。

**（1）ログ・ファイル機能**

ログ・ファイル機能には，現在アクティブなデータ・ウインドウのログ・ファイルに関連する File メニューのコマンド（New コマンドと Exit コマンドは例外）で実行される動作が含まれます。New コマンドは，ディフォールトのファイル名で空のログ・ファイルを作成します。ログ・ファイル機能に関連するコマンドは次のとおりです。

- Open...
  既存のログ・ファイルを開きます。

- Close
  現在アクティブなデータ・ウインドウのログ・ファイルを閉じ，それまで行われた変更をすべて保存します。

- Save As...
  現在アクティブなデータ・ウインドウのログ・ファイルを新しい名前で保存します。

保守／廃止

**（2）セッション機能**

Edit メニューのコマンドと File メニューの Print コマンドで，セッション機能を実行します。セッション機能で次の処理ができます。

- 選択したセッションを別のログ・ファイルに移動またはコピーします。
- 選択したセッションをログ・ファイルから削除します。
- 選択したセッションの印刷プレビューを表示します。
- 選択したセッションを印刷します。

セッションに関連するコマンドは次のとおりです。

- Cut
  選択したセッションをログ・ファイルから削除してクリップ・ボードに移します。

- Copy
  選択したログ・セッションをクリップ・ボードに移します。選択したセッションは削除されません。

- Paste
  クリップ・ボードからログ・セッションを現在の選択位置に挿入するか，ログ・ファイルに追加します。

- Print...
  選択したセッションを印刷します。セッションが選択されていない場合，このコマンドは無効です。

- Print Preview
  選択したログ・セッションを画面上に表示します。

**（3）イベント機能**

イベント機能によって，選択したイベントをテキスト形式でクリップ・ボードにコピーできます。

### 6.13.4　Language | Assembler

このコマンドは，アセンブラ言語パーサに切り替えます。式は，アセンブラ言語規則に従って評価されます。左にチェック・マーク（√）が付いている場合，このコマンドが有効です。

### 6.13.5　Language | C

このコマンドは，C 言語パーサに切り替えます。式は，ANSI C 言語規則に従って評価されます。左にチェック・マーク（√）が付いている場合，このコマンドが有効です。このコマンドが使用できるのは，ロードされたリンク・ファイルに C ディバグ情報が含まれている場合だけです。

### 6.13.6　Simulation Model...

シミュレーション・モデルには，シミュレーション対象プロセッサの種類，クロック周波数，およびメモリ仕様が含まれます。シミュレーション・モデルは，テキスト形式のモデル・ファイルに格納されます。シミュレーション・モデルを変更するには，Model Wizard ボタンをクリックして，モデル・ファイルの該当項目を変更します。シミュレーション・モデル・ファイルの構文の詳細については，第 7 章　モデルを参照してください。

Target System Model ダイアログ・ボックスで，現在のモデルの確認，モデルの主要なパラメータの表示，モデル・ファイルで定義されている特定のシミュレーション・モデルのインストールを実行できます。

**注意　このダイアログ・ボックスでは，既存の共通モデル定義ファイルを開くことや，変更することもできます。**

ID77016 で Target System Model ダイアログ・ボックスを表示した場合，このダイアログ・ボックスには Connected to フィールドが含まれます。このフィールドで，ディバグ・デバイスを選択できます。

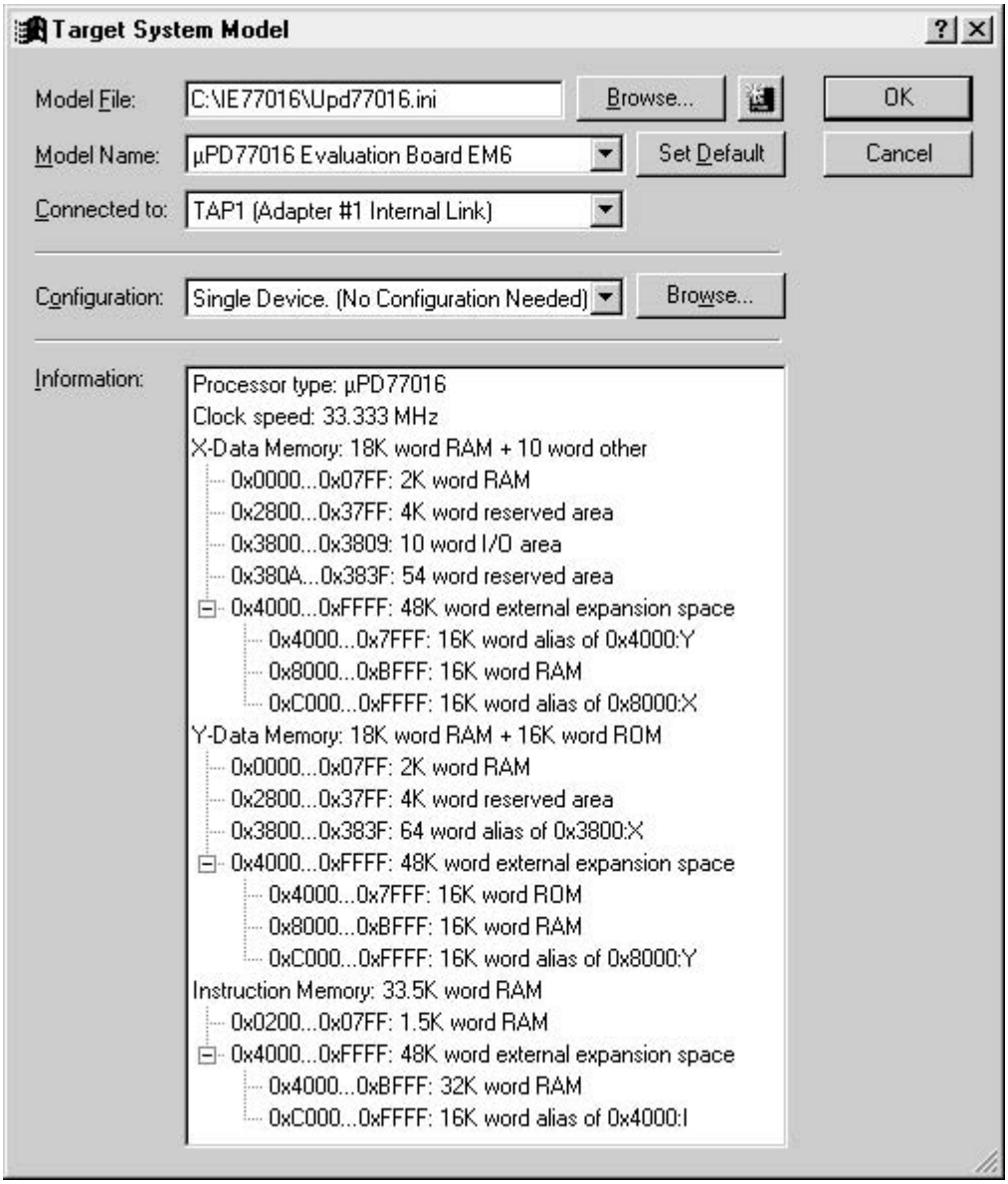
複数の JTAG デバイスをサポートするライブラリ[注]が IE-77016-98/PC に接続されている場合，Target System Model ダイアログ・ボックスには，HSDL（Hierarchical Scan Data Language：階層スキャン・データ言語）ファイルを指定する Configuration コントロールと，JTAG ユニット識別子を選択する Unit Identifier コントロールが表示されます。複数の JTAG デバイス・ライブラリと必要な HSDL ファイル入力の詳細については，Atair 社の Multi Device JTAG Extension マニュアルを参照してください。

**注　開発中**

図 6－23 は Target System Model ダイアログ・ボックスです。

保守／廃止

**図** 6ー23　Target System Model **ダイアログ・ボックス**




**（1）** Target System Model **ダイアログの構成要素**

- Model File

  **このフィールドには，現在使用中のモデル・ファイルの名前とパスが表示されます。**

- Browse...

  **モデル・ファイルがない場合または定義ファイルに目的のモデルが含まれていない場合は，別の定義ファイルを選択する必要があります。ファイルを選択するためのダイアログ・ボックスが表示されます。**

- Model Wizard

  **このボタンを選択すると，次の** Model **ウィザード起動用のサブ・コマンドを指定できます。**

  New...

  Model **ウィザードを起動して新しいモデルを作成します。このサブ・コマンドを選択すると，任意の** DSP **を対象とする新しいモデルを作成できます。その他のモデル・プロパティは，ユーザが指定します。**Model **ウィザードが終了すると，ファイル選択ダイアログ・ボックスが表示されるので，モデルの名前と格納場所を指定してください。**

  Clone...

  Model **ウィザードを起動して現在アクティブなモデルの複製を作成します。ユーザが属性を変更しないかぎり，複製モデルは元のモデルの属性をすべて引き継ぎます。**Model **ウィザードが終了すると，ファイル選択ダイアログ・ボックスが表示されるので，モデルの名前と格納場所を指定してください。**

  Edit...

  Model **ウィザードを起動して現在アクティブなモデルを編集します。現在のモデルは，編集後のモデルに置き換えられます。**

  Rename...

  **モデル定義ファイルの現在のモデルの名前を変更します。**

  Delete

  **モデル定義ファイルから現在のモデルを削除します。**

- Model Name

  **現在モデル定義ファイルに記述されているモデルの名前が表示されます。目的のモデルが表示されている行をクリックすれば，そのモデルを選択できます。このフィールドを利用できるのは，モデル・ファイルではなくてモデル定義ファイルを使用している場合です。**

- Set Default

  **モデル定義ファイル内の選択されたファイルをディフォールト・モデルとして設定できます。このモデルは，起動時に初期モデルとしてロードされます。**

- Connected to

  Connected to **リストには，使用できるディバグ・デバイス接続（テスト用アクセス・ポート）が表示されます。エミュレーションを正しく実行し，最良のディバグ・パフォーマンスを実現するには，**DSP **タイプ，**DSP **への接続，ターゲット・システムの構成と機能を** ID77016 **が認識している必要があります。これらの情報は，**debuggee **のデバイス接続に含まれています。接続デバイスは，外付けデバイス（シリアル・バス・ケーブルで** IE-77016-98/PC **に接続されたターゲット** DSP**）と内蔵デバイス（**IE-77016-98/PC **に搭載されたターゲット** DSP**）のどちらでもかまいません。**

- Configuration

  このフィールドには，現在使用中の HSDL（Hierarchical Scan Data Language: 階層スキャン・データ言語）ファイルの名前とパスが表示されます。HSDL ファイルには，IE-77016-98/PC に接続されたターゲット・システムに関する情報が記述されています。このフィールドが利用できるのは，複数の JTAG デバイス・ライブラリが IE-77016-98/PC に接続されている場合です。複数の JTAG デバイス・ライブラリが接続されているのに，IE-77016-98/PC を 1 つのデバイスで使用しなければならない場合は，ドロップダウン・リストから “ Single Device. (No Configuration Needed) ” 項目を選択します。

- Browse...

  このボタンで，HSDL ファイルをブラウズできます。このボタンを利用できるのは，複数の JTAG デバイス・ライブラリが IE-77016-98/PC に接続されている場合だけです。

- Unit Identifier

  このリストは，IE-77016-98/PC に接続されているターゲット・システム上の JTAG デバイスに割り当てられたユニット ID を選択するために使用します。このボタンを利用できるのは，複数の JTAG デバイス・ライブラリが IE-77016-98/PC に接続されていて，1 つ以上のデバイスに関する情報を含む HSDL ファイルがロードされている場合だけです。

- Information

  現在選択されているモデルの概要が表示されます。表示される情報の範囲はモデルによって異なりますが，少なくても，選択されたチップ・タイプとクロック速度に関する情報は含まれています。補足情報として，外部メモリなどに関するその他の重要情報を表示することもできます。このリストは表示されるだけです。該当するメモリ領域が表示されている行をダブルクリックすれば，メモリ・マップ・ビューを拡張表示したり非表示にすることができます。拡張表示したメモリ・マップ・ビューには，外部メモリと内蔵メモリ，エイリアス，I/O，および予約メモリの開始アドレス，終了アドレスと総ワード数が表示されます。

- OK

  新しいモデルを選択してダイアログ・ボックスを閉じます。

- Cancel

  処理を取り消してダイアログ・ボックスを閉じます。前回選択したモデルは有効なままです。

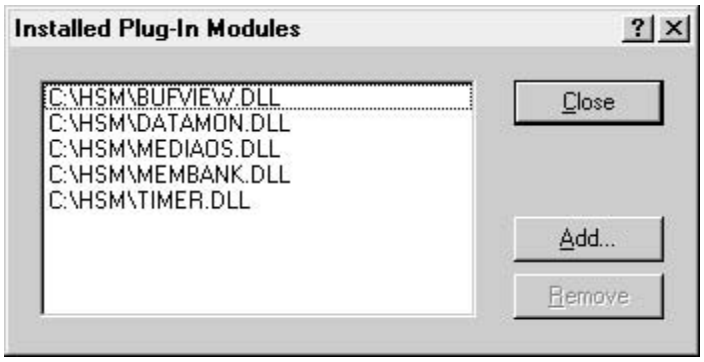
保守／廃止

### 6.13.7　Installed Plug-In Modules...

このダイアログ・ボックスで，外部ライブラリを検査，追加，削除します。このダイアログ・ボックスには，カスタム機能をシミュレーションに接続するための HSM77016 のアプリケーション・インタフェースが表示されます。アプリケーション・インタフェースの詳細については，オンライン・ヘルプを参照してください。

図 6－24 は Installed Plug-In Modules ダイアログ・ボックスです。

図 6－24　Installed Plug-In Modules ダイアログ・ボックス




（1）Installed Plug-In Modules ダイアログ・ボックスの構成要素

- Plug-In Modules List
  現在インストールされているプラグイン・モジュールが表示されます。

- Add...
  このボタンを選択すると，プラグイン・モジュールをブラウズするためのファイル選択ダイアログ・ボックスが表示されます。

- Remove
  選択したプラグイン・モジュールをシミュレーションから削除します。

- Close
  ダイアログ・ボックスを閉じます。

### 6.13.8　Options...

Options...コマンドを選択すると，Options ダイアログ・ボックスが表示されます。このダイアログ・ボックスは，シミュレーション実行とユーザ・インタフェース項目を調整するために使用します。

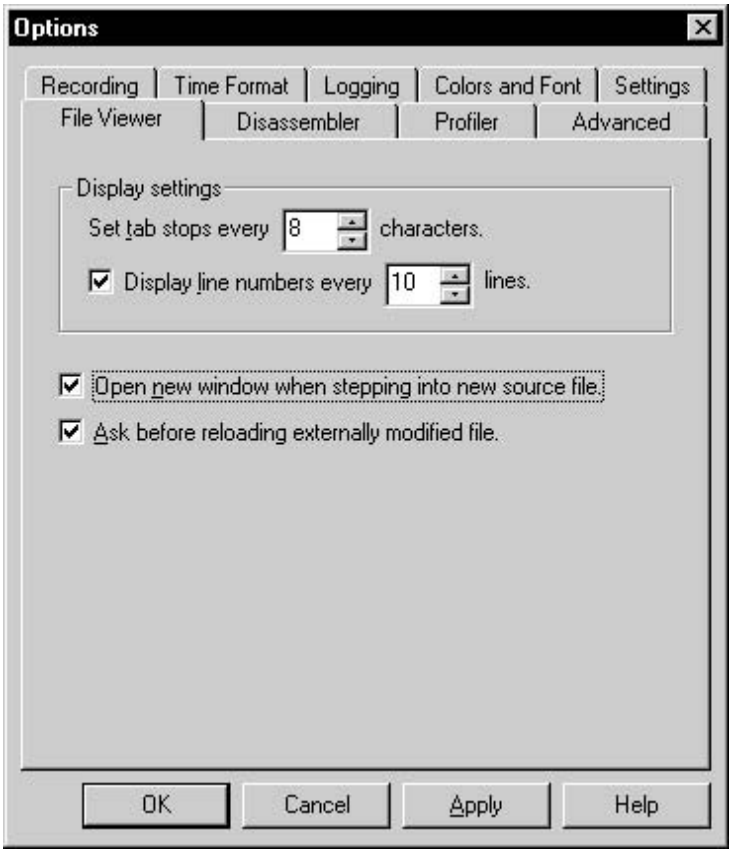
（1）File Viewer タブ

File Viewer タブで，テキスト・ウインドウの表示方法を設定したり変更できます。

図 6－25 は File Viewer タブです。

保守／廃止

**図** 6ー25　File Viewer **タブ（**Options **ダイアログ・ボックス）**




**（**2**）**File Viewer **タブの構成要素**

- Set tab stops every ... characters.
  **タブ間隔を設定します。**1 **から** 16 **までの整数値を指定できます。**

- Display line numbers every ... lines.
  View **ウインドウ，**Module **ウインドウ，**Timing File **ウインドウでの行番号表示の単位を設定します。指定できる範囲は，**1**（各行に行番号を表示）から** 100**（**100 **行ごとに行番号を表示）までです。このボックスにチェック・マークが付いていないと，行番号指定は無効になります。**

- Open new window when stepping into new source file.
  **このオプションにチェック・マークが付いている場合，まだロードされていないソース・ファイルにソース・レベル・シミュレーションが達すると，**HSM77016 **が新しいウインドウを開きます。**

- Ask before reloading externally modified file.
  **このボックスにチェック・マークがついている場合，現在ロードされているファイルが別のアプリケーションで変更されるとメッセージが表示されます。チェック・マークがついていない場合，メッセージなしに，変更されたファイルが再ロードされます。**
  **ダイアログ・ボックスを閉じると，新しい設定が有効になります。**

- Cancel
  **新たに設定した内容を拒否してダイアログ・ボックスを閉じます。**

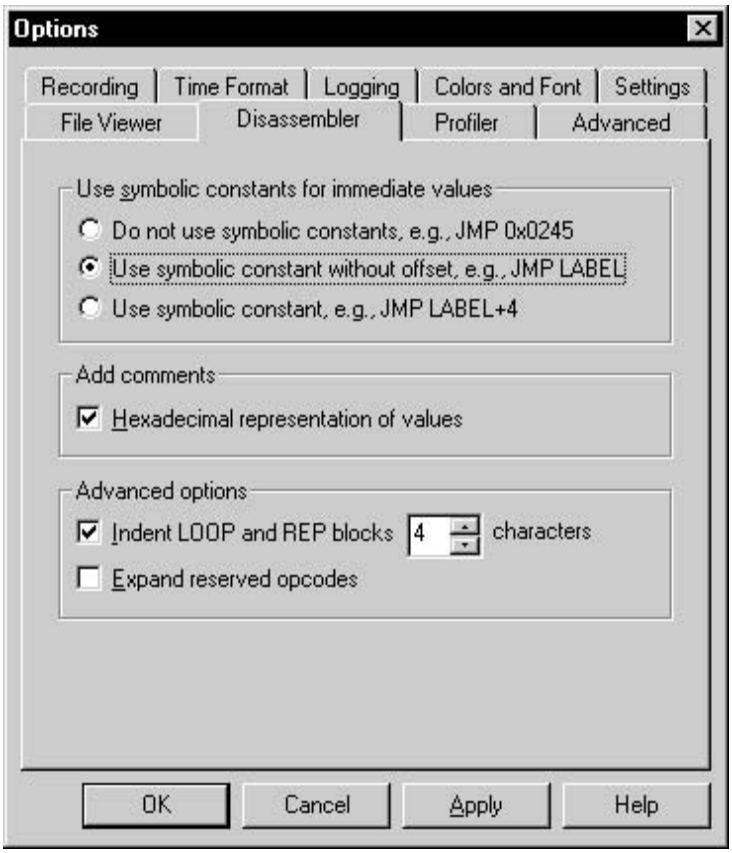
保守／廃止

(3) Disassembler タブ

HSM77016 では，インストラクション・メモリ空間にある命令を表示するために内蔵逆アセンブラが使用されます。逆アセンブラは，μPD77016 命令をニモニック形式に逆変換します。Disassembler タブの設定によって，ニモニック形式で表示されるビット・パターンが決まります。

図 6－26 は Disassembler タブです。

図 6－26　Disassembler タブ（Options ダイアログ・ボックス）




(4) Disassembler タブの構成要素

- Do not use symbolic constants, e.g., JMP 0x245

  このオプションを選択すると，シンボル名と数値が一致する場合でも，数値の代わりにシンボル名は使用されません。

- Use symbolic constant without offset, e.g., JMP LABEL

  このオプションを選択すると，シンボル名と数値が正確に一致する場合だけ，数値の代わりにシンボル名が使用されます。

- Use symbolic constant, e.g., JMP LABEL+4

  このオプションを選択すると，可能な場合に数値の代わりにシンボル名と式が使用されます。シンボル値が数値と一致しない場合は，シンボル値は整数に置き換えられます。

- Hexadecimal representation of values

  **このボックスにチェック・マークが付いている場合，逆アセンブルされた式の** 16 **進値がコメントとして表示されます。**

- Indent LOOP and REP blocks ... characters

  **このボックスにチェック・マークが付いている場合，プログラム構造を見やすくするために，ループ・ブロックおよびリピート・ブロックが自動的にインデントされます。インデント幅を文字数で指定してください。**1 **から** 25 **までの整数値を指定できます。**

- Expand reserved opcodes

  $\mu$PD77016 **命令は，多くのフィールドに分かれる** 32 **ビットの命令語から構成されています。フィールドには，予約ビットの組み合わせが含まれている場合があります。このボックスにチェック・マークが付いていない場合，予約オペコードを含むインストラクション・メモリ・アドレスは“** reserved **”で示されます。チェック・マークが付いている場合，逆アセンブラは非予約フィールドのニモニックを展開して表示されます。**

- OK

  **ダイアログ・ボックスを閉じ，新しいオプションを有効にします。該当するウインドウが新しいオプションで更新されます。**

- Cancel

  **操作を取り消してダイアログ・ボックスを閉じます。以前のオプションがそのまま有効になります。**

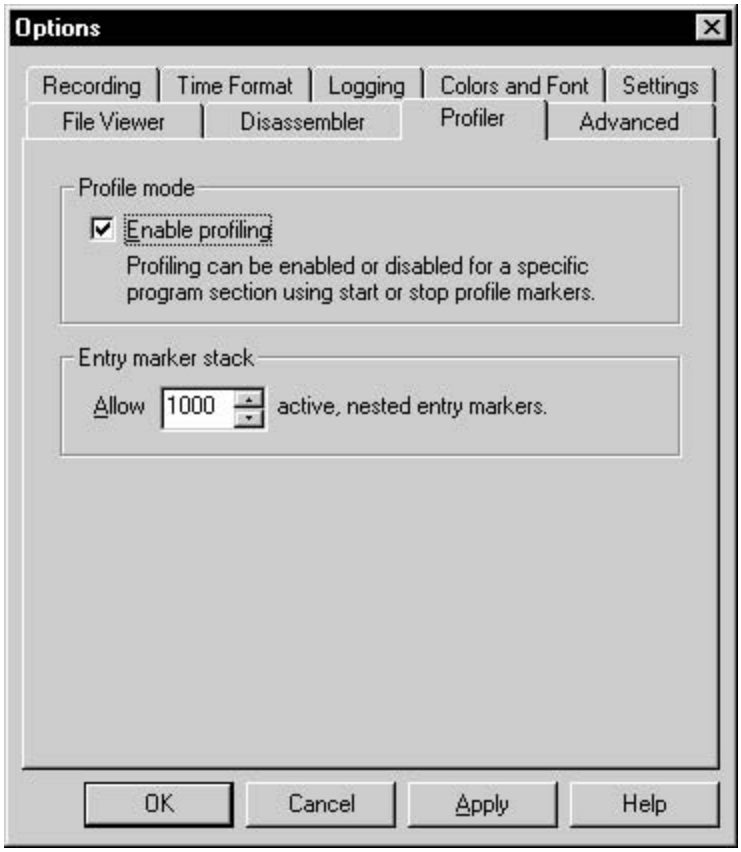
**（**5**）**Profiler Tab

Profiler **タブは，プロファイリングを有効にし，アクティブ・マーカ・スタック・サイズを設定するために使用します。**

**図** 6－27 **は** Profiler **タブです。**

保守／廃止

**図** 6－27　Profiler **タブ（**Options **ダイアログ・ボックス）**




**（**6**）**Profiler **タブの構成要素**

- Enable profiling

  **このコマンドにチェック・マークが付いている場合，プロファイリングが有効になります。ただし，実行が** Stop **マーカに達すると，このコマンドは無効になります。**

- Allow ... active, nested entry markers.

  **アクティブ・マーカ・スタック・サイズを入力してください。**10 **から** 30000 **までの整数値を指定できます。**

- OK

  **設定を確認してダイアログ・ボックスを閉じます。**

- Cancel

  **設定を更新しないでダイアログ・ボックスを閉じます。**

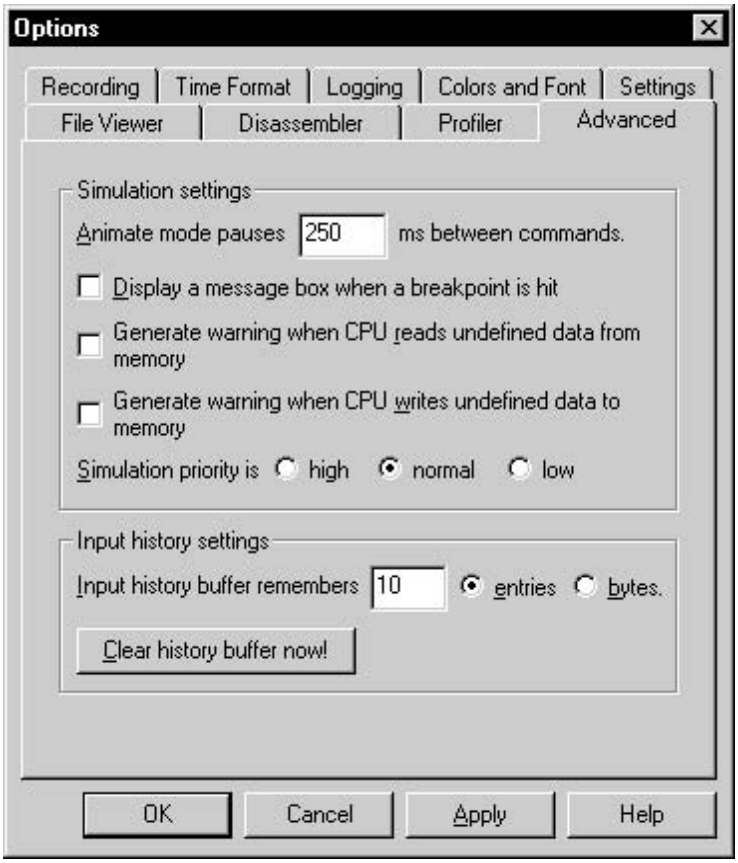
保守／廃止

（7）Advanced タブ

Advanced タブは，シミュレーションの設定とエディット・バーの履歴を調整するために使用します。

図 6－28 は Advanced タブです。

図 6－28　Advanced タブ（Options ダイアログ・ボックス）




（8）Advanced タブの構成要素

- Animate mode pause ... ms between commands.

  シミュレーション・ステップ間の遅延時間を 1/1000 秒単位で指定します。時間基準はホスト・コンピュータのシステム・クロックです。有効な入力値は 0（最小遅延）から 30000（30 秒の遅延）までの整数です。

- Display a message box when a breakpoint is hit

  このボックスにチェック・マークが付いている場合，ブレークポイントへの到達を通知するメッセージが表示されます。

- Generate warning when CPU reads undefined data from memory

  このボックスにチェック・マークが付いている場合，シミュレーション実行中にメモリから不定データが読み出されたことを通知するメッセージが表示されます。

- Generate warning when CPU writes undefined data to memory

  このボックスにチェック・マークが付いている場合，シミュレーション実行中にメモリに不定データが書き込まれたことを通知するメッセージが表示されます。

- Simulation priority is high/ normal/ low

  **これらのオプション・ボタンは，**Windows **がシミュレーション・アプリケーションの実行に割り当てる優先順位を決めるために使用します。**

- Input history buffer remembers

  **ヒストリ・リストの最新の指定入力項目数を保持するための値を指定し，次のシミュレーション・セッションで使用するためにこれらの項目を保存します。有効な値は** 1 **から** 10000 **までの整数です。**

- entries

  **このオプションを選択すると，バッファ・サイズ値が項目数として適用されます。たとえば，バッファ・サイズとして** 10 **を指定すると，最後の** 10 **個の項目がヒストリ・リストに保存されます。**

- bytes

  **このオプションを選択すると，バッファ・サイズ値がバイト数として適用されます。たとえば，バッファ・サイズとして** 100 **を指定すると，最大** 100 **バイトの項目が入力バッファに保存されます。**

- Clear history buffer now!

  **このボタンをクリックすると，ヒストリ・リストの入力項目が削除されます。**

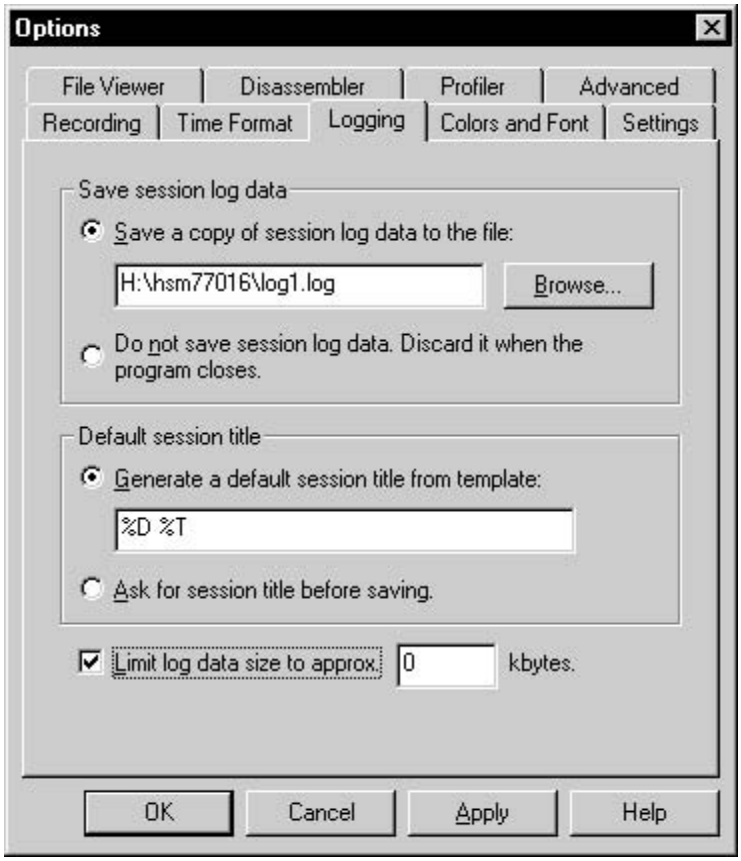
保守／廃止

(9) Logging タブ

Logging タブを使用して，ログ・ファイル名を選択し，ディフォールトのログ・セッション・タイトルを指定できます。

図 6－29 は Logging タブです。

図 6－29　Logging タブ（Options ダイアログ・ボックス）




(10) Logging タブの構成要素

- Save a copy of log session data to the file.

  このオプションを選択すると，編集フィールドにログ・ファイル名を入力できます。このファイルに，別のファイルを指定するまでログ・セッションがすべて格納されます。

- Browse...

  このボタンを選択すると，ファイル選択ダイアログ・ボックスが表示されます。このダイアログ・ボックスでログ・ファイルの名前とパスを選択します。このボタンが有効なのは，“ Save a copy of log session data to the file. ” にチェック・マークが付いている場合だけです。

- Do not save session log data. Discard it when the program closes.

  このオプションを選択すると，ログ・セッション・データがファイルに保存されません。アプリケーション終了時にログ・セッション・データが破棄されます。

保守／廃止

- Generate default session title from template.
  **ログ・セッション名を指定する場合や，テンプレートからディフォールトのセッション・タイトルを作成する場合は，このオプションを選択します。有効なテンプレート項目は次のとおりです。**

  %D **：長い日付け形式（日と月が書き出されます）。**
  %d **：短い日付け形式（日と月が数値で示されます）。**
  %T **：長い時間形式（秒も表示されます）。**
  %t **：短い時間形式（秒は表示されません）。**
  %% **：タイトル・バーに%文字を指定します。**

- Ask for session title before saving.
  **このオプションを選択すると，**Rename Log Session **ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，アプリケーション終了時にログ・セッション名を指定します。**

- Limit log data size to approx. ... kbytes.
  **このオプションで，ログ・データのサイズの上限を設定します。**

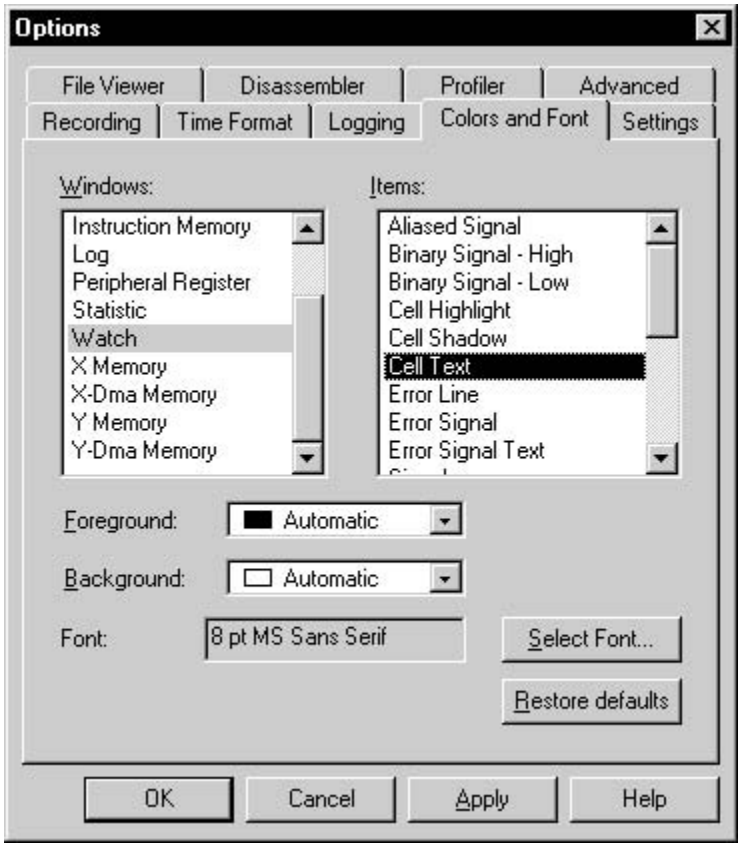
保守／廃止

（11）Colors and Font Tab

Colors and Font タブで，HSM77016 のデータ・ウインドウで使用するウインドウ要素の色とフォントを指定できます。

図 6－30 は Colors and Font タブです。

図 6－30　Colors and Font タブ（Options ダイアログ・ボックス）




（12）Colors and Font タブの構成要素

- Windows

  すべてのウインドウのリストが表示されます。このリストから複数のウインドウを選択できます。

- Items

  変更できるすべての色項目が表示されます（データ・ウインドウの処理別に分けられています）。このリストから複数の色項目を選択できます。

  特定の項目の色設定を変更するには，その項目の行をクリックします。

- Foreground

  このドロップダウン・リストには，文字やセル枠に適用される色設定が示されます。

- Background

  このドロップダウン・リストには，文字の背景やセルの影に適用される色設定が示されます。

- Font
  Font ダイアログ・ボックスで選択されたフォントが表示されます。

- Select Font...
  データ・ウインドウのフォント・プロパティを設定するための Font ダイアログ・ボックスが表示されます。選択したデータ・ウインドウごとにフォントの種類とサイズを指定できます。
  Sample ボックスの文字列は，選択した字体と属性で表示されます。ユーザ・システム上で使用できる任意の ANSI フォント・ファイル名を選択できます。実際に使用されるフォントは Windows に依存するので，ユーザが選択したものとは異なる場合があります。ANSI フォントでないプリンタ・フォントや記号フォントを選択することはできません。
  Size リスト・ボックスに表示されるフォント・サイズは Windows のものであり，すべてを選択できるとはかぎらないので注意してください。ただし，このリスト・ボックスにないサイズを入力ボックスに直接入力することはできます。
  次のエミュレーション用にフォント設定を保存する場合は，Settings タブの Generic Settings オプションにチェック・マークを付けてください。

- Restore default
  このボタンを押すと，ディフォールトの色とフォントの設定が復元されます。

**注意**　Tools メニューの Options...コマンドで Generic Settings オプションを表示し，このオプションにチェック・マークを付けると，データ・ウインドウの色設定が保存されます。

**（13）色項目**

次の色項目を設定できます。

- 3D-Highlight：3 次元効果を出す小さい縁取り
- 3D-Shadow：3 次元効果を出す小さい縁取り
- Binary Signal – High：現在ハイ・レベルの 2 進シグナルの色
- Binary Signal – Low：現在ロウ・レベルの 2 進シグナルの色
- Breakpoint：ブレークポイント・シンボルの色
- Breakpoint Cell：有効な（アクティブな）ブレークポイントを含むセルの色
- Cell Text：データ出力領域
- Changed Cell Text：HSM77016 のコマンドによって変更されたデータ出力領域
- Error Line：データ・ウインドウ・エラー・メッセージを表示する行
- Error Signal：エラー・シグナル行の色
- Error Signal Text：エラー・シグナル文字の色，フォント，背景色
- Error Static Text：エラー条件を示すセル内の静的文字の色
- Graphic：Statistic ウインドウのグラフィック・バー
- Hatched Signal Block：Watch ウインドウの網かけ領域の色
- High Impedance Signal：現在ハイ・インピーダンスの 2 進シグナルの色
- Hit Line：到達したブレークポイントを含む行
- Hit Static Text：到達したブレークポイントを含む行の静的テキストの色
- Inactive Breakpoint Cell：無効なブレークポイントを含むセル

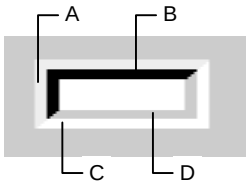
- Signal：Watch ウインドウのシグナル行の色
- Signal Pane Background：Watch ウインドウの Signal ブロックの背景色
- Signal Text：Watch ウインドウのシグナル・テキストの色，フォント，背景色
- Source Lines：ソース・コードを含む Instruction Memory ウインドウの行
- Static Text：データ・ウインドウのセルで使用されている静的テキスト
- Timecursor：Watch ウインドウのタイム・カーソルの色
- Timecursor Tooltip Text：Watch ウインドウのタイム・カーソル・ツール・チップの色
- Timescale Decade：Watch ウインドウのタイム・スケールの 10 色
- Timescale Division：Watch ウインドウの 10 ごとのタイム・スケール区分の色
- Undefined Signal：Watch ウインドウの未定義シグナルの色
- Window Text：データ・ウインドウで使用されている静的テキスト

### （14）色の指定

色を指定するには，Item リストから変更対象の色項目を選択し，Foreground および Background ドロップダウン・リストを使用して文字色や背景色を変更します。新しい色は，該当するデータ・ウインドウにすぐに適用されます。OK ボタンをクリックすると，変更が確定します。

3 次元効果に使用される色は，図 6－31 に示した配色になっています。

**図 6－31　3 次元効果**




A ：3D Shadow / Foreground Color 属性

B ：3D Shadow / Background Color 属性

C ：3D Highlight / Foreground Color 属性

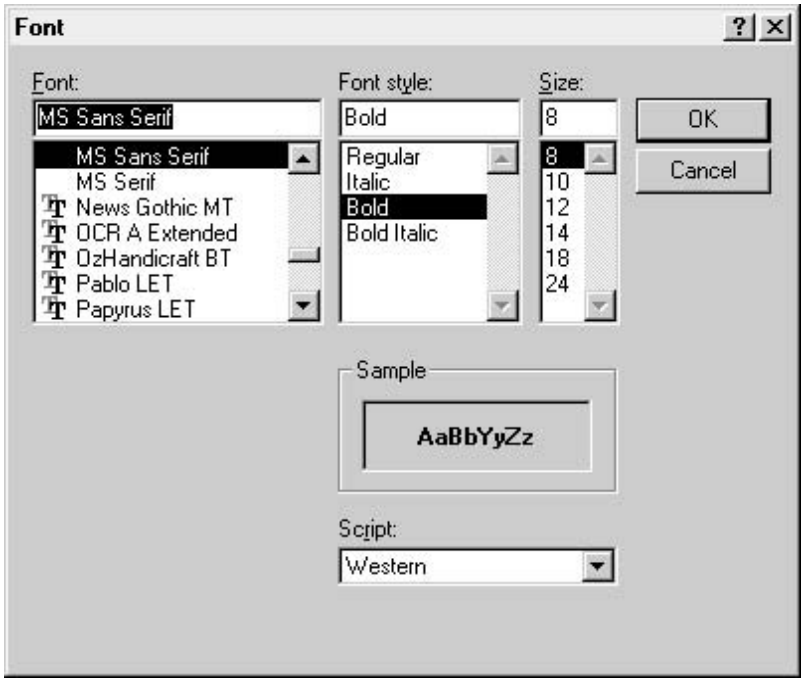
D ：3D Highlight / Background Color 属性

### （15）Font ダイアログ・ボックス

フォントの種類とサイズは，すべて HSM77016 データ・ウインドウで指定できます。Sample ボックスの文字は，選択した字体と属性で表示されます。ユーザ・システム上で使用できる ANSI フォント・タイプ名を選択できます。実際に使用されるフォントは Windows に依存するので，ユーザが選択したものとは異なる場合があります。ANSI フォントでないプリンタ・フォントや記号フォントを選択することはできません。図 6－32 は Font ダイアログ・ボックスです。

保守／廃止

図 6－32　Font ダイアログ・ボックス




**注意**　Size リスト・ボックスに表示されるフォント・サイズは，Windows のものであり，すべてを選択できるとはかぎらないので注意してください。ただし，このリスト・ボックスにないサイズを入力ボックスに直接入力することはできます。

次のエミュレーション用にフォント設定を保存する場合は，Settings タブの Generic Settings オプションにチェック・マークを付けてください。詳細については，（16）Settings タブを参照してください。

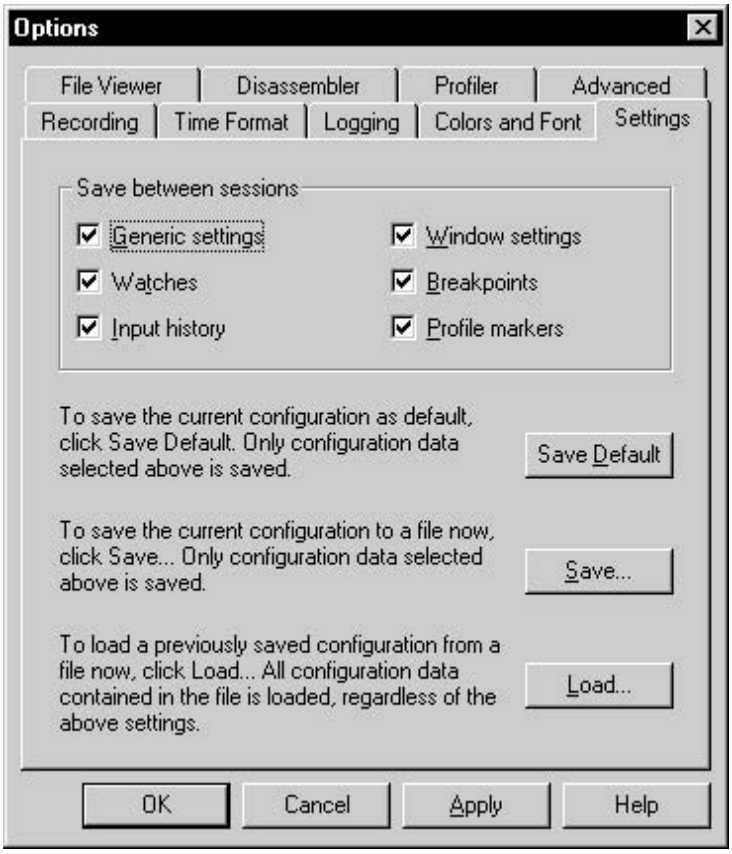
保守／廃止

（16）Settings タブ

Settings タブを使用して，指定した内容を設定ファイルに保存したり，設定ファイルから設定を再ロードできます。

図 6－33 は Settings タブです。

図 6－33　Settings タブ（Options ダイアログ・ボックス）




（17）Settings タブの構成要素

- Generic Settings

  このボックスにチェック・マークが付いている場合，次の設定が保存されます。

  - HSM77016 のデータ・ウインドウ用の色設定
  - 逆アセンブラの設定
  - HSM77016 のすべてのデータ・ウインドウに共通の設定
  - Settings タブの設定

- Watches

  このボックスにチェック・マークが付いている場合，Watch ウインドウに含まれている変数と式が保存されます。

- Input history

  入力ヒストリ設定を保存するには，このボックスにチェック・マークを付けます。

- Window settings

  **このボックスにチェック・マークが付いている場合，それぞれのデータ・ウインドウの設定が保存されます。たとえば，データ・ウインドウの表示位置とサイズ，行の数とサイズ，選択した表示形式などが保存されます。**

- Breakpoints

  **このボックスにチェック・マークが付いている場合，**Breakpoint **ウインドウに含まれているブレークポイントとその属性がすべて保存されます。**

- Profile markers

  **このボックスにチェック・マークが付いている場合，**Statistic **ウインドウに含まれているプロファイリング・マーカとその属性がすべて保存されます。**

- Save Default

  **このボタンをクリックすると，現在の設定（**Save between sessions **コントロール群の現在の設定）がディフォールト設定として保存されます。アプリケーション終了時にディフォールト設定が上書きされないように，現在の設定を保存したあと，** Save between sessions **の設定が自動的に削除されます。**

- Save...

  **このボタンを選択すると，ファイル選択ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，現在の設定を格納する設定ファイルを指定します。**

- Load...

  **このボタンを選択すると，ファイル選択ダイアログ・ボックスが表示されます。このダイアログ・ボックスで，ロードする設定ファイルを選択します。**

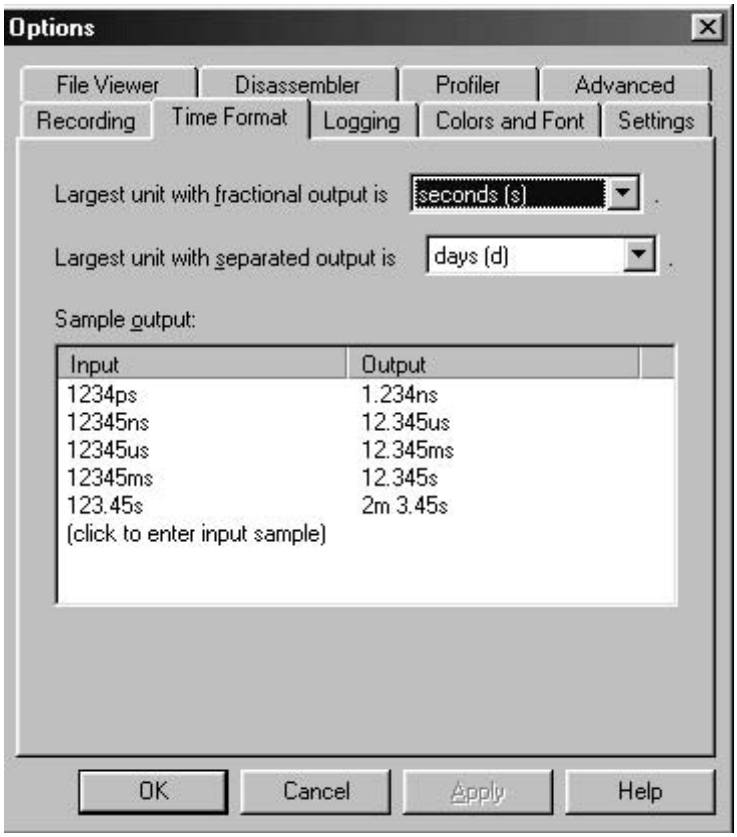
保守／廃止

（18）Time Format タブ

Options ダイアログ・ボックスの Time Format タブで，HSM77016 の時間出力の形式を選択できます。指定できる単位は日，分，秒，ミリ秒，マイクロ秒，ナノ秒，ピコ秒です。

図 6－34 は Time Format タブです。

図 6－34　Time Format タブ（Options ダイアログ・ボックス）




（19）Time Format タブの構成要素

- Largest unit with fractional output is ... .

  少数出力で使用される最大の時間単位を指定します。

- Largest unit with separated output is ... .

  時間出力を表示する最大の時間単位を指定します。

- Sample output

  上記のフィールドにこれまで入力された設定値が表示されます。このリスト内の時間項目を選択して F2 キーを押すと，その項目を編集できます。Delete キーを押すと，その項目を削除できます。

  Time Format タブで行った設定は保存されません。したがって，HSM77016 を閉じて再び開くとその設定は失われます。

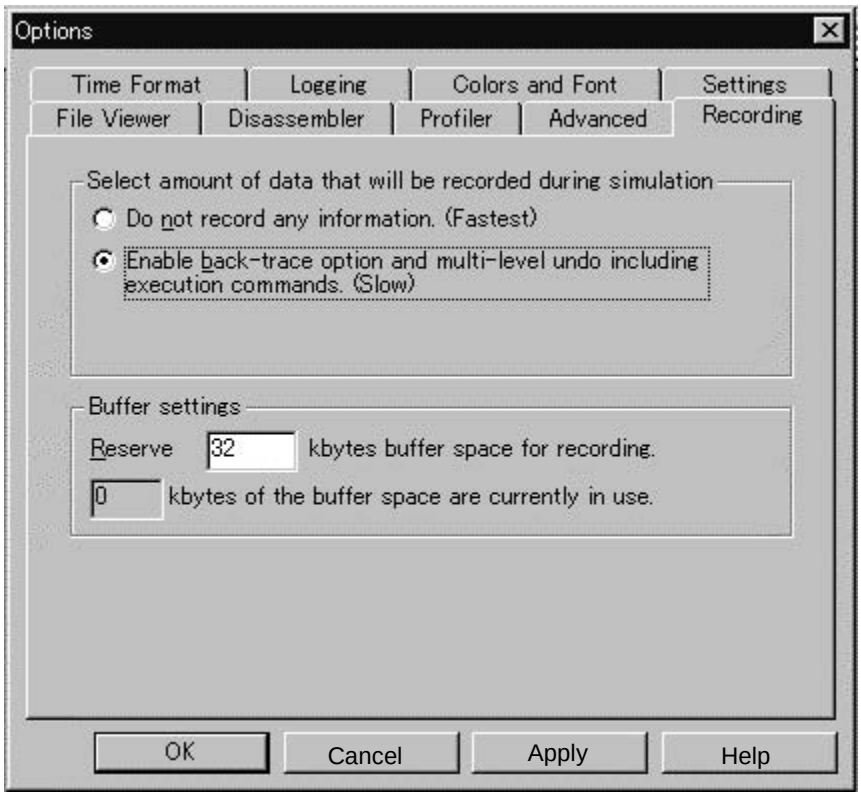
保守／廃止

**（**20**）**Recording **タブ**

Options **ダイアログ・ボックスの** Recording **タブで，**HSM77016 **のシミュレーション履歴（**undo**，バックトレース動作など）の記録をファイルに残したり，シミュレーション履歴ファイルのサイズを設定します。**

**シミュレーション履歴を記録すると，シミュレーション速度が遅くなります。ユーザは，シミュレーション速度とシミュレーション履歴のどちらを優先するか選択することができます。**

**図** 6－35　Recording **タブ（**Options **ダイアログ・ボックス）**




**（**21**）**Recording **タブの構成要素**

- Do not record any information.
  **シミュレーション履歴を記録しません。シミュレーションは最高速度で動作します。**

- Enable back-trace and multi-level undo including execution commands.
  **バックトレースおよび** undo **情報を記録します。シミュレーション履歴のバッファ・サイズは** Buffer Setting **タブ内で設定できます。このオプションを選択すると，シミュレーション速度が遅くなりますので，注意してください。**

- Reserve ... kbytes buffer space for recording.
  **レコード・ファイルのサイズを設定します。レコード・バッファがいっぱいになると，古いデータから破棄されます。**

- ... kbytes of the buffer space are currently in use.
  **現在使用中のバッファ・サイズを示します。**

保守／廃止

## 6.14　Window メニュー

Window メニューには，標準の MDI 機能（Tile，Cascade，Arrange Icons，Close All）および，HSM77016 のデータ・ウインドウまたはシミュレーション・プラグインでアクセスできるデータ・ウインドウを表示したり手前にもってくるためのコマンドが含まれています。ウインドウの表示コマンドをすでに開いているウインドウに適用した場合，最初に開いたウインドウが手前に表示され，そのコマンドにチェック・マーク（√）が付きます。

標準の MDI 機能の詳細については，Windows のマニュアルを参照してください。

### 6.14.1　Tile

開いている各ウインドウが重ならないように，HSM77016 のメイン・ウインドウを表示します。

### 6.14.2　Cascade

各ウインドウのタイトル・バーが見えるようにウインドウを重ねて表示します。

### 6.14.3　Duplicate

アクティブなデータ・ウインドウと同じ種類の別のウインドウを表示します。ウインドウのタイトル・バーに示される番号で，各ウインドウを区別できます。ただし，Log，Watch，Breakpoint の各ウインドウには無効です。

### 6.14.4　Arrange Icons

すべてのウインドウのアイコンを HSM77016 のメイン・ウインドウの下部に並べて表示します。

### 6.14.5　Close All

開いているすべてのウインドウを閉じます。ロードされているファイル（テキスト・ウインドウに表示されているファイル以外）は，該当するデータ・ウインドウを閉じても HSM77016 から削除されません。ブレークポイントとウォッチ式はそのまま残ります。シミュレータ・セッション間に，ブレークポイントとウォッチ式を Tools メニューの Options...コマンドで保存することができます。

### 6.14.6　CPU Register

このコマンドは，CPU Register ウインドウを表示します。このウインドウには，現在のシミュレーション・モデルのすべてのレジスタ情報が表示されます。CPU Register ウインドウがすでに開いている場合は，手前に表示されます。開いていない場合は，新しいウインドウが表示されます。別の CPU Register ウインドウを開くには，Windows メニューの Duplicate コマンドを使用してください。

CPU Register ウインドウの詳細については，5.2.1　CPU Register ウインドウを参照してください。

### 6.14.7　Peripheral Register

このコマンドは，Peripheral Register ウインドウを表示します。このウインドウには，シミュレーション・モデル・プロセッサのポート・レジスタ，ポート・フラグ，およびウエイト・サイクル・レジスタが表示されます。Peripheral Register ウインドウがすでに開いている場合は，手前に表示されます。開いていない場合は，新しいウインドウが表示されます。。別の Peripheral Memory ウインドウを開くには，Windows メニューの Duplicate コマンドを使用してください。Peripheral Register ウインドウの詳細については，5.2.2　Peripheral Register ウインドウを参照してください。

保守／廃止

### 6.14.8　Instruction Memory

**このコマンドは，**Instruction Memory **ウインドウを表示します。このウインドウには，シミュレーション・モデル・プロセッサのインストラクション・メモリ空間が表示されます。**Instruction Memory **ウインドウがすでに開いている場合は，手前に表示されます。開いていない場合は，新しいウインドウが表示されます。別の** Instruction Memory **ウインドウを開くには，**Windows **メニューの** Duplicate **コマンドを使用してください。**

Instruction Memory **ウインドウの詳細については，**5.1　**メモリ・ウインドウを参照してください。**

### 6.14.9　X-Data Memory

**このコマンドは，**X-Data Memory **ウインドウを表示します。このウインドウには，シミュレーション・モデル・プロセッサの** X **データ・メモリ空間が表示されます。**X-Data Memory **ウインドウがすでに開いている場合は，手前に表示されます。開いていない場合は，新しいウインドウが表示されます。別の** X-Data Memory **ウインドウを開くには，**Windows **メニューの** Duplicate **コマンドを使用してください。**

X-Data Memory **ウインドウの詳細については，**5.1　**メモリ・ウインドウを参照してください。**

### 6.14.10　Y-Data Memory

**このコマンドは，**Y-Data Memory **ウインドウを表示します。このウインドウには，シミュレーション・モデル・プロセッサの** Y **データ・メモリ空間が表示されます。**Y-Data Memory **ウインドウがすでに開いている場合は，手前に表示されます。開いていない場合は，新しいウインドウが表示されます。別の** Y-Data Memory **ウインドウを開くには，**Windows **メニューの** Duplicate **コマンドを使用してください。**

Y-Data Memory **ウインドウの詳細については，**5.1　**メモリ・ウインドウを参照してください。**

### 6.14.11　X-DMA Memory

**このコマンドは，**X-DMA Memory **ウインドウを表示します。このウインドウには，**X-DMA **データ・メモリ空間が表示されます。これは，**μPD77116 DMA **メモリ・コントローラがアクセスできる外部メモリ空間としてのメモリ空間です。**

μPD77116 **に対するディバグ・セッションの場合だけ，**X-DMA Memory **ウインドウを表示することができます。**

X-DMA Memory **ウインドウの詳細については，**5.1　**メモリ・ウインドウを参照してください。**

### 6.14.12　Y-DMA Memory

**このコマンドは，**Y-DMA Memory **ウインドウを表示します。このウインドウには，**Y-DMA **データ・メモリ空間が表示されます。これは，**μPD77116 DMA **メモリ・コントローラがアクセスできる外部メモリ空間としてのメモリ空間です。**

μPD77116 **に対するディバグ・セッションの場合だけ，**Y-DMA Memory **ウインドウを表示することができます。**

Y-DMA Memory **ウインドウの詳細については，**5.1　**メモリ・ウインドウを参照してください。**

保守／廃止

### 6.14.13　Breakpoint

Breakpoint コマンドは，Breakpoint ウインドウを表示します。このウインドウで次の操作ができます。

- 定義されているブレークポイントの現在の状態やグループおよびパス・カウンタを調べます。
- ブレークポイントをシミュレーションに追加します。
- 定義されているブレークポイントの状態，グループ，またはパス・カウンタの属性の編集や変更を行います。
- ブレークポイント・グループを有効または無効にします。
- ブレークポイントの動作と条件を追加します。
- 定義されているブレークポイントをシミュレーションから削除します。

Breakpoint ウインドウの詳細については，5.3　Breakpoint ウインドウを参照してください。

### 6.14.14　Log

Log コマンドは，Log ウインドウを表示します。このウインドウにはログ・バッファの内容が表示されます。

Log ウインドウの詳細については，5.5　Log ウインドウを参照してください。

### 6.14.15　Watch

Watch コマンドは，Watch ウインドウを表示します。このウインドウで，ユーザが指定した変数と式を監視します。

Watch ウインドウの詳細については，5.4　Watch ウインドウを参照してください。

### 6.14.16　Statistic

Statistic コマンドは，Statistic ウインドウを表示します。このウインドウでは，次の操作ができます。

- プロファイリング・セッションで収集された統計データを調べます。
- プロファイリング・マーカを設定したり削除します。
- 統計データをリセットします。

### 6.14.17　開いているウインドウのリスト

このリストに次のウインドウが表示されます。

- テキスト・ウインドウ
- 複製されたデータ・ウインドウ

HSM77016 のウインドウは重なり合ったりほかのウインドウを隠すことがあるので，マウスのクリックでは目的のウインドウをアクティブにできない場合があります。そのため，Windows メニュー項目の後ろに，開いているウインドウのリストが番号順に表示されます。このリストからウインドウを選択すると，そのウインドウがアクティブになります。リストでは，アクティブ・ウインドウがチェック・マーク（√）で示されます。1 つのウインドウを閉じると，リストの番号が付け直されます。ウインドウを 10 個以上開いている場合，リストには More Windows...コマンドが表示されます。このコマンドを選択すると，リストにすでに表示されているものも含めて，開いているすべてのウインドウを示すダイアログ・ボックスが表示されます。

保守／廃止

# 6.15　Help メニュー

Help メニューの各コマンドで，オンライン・ヘルプ，μPD77016 チップに関するヘルプ情報，バージョン情報，および HSM77016 に登録されたプラグインのバージョン情報にアクセスできます。

## 6.15.1　Contents

Contents コマンドは，HSM77016 の主要なヘルプ項目すべてのリストを表示します。ヘルプ項目は次のように分かれています。

- User Interface：シミュレーション（データ・ウインドウ）処理および HSM77016 のユーザ・インタフェースの構成要素に関する項目
- Commands：HSM77016 のメニュー・コマンドの説明項目
- Files：HSM77016 のファイルの種類に関する情報
- Windows：HSM77016 のデータ・ウインドウ情報
- Reference：数値形式や演算子など，その他の情報
- Keyboard：キーボードのショートカットとインタフェースの情報
- Dialog Index：HSM77016 のダイアログ・ボックスのアルファベット順のリスト

## 6.15.2　Using Help

Using Help コマンドは，ヘルプの使い方を表示します。

## 6.15.3　On-Line Manual

このコマンドは，μPD77016 ファミリの機能についての情報を表示します。主なヘルプ項目は次のとおりです。

- 内部ブロック
- レジスタ・セット
- アドレシング・モード
- データ形式
- 命令セット
- 割り込み
- 内蔵周辺回路
- メモリ・インタフェース
- AC/DC ターゲット仕様

## 6.15.4　Instruction Set

このコマンドは，μPD77016 命令セットについての詳しい説明を命令の機能グループごとに表示します。

## 6.15.5　READ-ME Information

このコマンドは，現在のバージョンに関する最新の情報を含むアプリケーション readme ヘルプ・ファイルを表示します。

保守／廃止

### 6.15.6　About...

このダイアログ・ボックスに，HSM77016 に関する情報が表示されます。横線の上には，アプリケーションの名前とアイコン，プログラム・バージョン番号，著作権表示，およびプログラム作成者の名前が表示されます。横線の下には，ユーザ・システムと現在のシステム資源に関する情報が表示されます。

図 6－36 は About...ダイアログ・ボックスです。

図 6－36　About...ダイアログ・ボックス

保守／廃止

★

# 第7章　モデル

## 7.1 概　　要

μPD77016 ファミリ用のプログラムはモデルに基づいて開発されています。モデルは，ターゲット環境に関する基本情報を格納するために特別な構文を使用するファイルです。このようなファイルの拡張子は “ .model ” です。ターゲット環境情報としては，プロセッサ・タイプ，プロセッサの属性，外部メモリ領域などがあります。

HSM77016 パッケージに同梱の Model Editor を使用すれば，モデル定義ファイルを編集しなくても，新しいモデルを作成したり既存のモデルを変更できます。

Model Editor は，基本的に次の 2 つの部分に分かれています。

- Model File ウィザード

  このウィザードを利用して，HSM77016 の内部でモデル・ファイルを作成，複製，編集できます。Target System Model ダイアログ・ボックスの Model Wizard ボタンで Model File ウィザードを表示できます。

- シェル拡張機能

  この機能を利用して，登録されているその他の種類のファイルと同様にモデル・ファイルの作成，移動，コピー，編集，モデル・ファイルの名前の変更やショートカットの作成ができます。これらの処理は，エクスプローラの内部で行うことも，デスクトップ上で直接行うこともできます。

## 7.2 モデルと debuggee

モデルは，ターゲット環境に関する基本情報を提供します。ターゲット環境情報には次のものがあります。

- 固有モデル名
- プロセッサ・タイプ
- クロック速度
- 外部 X データ・メモリおよび外部 Y データ・メモリのメモリ領域（オプション）
- 外部インストラクション・メモリのメモリ領域（オプション）
- モデルがロードされるか処理がリセットされるたびに，HSM77016 または ID77016 で実行される式（オプション）
- モデルの説明（オプション）

ID77016 には，プログラムを実行する環境とチップに関するさらに詳しい情報が必要です。このような補足情報は “ debuggee ” と呼ばれます。補足情報には次のものがあります。

debuggee は既存のモデルに基づいているので，そのモデル・ファイルに常駐します。

保守／廃止

- 固有 debuggee 名
- debuggee の元になるモデル
- リンク名

  リンクとはターゲット・ハードウエアと通信する手段です。利用できるすべてのリンクの名前が付いたリストを，ターゲット・ハードウエアのドライバで提供する必要があります。
- 補足 2 進情報

  たとえば，ターゲット・システムでホストを起動できるかどうかなどの情報です。
- debuggee がリセットされるたびに実行される初期化（オプション）

### 7.2.1　モデル定義ファイル

Atair DSP ツールの以前のバージョンでは，モデルの定義に “ モデル定義ファイル（UPD77016.INI ファイル）” が利用されていました。この種のファイルには，すべてのモデル，チップの説明，debuggee が含まれます。

モデルを作成，編集するには，モデル定義ファイルの正確な構文規則を学んで，このファイルを編集する必要がありました（モデル定義ファイルとその構文規則の詳細については，7.3　**モデル・ウィザード**を参照してください）。

Model Editor はモデル定義ファイルをロードでき，またモデル定義ファイルに含まれるすべてのモデルを編集できます。しかし，モデル定義ファイルの処理には 1 つ大きな制約があり，それは Model Editor で作成する新しいモデルは，“ .model ” という種類になり，それぞれ独自のファイルに格納されるということです（このような新しいモデルの詳細については，7.2.2　**モデル・ファイル**を参照してください）。モデル定義ファイルに格納される新しいモデルを作成することはできません。

### 7.2.2　モデル・ファイル

モデル・ファイルには，1 つのモデル，そのモデルの対象となるチップ，そのモデルに基づく debuggee（これはオプションです）が含まれます。

モデル・ファイルの内部構成はモデル定義ファイルと基本的に同じですが，1 つ大きな違いがあります。モデル・ファイル内において，モデルの名前はモデル・ファイル内の特定の項目を示すのではなく，ファイル名そのものであるということです。これによって，モデル処理が簡単になります。登録されているその他の種類のファイルと同様に，エクスプローラ内またはデスクトップ上で直接，モデルを作成，コピー，移動，名前の変更，編集できます。この機能を利用する詳しい方法については，7.4　**シェル拡張機能**を参照してください。

### 7.2.3　チップ情報ファイル

チップ情報ファイルには，サポートされているすべてのチップに関する情報が含まれています。このファイルは最初，HSM77016 がインストールされているディレクトリにあります。

さらに，各モデル・ファイルには，モデルの対象となるチップに関する情報が含まれています。したがって，最初から作成したモデルは，チップ情報ファイルで説明されているチップの中の 1 つを対象とするだけです。複製または編集されたモデルでは，チップ情報ファイルまたはモデル・ファイルで説明されているチップを選択できます。

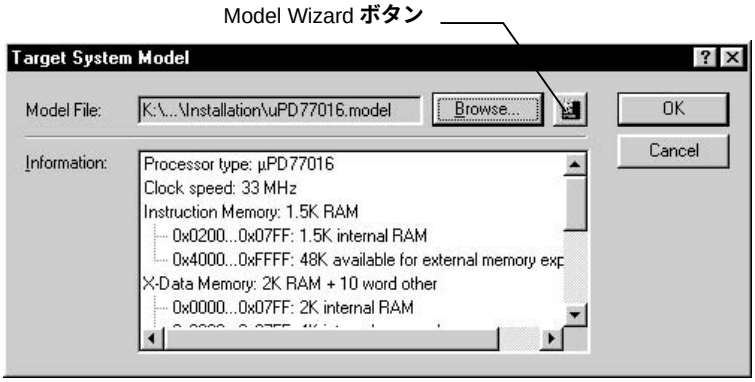
複製または編集されるモデル・ファイルとチップ情報ファイルの両方で記述されているチップのどちらかを選択する場合は，チップに関する記述が有効であるかぎり，モデル・ファイルのチップが優先されます。両方のチップ記述が有効で，記述に違いがある場合は，どちらかを選択するようメッセージが表示されます。

保守／廃止

# 7.3　Model ウィザード

Model ウィザードは，HSM77016 の Target System Model ダイアログ・ボックスから起動します。図 7－1 は，Target System Model ダイアログ・ボックスのウィザード起動用ボタンを示しています。

図 7－1　Model Wizard ボタン




## 7.3.1　Model Wizard ボタンのコマンド

- New

  このコマンドは，Model ウィザードを起動し，チップ情報ファイルに含まれる DSP を対象とする新しいモデルを作成します。ユーザは，モデルのすべてのプロパティを指定しなければなりません。Model ウィザードが終了するとファイル選択ダイアログ・ボックスが表示されるので，モデルの名前と格納場所を指定してください。

- Clone...

  このコマンドは，Model ウィザードを起動し，現在アクティブなモデルに基づいて新しいモデルを作成します。ユーザが属性を変更しないかぎり，複製モデルは元のモデルの属性をすべて引き継ぎます。Model ウィザードが終了するとファイル選択ダイアログ・ボックスが表示されるので，モデルの名前と格納場所を指定してください。

- Edit...

  このコマンドは，Model ウィザードを起動し，現在アクティブなモデルを変更します。新しいモデルは作成されません。現在のモデルが，変更されたモデルに置き換えられます。モデルの複製と同様に，どの属性でも変更や削除ができ（モデル名は例外），新しい属性を追加することもできます。

- Rename...

  このコマンドは，アクティブなモデル定義ファイル内の現在選択されているモデルの名前を変更するためのダイアログ・ボックスを表示します。このコマンドを使用できるのは，ディバグ・セッションが以前のモデル定義ファイルに基づいて行われている場合だけです。

• Delete

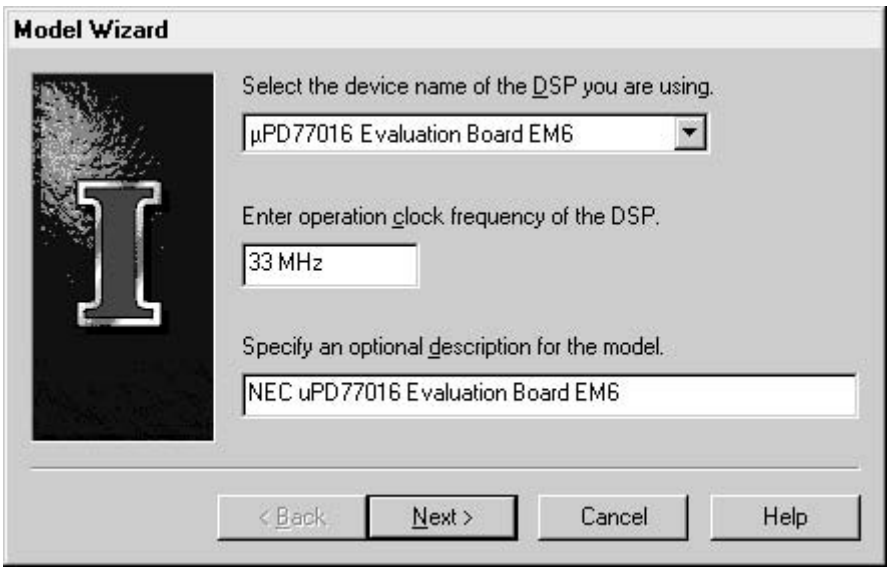
**このコマンドは，アクティブなモデル定義ファイル内の現在選択されているモデルを削除します。このコマンドを使用できるのは，ディバグ・セッションが以前のモデル定義ファイルに基づいて行われている場合だけです。削除されたモデルは復元できません。**

### 7.3.2　Model ウィザード・ページ I（DSP 選択）

**このページでは，モデルの基本属性として，モデルの対象となるプロセッサ・タイプおよびそのプロセッサのクロック周波数を指定します。**

**図** 7－2 **は** DSP **選択ページです。**

**図** 7－2　DSP **選択ページ**




**（**1**）**DSP **選択ページの構成要素**

• Select the device name of the DSP you are using.

$\mu$PD77016 **ファミリ・プロセッサのリストからターゲット** DSP **を選択します。モデルを複製または編集する場合は，現在アクティブなモデルのターゲット・プロセッサがこのフィールドに表示されます。新しいモデルを作成する場合このフィールドには何も表示されませんが，新しいモデルのターゲット** DSP **として，リストから任意のプロセッサを選択できます。プロセッサ・タイプ・リストには，現在アクティブなモデル定義ファイルで指定されているプロセッサ・タイプだけが表示されます。**

• Enter operation clock frequency of the DSP.

**指定したターゲット** DSP **のクロック周波数を入力します。入力できる周波数単位は，**Hz，KHz，MHz，GHz **です。**

• Specify an optional description for the model.

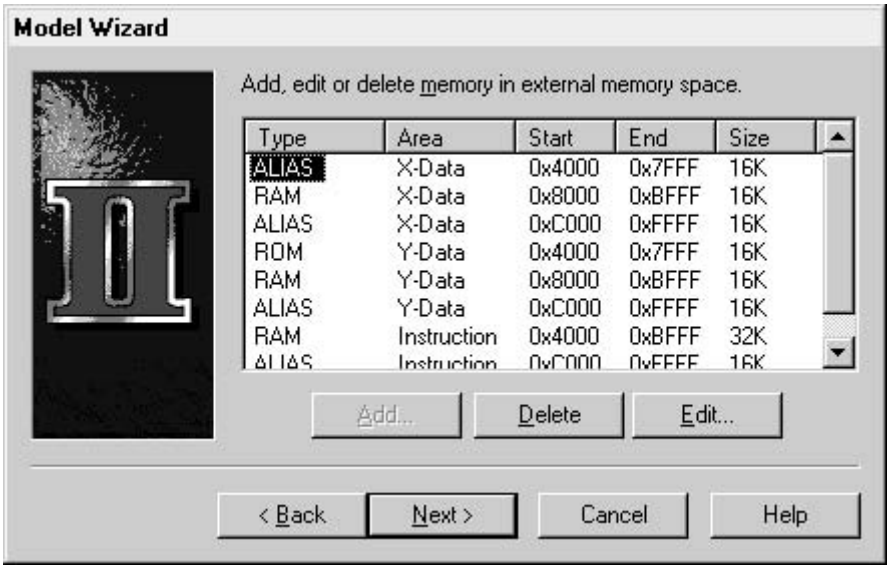
**このフィールドには，モデルに関するメモや補足説明を入力します。**

保守／廃止

### 7.3.3　Model ウィザード・ページ II（外部メモリ仕様）

このページには，インストラクション，X データ，および Y データの外部メモリ領域と，現在アクティブなモデルで定義されている空きメモリ領域が表示されます。モデルのメモリ領域は，追加，削除，または編集できます。プロセッサ・タイプを変更した場合，既存のメモリ領域が自動的に採用されます。問題が発生した場合，メモリ領域の競合が解消されるまでウィザードが停止します。図 7−3 は外部メモリ仕様ページです。

**図 7−3　外部メモリ仕様ページ**




#### (1) 外部メモリ仕様ページの構成要素

- メモリ・ブロック・リスト

  このリストには，メモリ・タイプ，メモリ空間，開始アドレスと終了アドレス，メモリ・サイズなどのメモリ・ブロック情報が表示されます。

- Add...

  このボタンをクリックすると，メモリ・ブロックが追加され，ブロック・アドレスなどのメモリ・ブロック属性，タイプ属性，およびメモリ・アクセス・ウエイト・サイクルを持つメモリ・ブロックを追加します。このボタンは，少なくとも 1 つの空きメモリ・ブロックが存在する場合のみ有効です。

- Delete

  メモリ・ブロック・リストから現在選択されているメモリ・ブロックを削除します。空きメモリ・ブロックが選択されている場合，このボタンは無効です。

- Edit

  選択されたメモリ・ブロックを編集し，メモリ・ブロック属性を変更します。このボタンは，1 つのブロックが選択されている場合のみ有効です。

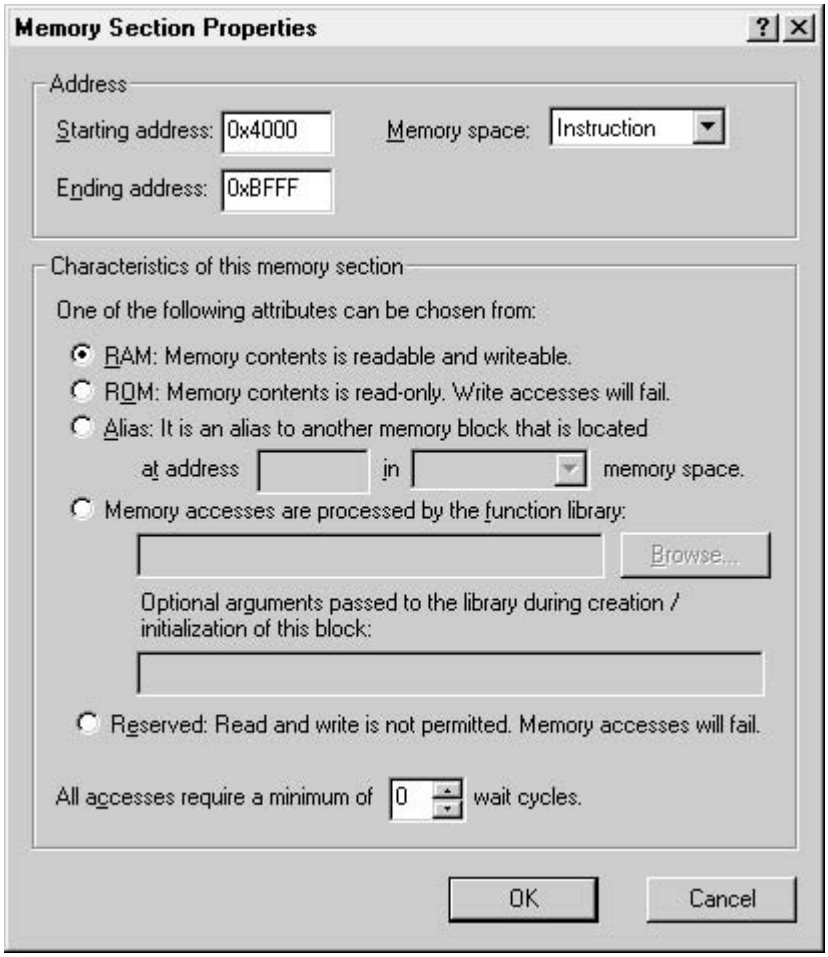
保守／廃止

## 7.3.4　Memory Section Properties ダイアログ・ボックス

このダイアログ・ボックスで，選択されている外部インストラクション・メモリ・ブロックまたはデータ・メモリ・ブロックの属性を編集または変更できます。

図 7－4 は Memory Section Properties ダイアログ・ボックスです。

図 7－4　Memory Section Properties ダイアログ・ボックス




### (1) Memory Section Properties ダイアログ・ボックスの構成要素

- Starting address

  このフィールドにメモリ・ブロックの開始アドレスを入力します。

- Ending address

  このフィールドにメモリ・ブロックの終了アドレスを入力します。

- Memory space

  ドロップダウン・リストからメモリ空間を選択します。このリストには，現在選択されているプロセッサ・タイプに適したメモリ空間が表示されます。

- RAM

  この属性を選択すると，メモリ・ブロックが RAM 内に作成されます。

- ROM

  **この属性を選択すると，メモリ・ブロックが** ROM **内に作成されます。**

- Alias

  **この属性を選択すると，指定のアドレスに位置するメモリ・ブロックに対するエイリアスとしてメモリ・ブロックが作成されます。**start address **にはエイリアスになるメモリ・ブロックの開始アドレスを入力し，**memory space **ではエイリアス・メモリ・ブロックを配置するメモリ空間を選択します。**memory space **リストに表示されるメモリ空間は，編集対象のメモリ・ブロックに対して選択されたメモリ空間によって変わります。**

- Memory accesses are processed by the function library

  **このオプションを選択すると，指定のメモリ・ブロックが外部ライブラリによって処理されます。指定のメモリ・ブロックへのメモリ・アクセスを処理する外部ライブラリの名前とパスを入力するか，**Browse... **ボタンをクリックして外部ライブラリを表示します。指定した外部ライブラリのロード時にライブラリに渡すコマンド行引数を指定します。**

- Reserved

  **この属性を選択すると，指定のメモリ・ブロックが予約メモリ・ブロックになります。予約メモリ・ブロックにアクセスすると，エラー・メッセージが表示されます。**

- All accesses require a minimum of ... wait cycles.

  **指定のメモリ・ブロックに必要なウエイト・サイクルの数を入力します。ボックスの右側のボタンをクリックすると，**0 **から** 255 **までのウエイト・サイクル値を表示できます。**

- OK

  **変更内容を有効にしてダイアログ・ボックスを閉じます。**

- Cancel

  **変更内容を無効にしてダイアログ・ボックスを閉じます。**

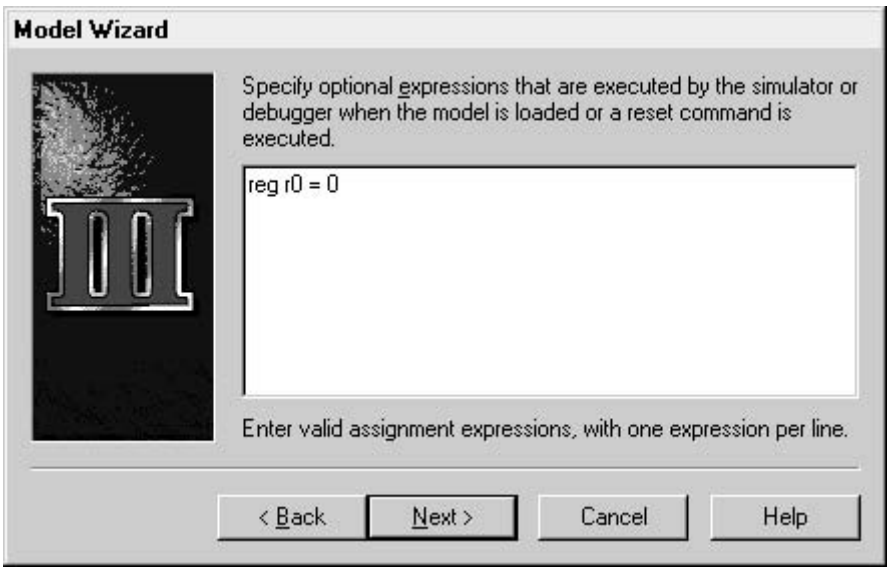
保守／廃止

### 7.3.5　Model ウィザード・ページ III（式処理開始）

このページで，ターゲット・プロセッサのリセット時に実行される代入式を指定します。Ctrl＋Enter キーを押すと，式編集フィールドの行で改行されます。

図 7－5 は式処理開始ページです。

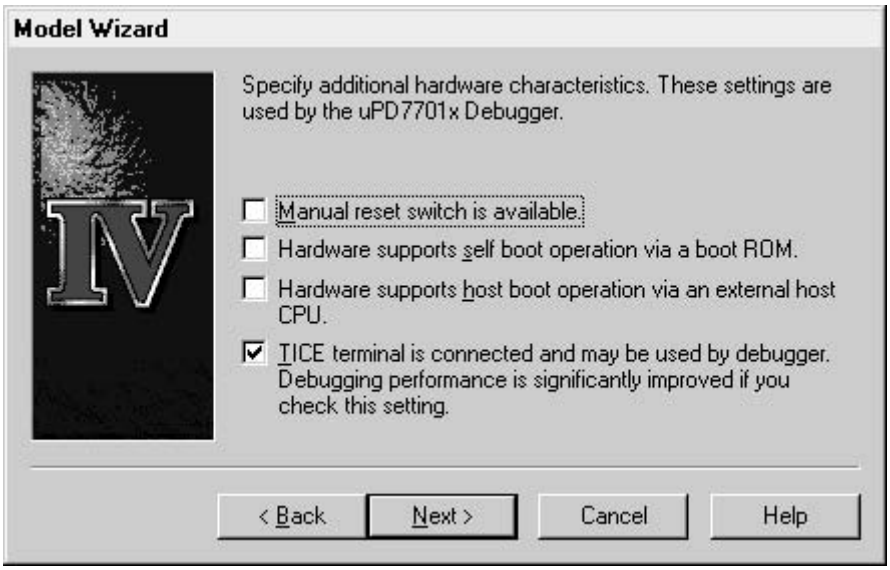
図 7－5　式処理開始ページ




### 7.3.6　Model ウィザード・ページ IV（ディバッガ・ハードウエア設定）

このページで，ディバッガに接続された IE-77016-98/PC に関する設定を行います。図 7－6 はディバッガ・ハードウエア設定ページです。

図 7－6　ディバッガ・ハードウエア設定ページ



保守／廃止

### (1) ディバッガ・ハードウエア設定ページの構成要素

- Manual reset switch is available.

  IE77016-98/PC に接続されたターゲット・システムを手動でリセットできる場合，このオプションにチェック・マークを付けてください。この設定は，Select Start-Up Options ダイアログ・ボックスでの選択に影響します。このダイアログ・ボックスは，ディバグ・セッション中にターゲット・システム・プロパティが変更されている場合または ID77016 起動時に表示されます。Manual reset switch is available オプションにチェック・マークが付いている場合，Select Start-Up Options ダイアログ・ボックスの Wait for externally applied reset signal オプションを利用できます。

- Hardware supports self boot operation via a boot ROM.

  IE-77016-98/PC に接続されたターゲット・システムに起動用 ROM が搭載されている場合，このオプションにチェック・マークを付けてください。この設定は，Select Start-Up Options ダイアログ・ボックスでの選択に影響します。このダイアログ・ボックスは，ディバグ・セッション中にターゲット・システム・プロパティが変更されている場合または ID77016 起動時に表示されます。Hardware supports self boot operation via a boot ROM オプションにチェック・マークが付いている場合，Select Start-Up Options ダイアログ・ボックスの Boot from external memory or host processor オプションを利用できます。

- Hardware supports host boot operation via an external host CPU.

  IE-77016-98/PC に接続されたターゲット・システムが外部メモリまたはホスト・プロセッサで起動できる場合，このオプションにチェック・マークを付けてください。この設定は，Select Start-Up Options ダイアログ・ボックスでの選択に影響します。このダイアログ・ボックスは，ディバグ・セッション中にターゲット・システム・プロパティが変更されている場合または ID77016 起動時に表示されます。Hardware supports host boot operation via an external host CPU オプションにチェック・マークが付いている場合，Select Start-Up Options ダイアログ・ボックスの Boot from external memory or host processor オプションを利用できます。

- TICE terminal is connected and may be used by debugger.

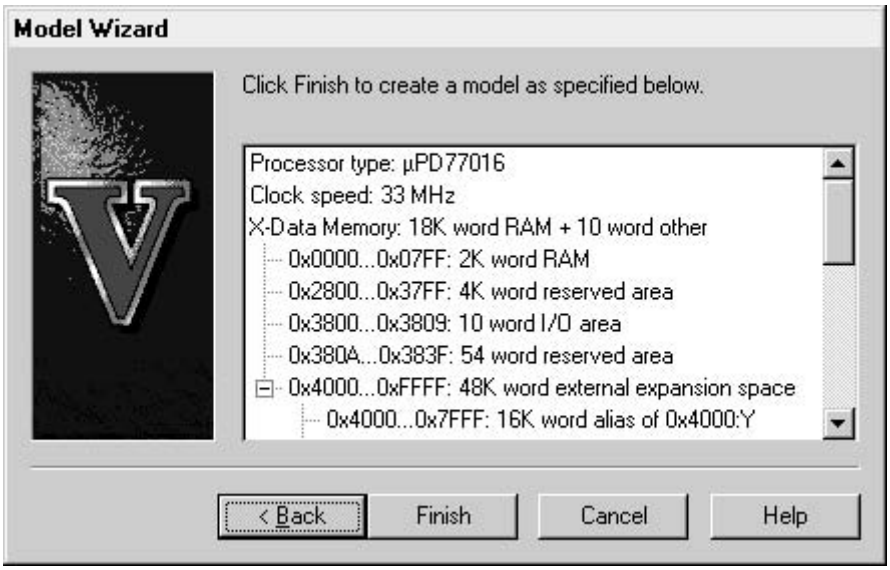
  JTAG ポートの TICE（インサーキット・エミュレーション）端子が JTAG 回線に接続されている場合，このオプションにチェック・マークを付けてください。ID77016 で TICE 端子を使用できると，ディバグ・パフォーマンスが向上します。

保守／廃止

### 7.3.7　Model ウィザード・ページ V（モデル要約情報）

このページには，Model ウィザードで作成したモデルの要約情報，または Shell Extension Properties ダイアログ・ボックスで編集したモデルの要約情報が表示されます。この要約には，モデルの対象となるプロセッサ・タイプとプロセッサ・クロック周波数が含まれます。さらに，ブロック開始アドレス，ブロック・サイズ，およびメモリ・タイプなどのインストラクション・メモリ・ブロックとデータ・メモリ・ブロックの情報も含まれます。これらのモデル属性を確認したら，Finish ボタンを押します。現在アクティブなモデル・ファイルに新しいモデルまたは修正モデルが作成され，いつでも使用できる状態になります。図 7－7 はモデル要約情報ページです。

**図 7－7　モデル要約情報ページ**

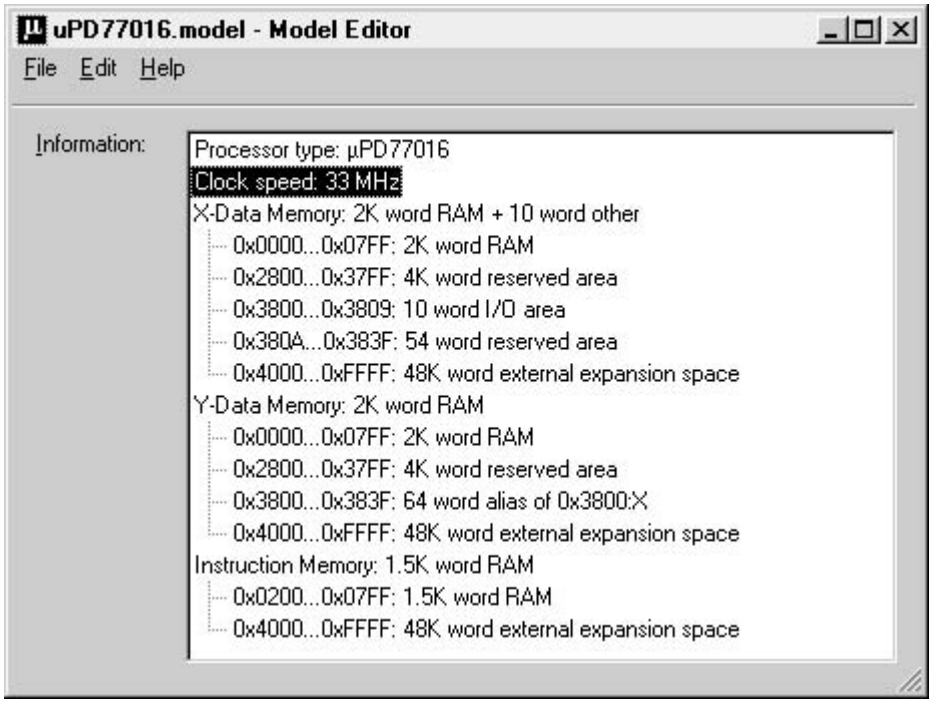


保守／廃止

## 7.4　シェル拡張機能

アプリケーションを起動しないでモデルを変更するには，Windows エクスプローラ内でモデルを右クリックし，コンテキスト・メニューの Open コマンドを実行します。これで，図 7－8 に示した Model Editor のメイン・ウインドウが表示されます。 Model Editor のメイン・ウインドウ内で，7.3 Model ウイザードで説明したモデルとモデル定義ファイルの変更がすべてできます。

図 7－8　Model Editor のメイン・ウインドウ



〔メ　モ〕

# 第8章　タイミング・ファイル

## 8.1 タイミング・ファイルとは

タイミング・ファイルとは，特定の時間に実行される動作またはイベントによって始動される動作を記述しているテキスト・ファイルのことです。各タイミング・ファイルは，μPD77016 ファミリのシミュレーションと，またはほかのタイミング・ファイルと並行に動作する小規模のシミュレーションを記述しています。提供される機能は次のとおりです。

- 複数のタイミング・ファイルは同時に起動できます。これらのファイルの数は，システムの空きメモリやシミュレーションの速度によってのみ制限されます。多くのファイルがアクティブになっている場合には，システム・メモリが減少し，シミュレーション速度が低下します。
- 複数のデータ・ファイルが入出力に使用できます。各タイミング・ファイルは，任意の数のデータ・ファイルとの間で，入出力が可能です。入力または出力固定のタイミング・ファイルは一切必要ありません。
- プロセッサ・クロックとの同期または非同期動作が可能です。
- ほぼすべてのシミュレーション・イベントをタイミング・ファイルの同期化に使用できます。
- ほぼすべてのシミュレーション条件を，テキストを含むデータ・ファイルに出力できます。
- 高度構造化言語は，IF-THEN-ELSE, REPT, DO の各ブロックおよびタイミング・ファイル変数を含んでいます。
- 明示的および非明示的ブレークポイントを指定できます。

タイミング・ファイルは，未処理のテキスト・フォーマットを出力できるテキスト・エディタならどのエディタでも作成が可能です。タイミング・ファイルは，タイミング・ファイル・ウインドウがアクティブの場合には行ごとに実行できます。実行が WAIT コマンドに達すると，タイミング・ファイルの処理は，WAIT 条件が真になるまで当該行にとどまっています。使用できるのは，Run メニューの Run, Break, Trace および Animate | Trace コマンドです。

ステップごとにプログラム・カウンタの値を記録するタイミング・ファイルの例を次に示します。

```
; open the output file which will receive the data

OPEN OUTPUT "TRACE.DAT"
DO              ; loop forever
  OUTPUT "IP=",IP  ; output the IP-value
  WAIT 1         ; wait for next step
ENDDO            ; repeat
```

保守／廃止

## 8.2　タイミング・ファイルの構文とコマンド

タイミング・ファイルは，行ごとに 1 つのコマンドを含んでいるかあるいはコマンドなしのテキスト・ファイルです。

[*blanks* | *tabs* ]... [*command* ] [;*comment* ] CR/LF

次の表は使用できるタイミング・ファイルのコマンドの概要を示しています。

**表 8−1　データ操作コマンド**

| コマンド | 説　明 |
|---|---|
| SET | 式の値をディスティネーションに割り当てる。 |
| OPEN | 入力または出力のデータ・ファイルを開く。 |
| OUTPUT | データまたはテキストをデータ・ファイルに出力する。 |
| INPUT | データ・ファイルからのデータを入力し，データをディスティネーションに割り当てる。 |
| CLOSE | データ・ファイルを閉じて，チャネルが再使用できるようにする。 |

**表 8−2　実行フロー制御コマンド**

| コマンド | 説　明 |
|---|---|
| BREAK | 非明示のブレークポイントでタイミング・ファイルの実行を止める。 |
| IF-ELSE-ENDIF | 条件付き実行。 |
| REPT-ENDREPT | 固定ループ・カウンタ付き実行ループ。 |
| DO-EXIT-ENDDO | エグジット条件付き実行ループ。 |
| END | タイミング・ファイルの実行を止める。 |

**表 8−3　タイミング・コマンド**

| コマンド | 説　明 |
|---|---|
| WAIT Time | タイミング・ファイルの遅延実行。 |
| WAIT Cycle | プロセッサ・ステップ数だけウエイトする。 |
| WAIT Condition | 式の評価値が 0 以外になるまでタイミング・ファイルの実行を中断する。 |

次に，可能な各タイミング・ファイル・コマンドを，その構文といくつかの例とともに示します。使用 *destination* と *expression* に代入される有効値は，HSM77016 の演算子によって組み合わせることができる数字と変数です。HSM77016 の数字のフォーマット，演算子および変数についての詳細は，**第 11 章　レファレンス**を参照してください。

保守／廃止

## 8.2.1　データ操作コマンド

次に示しているのは，ファイルへのデータ入力およびデータ出力に使用するコマンド，ならびにメモリ・セル，レジスタまたはポートを直接修正する際に使用するコマンドです。

（1）SET

*expression* の値を *destination* に割り当てます。割り当てられた値は，コマンド実行時の *expression* の値であって，タイミング・ファイル・ロード時の *expression* の値ではありません。

- 構文

**SET** *destination* **=** *expression*

- 備考

ある値を持つ SET，???，または INPUT がいったん入力／出力ピンまたはポートに使用されると，その値が表明値となります。これはハードウエアの場合にはポートに接続しているドライバ・チップをオンに切り替えることと同等です。その値は，別のコマンドで変更になるまで，または CPU がピンまたはポートを出力に切り替えるまでアクティブのままです。その値を削除するには，SET *destination* = *@@@*を使用します。これはハードウエアの場合にはドライバ・チップをオフに切り替えることと同等です。

*destination* および *expression* の構文については，第 11 章　レファレンスを参照してください。

- 例

;X メモリの位置を 0x380-0x1234 に設定

```
SET *0x3800:X=0x1234
```

;P0 端子を１に設定

```
SET PIN P0=1
```

（2）OPEN

入力または出力のデータ・ファイルを開きます。APPEND が使用される場合，出力のファイルが開かれ，新たなデータが既存のデータの終わりに追加されます。INPUT を使用する場合，そのファイルが存在しなければなりません。*channel* が省略されると，1 が想定されます。チャネルが存在する場合，チャネルの番号の範囲は 1～63 でなければなりません。

- 構文

**OPEN** {**INPUT** | **OUTPUT** | **APPEND**} [#*channel* ] **"***filename* **"**

- パラメータ

**INPUT**　データ入力のファイルを開きます。ファイルはすでに存在していなければなりません。ファイルが存在していない場合，タイミング・ファイル・ランタイム・エラーが発生します。

**OUTPUT**　データ出力のファイルを開きます。ファイルがすでに存在している場合，開く前に削除されます。

**APPEND**　データ出力のファイルを開きます。ファイルがすでに存在している場合，削除されずに，データはファイルの終わりに付加されます。

*channel*　特定のチャネルを占有します。*channel* が指定されないと，1 が想定されます。チャネル番号の範囲は 1～63 でなければなりません。チャネルは空でなければなりません。さもないとタイミ

ング・ファイル・ランタイム・エラーが発生します。INPUT ファイルのチャネルは OUTPUT および APPEND ファイルのチャネルと区別されます。したがって，同じチャネル番号を使用できます。

*filename*　データ・ファイルのファイル名。相対パスは，使用される場合，タイミング・ファイルのパスと関係しています。filename にディフォールト拡張はありません。

• 例

**OPEN INPUT** "MYDATA.DAT"
**OPEN OUTPUT** "E:¥77016¥WORK¥RESULTS.DAT"
**OPEN APPEND** #2 "..¥TRACE.DAT"

(3) OUTPUT

データまたはテキストをデータ・ファイルに出力します。データ・ファイルはあらかじめ OPEN コマンドで開いておかなければなりません。すべてのデータおよびテキストはデータ・ファイルの 1 行に書き込まれます。コマンドがコンマで終了している場合，データ・ファイルには CR/LF は書き込まれません。*channel* の指定が省略されると，1 が想定されます。チャネルが存在する場合，チャネル番号の範囲は 1〜63 でなければなりません。

• 構文

**OUTPUT** [#*channel* ] [{*expression* | "*text* " |
**FORMAT** *output_format_list* }[,]]...
*output_format_list* ::= {*base_option* | *sign_option* |
*showbase_option* | *case_option* }...
*base_option* ::= {**DEC**|**HEX**|**OCT**|**BIN**|**FIX**
[*#_of_bits_before_dot*.*#_of_bits_after_dot* ]|
[*#_of_bits_after_dot* ]}
*sign_option* ::= {**SIGNED**|**UNSIGNED**}
*showbase_option* ::= {**HIDEBASE**|**SHOWBASE**}
*case_option* ::= {**UPPERCASE**|**LOWERCASE**}

• パラメータ

*channel*　特定の出力チャネルを使用します。*channel* が指定されない場合，1 が想定されます。チャネル番号の範囲は 1〜63 でなければなりません。チャネルは，OPEN OUTPUT *#channel* コマンドで開かなければなりません。

*expression*　コマンド実行時に式の値を出力します。この値は，現行出力フォーマット・オプションで変換されます。

"*text* "　テキストを出力します。

**FORMAT**　任意の数の FORMAT ディレクティブを OUTPUT コマンド内で入力できます。これらのディレクティブはデータ・ファイルに何も出力しません。*expression* が指定されない場合，すべての出力チャネルは，別の出力 FORMAT ディレクティブで明示的に変更されないかぎり影響を受けます。ユーザが *output_format_list* でオプションを指定しない場合，前の値はアクティブのままです。

**DEC**　基数が 10 の 10 進数を出力します。

| | |
|---|---|
| **HEX** | **基数が** 16 **の** 16 **進数を出力します。** |
| **OCT** | **基数が** 8 **の** 8 **進数を出力します。** |
| **BIN** | **基数が** 2 **の** 2 **進数を出力します。** |
| **FIX** | **一定の並びで固定小数点数を出力します。** |
| **SIGNED** | **数字を符号付き** 2 **の補数として解釈します。** |
| **UNSIGNED** | **数字を符号なしの数字として解釈します。** |
| **HIDEBASE** | **非基底プレフィクスまたはポストフィクス・ストリングを出力します。** |
| **SHOWBASE** | **基底プレフィクスまたはポストフィクス・ストリングを出力します。** |
| **UPPERCASE** | **数字中の文字を大文字で出力します。** |
| **LOWERCASE** | **数字中の文字を小文字で出力します。** |

- **例**

```
OUTPUT *0x3800:X
OUTPUT FORMAT UNSIGNED HEX,"IP = ",IP,
        FORMAT FIX [9.31], " R0 = ",R0
OUTPUT "R0 = ",R0," R1 = ",R1
;最終行も次のように記述できます
OUTPUT "R0 = ",
OUTPUT R0,
OUTPUT " R1 = ",
OUTPUT R1
```

**（**4**）**INPUT

**データ・ファイルからのデータ値を入力し，ディスティネーションに割り当てます。***channel* **の指定が省略されると，**1 **が想定されます。チャネルが存在する場合，その番号の範囲は** 1〜63 **でなければなりません。**

- **構文**

```
INPUT [ # channel ] [{ destination |
                    FORMAT input_format_list }[,]]...
input_format_list   ::= base_option
base_option         ::= {DEC|HEX|OCT|BIN|FIX
                    [#_of_bits_before_dot.#_of_bits_after_dot ]|
                    [#_of_bits_after_dot ]}
```

保守／廃止

• パラメータ

*channel*　特定の入力チャネルを使用します。*channel* が指定されない場合，1 が想定されます。チャネル番号の範囲は 1〜63 でなければなりません。チャネルは，OPEN INPUT *#channel* コマンドで開かなければなりません。

*destination*　値を *destination* に入力します。この値は現行入力フォーマット・オプションで変換されます。

**FORMAT**　FORMAT ディレクティブは，データ・ファイルからのものは何も入力しません。*expression* が指定されない場合，すべての入力チャネルは，別の入力 FORMAT ディレクティブで明示的に変更されないかぎり影響を受けます。これらのオプションは，ディフォールト・オプションとして適用され，データ・ファイルの値が上書きできます。

**DEC**　基数が 10 の 10 進数を出力します。

**HEX**　基数が 16 の 16 進数を出力します。

**OCT**　基数が 8 の 8 進数を出力します。

**BIN**　基数が 2 の 2 進数を出力します。

**FIX**　一定の並びで固定小数点数を入力します。

• 例

**INPUT** R0

**INPUT FORMAT HEX,**IP

**INPUT FORMAT FIX [**9.31**],**R0

(5) CLOSE

データ・ファイルを閉じ，チャネルを再使用できるようにします。データ・ファイルは，END コマンドまたは end-of-file が検出されたり，あるいはタイミング・ファイルが HSM77016 のタイミング・ファイル・リストから削除されると必ず自動的に閉じます。channel の指定が省略されると，1 が想定されます。チャネルが存在している場合，チャネル番号の範囲は 1〜63 でなければなりません。

• 構文

**CLOSE** {**INPUT**|**OUTPUT**} [*#channel* ]

• パラメータ

**INPUT**　データ入力用のファイルを閉じます。このファイルは OPEN INPUT コマンドで開かなければなりません。OPEN INPUT コマンドで開かなかった場合，タイミング・ファイル・エラーが発生します。

**OUTPUT**　データ出力用のファイルを閉じ，データ・バッファをフラッシュします。このファイルは OPEN OUTPUT または OPEN APPEND コマンドで開かなければなりません。いずれかのコマンドで開かなかった場合，タイミング・ファイル・ランタイム・エラーが発生します。

*channel*　特定のチャネルを閉じ，解放します。channel が指定されない場合，1 が想定されます。

• 備考

このコマンドは出力ファイルに使用します。その場合，作成したデータ・ファイルが調べられる前に使用する必要があります。さもないとデータ・ファイルは不正確なものとなったり不良なものとなることがあります。END コマンド，タイミング・ファイルの end-of-file，またはエラー状態が検出されると，すべてのデータ・ファイルが自動的に閉じるので注意してください。

保守／廃止

• 例

**CLOSE INPUT**

; close channel 3 output file

**CLOSE OUTPUT** #3

## 8.2.2 実行フロー制御コマンド

（1）BREAK

タイミング・ファイルの実行を停止します（非明示的タイミング・ファイル・ブレークポイント）。ブレークの状態が Timing File ウインドウのタイトル・バーに表示されます。実行は再開できます。非明示的ブレークポイントは，ブレークポイント・プロパティの修正を可能にする Breakpoint ウインドウにリストされます。

• 構文

**BREAK**

• 例

**BREAK**

（2）IF

式は，0 以外の値であれば常に真となります。*if_command_lines* が実行されるのは，*if_expression* が真である場合のみです。*elif_command_lines* が実行されるのは，*if_expression* およびすべての先行 *elif_expressions* が偽でかつ現在の *elif_expressions* が真である場合のみです。任意の数の ELIF ブロックを使用できます。else_command_lines が実行されるのは，*if_expression* とすべての *elif_expressions* が偽である場合のみです。

• 構文

**IF** *if_expression*
　[*if_command_line*]...
[ **ELIF** *elif_expression*
　[*elif_command_line*]... ]...
[ **ELSE**
　[*else_command_line*]... ]
**ENDIF**

• 備考

2 つ以上のステートメントが実行されることは絶対にありません。IF コマンドは無制限にネストできます。

保守／廃止

• 例

```
IF *0x400:X == 0x0001
   ; X メモリ 0x400 番地の値が１の場合，常に
ELIF *0x400:X == 0x0002
   ; X メモリ 0x400 番地の値が 2 の場合，常に
ELIF *0x400:X == 0x0003
   ; X メモリ 0x400 番地の値が 3 の場合，常に
ELSE
   ; X メモリ 0x400 番地の値が１,2,3 以外の場合，常に
ENDIF
```

（3）REPT

REPT コマンドと ENDREPT コマンドとの間にあるコマンドは，整数値で指定されている回数だけ実行されます。1 つまたは複数の EXIT コマンド（オプション）があり，かつその式が 0 以外の値である場合，そのループは終了し，コマンド実行は ENDREPT コマンドのあとで続行されます。

• 構文

```
REPT repeat_count
        [[command_line]...
[EXITexpression]]...
ENDREPT
```

• パラメータ

| | |
|---|---|
| *repeat_count* | 0 以外の正の整数。REPT と ENDREPT 間のコマンドが *repeat_count* の回数または EXIT コマンドによって放棄されるまで実行されます。 |
| **EXIT** *expression* | REPT と ENDREPT 間に任意の数の EXIT コマンドを配置することができます。いったん *expression* が真に評価されると，繰り返し情報は破棄され，ENDREPT コマンド直後の実行が続行されます。 |

• 備考

REPT コマンドは，ループ・スタックにスペースがあるかぎり無制限にネストできます。

• 例

```
OPEN OUTPUT "TRACEIP.DAT"
REPT 100
        WAIT 1
        OUTPUT "IP=",IP
        EXIT IP>=0x0400
ENDREPT
CLOSE OUTPUT
```

(4) DO

DO と ENDDO 間のコマンドは，永久に，またはループが EXIT コマンドによって終了するまで実行されます。1 つまたは複数の EXIT コマンド（オプション）があり，かつその式が 0 以外の値である，そのループは終了し，コマンド実行は ENDDO コマンドの後で続行されます。

- 構文

**DO**

[[*command_line*]...

[**EXIT** *expression*]]...

**ENDDO**

- パラメータ

**EXIT** *expression*　DO と ENDDO 間に任意の数の EXIT コマンドを配置することができます。いったん *expression* が真に評価されると，ENDDO コマンド直後に実行が続行します。

- 備考

DO コマンドは無制限にネストできます。

- 例

```
OPEN OUTPUT "TRACEIP.DAT"
DO
  WAIT 1
  OUTPUT "IP=",IP
  EXIT IP>=0x0400
ENDDO
CLOSE OUTPUT
```

(5) END

タイミング・ファイルの実行を止め，すべてのデータ・ファイルを閉じて停止します。実行は再開できません。

- 構文

**END**

- 例

```
END
```

保守／廃止

## 8.2.3 タイミング・コマンド

（1）WAIT Condition

**タイミング・ファイルの実行は** WAIT **コマンドの時点で中断し，**expression **が** 0 **以外の値であると続行されます。**

- **構文**

  **WAIT COND** *expression*

- **パラメータ**

  *expression*　WAIT COND **コマンドが実行中に** expression **が真に評価されると，タイミング・ファイルの実行は中断されることはなく，次のコマンドで続行されます。**

- **例**

  **WAIT COND** (*0x300:X **&** 0x8000) **==** 0x8000

（2）WAIT Cycle

**タイミング・ファイルの実行は** WAIT **コマンドの時点で中断され，シミュレーションされている**μ PD77016 **ファミリの** *cycle_count* **クロック・サイクルがシミュレーションで中に経過した時点で続行されます。**

- **構文**

  **WAIT** [**TIME**] *cycle_count*

- **パラメータ**

  *cycle_count*　**正の整数。ゼロ** (0) **が指定されると，タイミング・ファイルはプロセッサのシミュレーションに再同期されます。**

- **例**

  **WAIT** 3

  **WAIT TIME** 0

（3）WAIT Step

**タイミング・ファイルの実行は，**WAIT **コマンドの時点で中断され，シミュレーションされている**μ PD77016 **ファミリの** *step_count* **ステップがシミュレーション中に経過した時点で続行されます。**

- **構文**

  **WAIT STEP** *step_count*

- **パラメータ**

  *step_count*　**正の整数。ゼロ** (0) **が指定されると，タイミング・ファイルはプロセッサのシミュレーションに再同期します。**

- **例**

  **WAIT STEP** 5

保守／廃止

（4）WAIT Time

**タイミング・ファイルの実行は，指定した時間（シミュレーション時間）にわたり中断します。**

- **構文**

**WAIT** [**TIME**] *delay_time* {**NS**|**US**|**MS**|**S**}

- **パラメータ**

| | |
|---|---|
| *delay_time* | **正の整数** |
| **NS** | **単位がナノ秒（**ns, $10^{-9}$**秒）の** *Delay_time* |
| **US** | **単位がマイクロ秒（**$\mu$s, $10^{-6}$**秒）の** *Delay_time* |
| **MS** | **単位がミリ秒（**ms, $10^{-3}$**秒）の** *Delay_time* |
| **S** | **単位が秒（**s**）の** *Delay_time* |

- **例**

**WAIT** 300 **ns**

**WAIT** 12 **us**

## 8.2.4 タイミング・ファイルの変数

**タイミング・ファイルでは，グローバルおよびローカル変数を指定できます。グローバル変数は，対応変数のグローバル宣言を含んでいるタイミング・ファイルすべてにとって共通なものです。ローカル変数は，ローカル宣言を含んでいるタイミング・ファイルで有効です。**Watch **ウインドウでタイミング・ファイル変数を参照する方法ついては** 5.4　Watch **ウインドウを参照してください。**

- **構文**

**GLOBAL** *var_name* [**,***var_name*]...

**LOCAL** *var_name* [**,***var_name*]...

- **パラメータ**

| | |
|---|---|
| *var_name* | **指定された変数の名前** |

- **備考**

**有効な変数名は，文字，数字およびアンダスコアを組み合わせたものです。変数名は，先頭が数字であってはならず，またブランクを含んではなりません。変数は，使用する前に宣言しなければなりません。**

- **例**

**GLOBAL** x,b_lock,_2loop

**LOCAL** y,reg_load,_varset1

保守／廃止

# 8.3　タイミング・ファイルの例

HSM77016 配に同梱されているタイミング・ファイル例を示します。

## 8.3.1　タイミング・ファイル HOSTRD16.TMG

```
; file:  HOSTRD16.TMG
; purpose:        Demonstrate sixteen bit data
;                 transfer via host interface
;          Read from low and high-byte of
;          built-in host port's HDT register
;          Output data values to data file
;          HOSTRD16.DAT
; software: HOSTDEMO.LNK must be loaded

local lowbyte      ; local variable to hold low byte of  ; data
set pin hcs = 1    ; terminate any read
; access, which
; might be active
set pin hrd = 1
open output "HOSTRD16.DAT"   ; data are stored in
; file "HOSTRD16.DAT"
output format showbase unsigned hex,
; select output format
rept 10            ; loop to read 10 bytes of
; data
  wait cond pin hre == 0    ; wait until data is
; available and no write
  wait cond pin hcs == 1    ; is in progress
  set pin hcs = 0           ; perform the access...
  set port ha = 0           ; select lower byte of HDT
  set pin hrd = 0           ; start output
  wait 100ns                ; access duration
  set lowbyte = port hd&0xFF; output low byte to temp
; variable
  set pin hrd = 1           ; end output
  wait 5ns                  ; delay
  set port ha = 1           ; select higher byte of HDT
  wait 5ns         ; delay
  set pin hrd = 0           ; start output
  wait 100ns                ; access duration
  output ((port hd&0xFF)<<8)|lowbyte
; output word data to file
  set pin hrd = 1           ; end output
  set pin hcs = 1           ; end host port access
endrept

close output
end
```

保守／廃止

## 8.3.2 タイミング・ファイル HOSTRD8.TMG

```
; file:    HOSTRD8.TMG
; purpose:  Demonstrate eight bit data transfer via
;         host interface
;         Read from high-byte of built-in host
;         port's HDT register
;         Output data values to data file HOSTRD8.DAT
; software: HOSTDEMO.LNK must be loaded

set pin hcs = 1           ; terminate any read
; access, which might
set pin hrd = 1           ; be active
open output "HOSTRD8.DAT"  ; data are stored in file
; "HOSTRD8.DAT"
output format showbase unsigned hex,
; select output format
rept 10                   ; loop to read 10 bytes of
; data
   wait cond pin hre == 0  ; wait until data is
; available
   wait cond pin hcs == 1  ; and no write is in
; progress
   set pin hcs = 0         ; perform the access...
   set port ha = 1         ; select higher byte of
; HDT
   set pin hrd = 0         ; start output
   wait 100ns              ; access duration
   output port hd & 0xFF   ; output host data to file
; (mask sign bits)
   set pin hrd = 1         ; end output
   set pin hcs = 1         ; end host port access
endrept

close output
end
```

保守／廃止

## 8.3.3 タイミング・ファイル HOSTWR16.TMG

```
; file:    HOSTWR16.TMG
; purpose:  Demonstrate sixteen bit data transfer
;         via host interface
;         Input data values from file HOSTWR16.DAT
;         Write to low and high-byte of built-in
;         host port's HDT register
; software: HOSTDEMO.LNK must be loaded.

local data                ; local variable receives
; data
set pin hcs = 1           ; terminate any write
; access, which might
set pin hwr = 1           ; be active

open input "HOSTWR16.DAT"  ; data are read from the
; file "HOSTWR16.DAT"
input format hex          ; select hex input format
rept 10                   ; loop to read 10 bytes of
; data
   wait cond pin hwe == 0   ; wait till write is
; allowed and no read is in
   wait cond pin hcs == 1  ; progress
   set pin hcs = 0         ; perform the access...
   set port ha = 0         ; select higher byte of HDT
   set pin hwr = 0         ; start input
   input data              ; input host data to temp
; variable
   set port hd = data&0xFF ; input low byte to host
; port
   wait 100ns              ; access duration
   set pin hwr = 1         ; end output
   wait 5ns                ; delay
   set port ha = 1         ; select higher byte of HDT
   wait 5ns                ; delay
   set pin hwr = 0         ; start output
   set port hd = (data>>8)&0xFF
; input high byte to host
; port
   wait 100ns              ; access duration
   set pin hwr = 1         ; end input
   set pin hcs = 1         ; end access
endrept

close input
end
```

保守／廃止

## 8.3.4 タイミング・ファイル HOSTWR8.TMG

```
; file:    HOSTWR8.TMG
; purpose:  Demonstrate eight bit data transfer via
;         host interface
;         Input data values from file HOSTWR8.DAT
;         Write to high-byte of built-in host
;         port's HDT register
; software: HOSTDEMO.LNK must be loaded.

set pin hcs = 1           ; terminate any write
; access, which might
set pin hwr = 1           ; be active
set pin hcs  = 0          ; initialize low byte with 0
set port ha = 0           ;
set pin hwr = 0           ;
set port hd = 0           ;
wait 100ns                ;
set pin hwr = 1           ;
set pin hcs = 1           ;
set port hd = @           ;

open input "HOSTWR8.DAT"  ; data are read from the
; file "HOSTWR8.DAT"
input format hex          ; select hex input format
rept 10                   ; loop to read 10 bytes of
; data
   wait cond pin hwe == 0 ; wait till write is allowed
   wait cond pin hcs == 1 ; and no read is in progress
   set pin hcs = 0        ; perform the access...
   set port ha = 1        ; select higher byte of HDT
   set pin hwr = 0        ; start input
   input port hd          ; input host data from file
   wait 100ns             ; access duration
   set pin hwr = 1        ; end input
   set pin hcs = 1        ; end access
endrept

close input
end
```

## 8.3.5 タイミング・ファイル SER1CLK.TMG

```
; file:  SER1CLK.TMG
; purpose:  Generate 5 MHz clock signal for serial
;         port 1 clock line SCK1
;         Clock generator for SER1OT16.TMG and
;         SER1IN16.TMG timing files
; software:   ---
; serial cnfg: port 1
; total cycle time = 25ns+25ns = 50ns ==> 20MHz

do
   set pin sck1 = 0
   wait 25ns
   set pin sck1 = 1
   wait 25ns
enddo
```

保守／廃止

## 8.3.6 タイミング・ファイル SER1IN16.TMG

```
; file:       SER1IN16.TMG
; purpose:    Demonstrate sixteen bit data transfer
;            via serial interface
;            Load values from SER1IN16.DAT
;            Shift input values to built-in serial
;            interface
; software:   SER1DEMO.LNK must be loaded
;            SER1CLK.TMG must be installed
; serial cnfg: port 1, 16 bits

set pin sien1 = 0          ; initialize SIEN1 line
open input "SER1IN16.DAT"  ;
rept 9                     ; loop to input 10 values
; from file
  if pin sien1 == 0        ; do only if new
; transmission start
    wait cond pin siak1 == 1        ; wait for serial input
; request
    wait 5 ns              ; logic delay
    set pin sien1 = 1      ; start serial input
  endif                    ;
  wait cond pin siak1 == 0 ; wait for serial input
; start confirmation
  set pin sien1 = 0        ;
  input port si1           ; read data from file to
; port SI1
; Note: Input to PORT SI1 performs shifting automatically. ; Input to PIN SI1 requires
explicit shifting procedure in
; timing file.

  rept 15                  ; wait 15 clock cycles for
; one data frame
    wait cond pin sck1 == 0; wait for rising edge
; (0->1)
    wait cond pin sck1 == 1         ;
  endrept                  ;
  wait 5ns                 ; make sure SIAK1 is updated
  if pin siak1 == 1        ; request next input
    set pin sien1 = 1      ; start next serial input
  endif                    ;

  wait cond pin sck1 == 0  ; wait for rising edge
; (0->1)
  wait cond pin sck1 == 1  ;
endrept                    ;
                           ; input last value without
; starting another input
wait cond pin siak1 == 0   ; wait for serial input
; start confirmation
set pin sien1 = 0          ;
input port si1             ; read data from file to
; port SI1
close input                ; close data file
end
```

保守／廃止

## 8.3.7　タイミング・ファイル SER1OT16.TMG

```
; file:       SER1OT16.TMG
; purpose:    Demonstrate sixteen bit data transfer
;            via serial interface
;            Shift output values from built-in serial
;            interface
;            Save values to SER1OT16.DAT
; software:   SER1DEMO.LNK must be loaded
;            SER1CLK.TMG must be installed
; serial cnfg: port 1, 16 bits

set pin soen1 = 0          ; initialize SOEN1 line
open output "SER1OT16.DAT"          ;
output format showbase unsigned hex,
; select output format
rept 10                    ; loop to input 10 values
; from file
  if pin soen1 == 0        ; do only if new
; transmission start
    wait cond pin sorq1 == 1; wait for serial output
; request
    wait 5 ns              ; logic delay
    set pin soen1 = 1      ; start serial output
  endif

  wait cond pin sorq1 == 0  ; wait for serial output
; start confirmation
  set pin soen1 = 0        ;
  rept 15                  ; wait 15 clock cycles for
; one data frame
    wait cond pin sck1 == 0; wait for rising edge
; (0->1)
    wait cond pin sck1 == 1         ;
  endrept                  ;
  wait 5ns                 ; make sure SO1 and SORQ1
; are updated
  output port so1&0xFFFF    ; write output data to
; file, mask sign bits
    if pin sorq1 == 1      ; request next output
    set pin soen1 = 1      ; start next serial output
  endif

  wait cond pin sck1 == 0   ; wait for rising edge
; (0->1)
  wait cond pin sck1 == 1   ;
endrept                    ;
close output               ; close data file
end
```

〔メ　モ〕

# ★ 第9章　アプリケーション・プログラム・インタフェース

## 9.1 概　　要

HSM77016 に組み込まれているアプリケーション・インタフェースは，カスタム作成関数（プラグイン・タイプ）がシミュレーションに接続できるようにします。カスタム関数は，次の領域において，その機能を拡張できます。

- カスタム・メモリ・マップ I/O レジスタのシミュレーションと動作
- タイミング・イベント，レジスタ，端子またはポート条件に基づくカスタム論理のシミュレーション
- 内蔵レジスタ，端子，またはポートの場合と同じ方法で HSM77016 ユーザ・インタフェースからアクセスできるカスタム・レジスタ，端子またはポートの作成
- カスタム・シミュレーションの拡張関数の状態を視覚化するカスタム・ウインドウおよびカスタム・メニューの作成

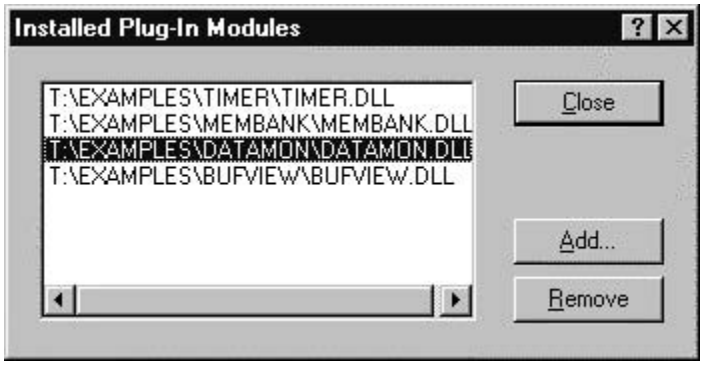
カスタム関数ライブラリの作成は，ユーザが 32 ビット Windows DLL 実行可能プログラム（Microsoft Visual C++® 2.0 以降など）を使用して行います。カスタム関数ライブラリは，HSM77016 が関数ライブラリとコミュニケートできるように関数をエクスポートする必要があります。カスタム関数ライブラリは，ラン・タイム時に HSM77016 とリンクします。したがって HSM77016 のユーザは，HSM77016 の実行可能プログラムに修正を加えることなくカスタム関数ライブラリのすべての関数を自由に実行させることができます。

この章では，アプリケーション・プログラム・インタフェースについて詳しく説明します。

## 9.2 ユーザ・インタフェース

アプリケーション・プログラム・インタフェースのユーザ・インタフェース部分は，Tools メニューからアクセスできます。Plug-In Modules...コマンドは，カスタム関数をシミュレーションに接続するために使用する Installed Plug-In Modules ダイアログを表示します。図 9－1に Installed Plug-In Modules ダイアログを示します。

図 9－1　Installed Plug-In Modules ダイアログ



保守／廃止

Installed Plug-In Modules ダイアログの構成要素

- Plug-In Modules List ：現在インストールされているプラグイン・モジュールを表示します。
- Add... ：プラグイン・モジュールをブラウズするためのファイル選択ダイアログを表示します。
- Remove ：選択したプラグイン・モジュールをシミュレーションから削除します。
- Close ：ダイアログを閉じます。

## 9.3 カスタム・ウインドウとウインドウ・メニュー

ユーザが作成したカスタム関数ライブラリは，HSM77016 のユーザ・インタフェースを独自のウインドウとメニュー関数で拡張することもできます。

Windows ダイアログ・テンプレートは，カスタム・ウインドウを機能と体裁の点から記述するために使用します。そのようなテンプレートは，ウインドウ開発のための標準ツール（Microsoft Visual C++など）を使用すれば作成できます。作成されたテンプレートは，ユーザ DLL にダイアログ・テンプレートとして格納されます。

次のウインドウ制御は，カスタム・ウインドウ用にサポートされています。

- Edit Control：カスタム・レジスタ，端子，およびポートを表示するための Edit Cells （例　*0x4000:X）
- Check Boxes：カスタム・ビットを編集，表示する Edit Cells
- Static Control：ラベル

カスタム・ウインドウは，レジスタ・ウインドウと外観が似ており，HSM77016 のユーザ・インタフェースにシームレスに統合されています。

- カスタム・ウインドウは，ウインドウ・メニューのコマンドで開くことができます。
- データの表示は，既存のフォーマット・オプションを使用すれば構成できます（2 進，10 進，16 進，...）。
- エディット・バーは，カスタム編集セル内のデータ値を編集するために使用します。
- すべての表示色は，Options...ダイアログで構成できます。
- カスタム・ウインドウの設定値は，設定ファイルに格納されます。

各カスタム・ウインドウは，カスタム・ウインドウがアクティブである場合にのみ使用できるメニューをサポートしています。また，ユーザによって HSM77016 の Tools メニューにカスタム・メニューを定義して追加することもできます。メニューは，ユーザ DLL のメニュー・リソースとして定義しなければなりません。このように定義したカスタム・メニュー・コマンドを起動した場合，HSM77016 はユーザ DLL にある定義済み関数を呼び出します。

保守／廃止

### 9.3.1　HSM77016 メニューの拡張

メニュー拡張機能により，トップ・レベルのプルダウン・メニューを挿入するだけでなく，新しいメニュー項目を HSM77016 のプルダウン・メニューに追加することができます。そのようなメニューのメッセージはすべて DLL に送られます。DLL は，メニュー・テンプレート・リソースを使用してメニューを定義します。メニュー・テンプレートは，メニュー・バーの項目およびすべてのメニューを含むメニューを定義します。メニュー・テンプレート・リソースを作成する方法については，DSP 開発ツールに添付されているマニュアルを参照してください。メニューを既存のメニュー・アーキテクチャに追加する場合は，メニュー・テンプレート・リソースのトップ・レベルのメニューすべてを HSM77016 のメニュー・バーに挿入します。新しいポップアップ・メニューがすでに存在しているトップ・レベル・メニューに対応している場合，このメニューに追加されます。メニュー項目は，File および Window メニューに追加できません。DLL メニューの定義はメニュー・リソースを使用して行われます。HSM77016 はそのリソースをロードし，すべてのコマンドおよび更新メッセージを DLL 定義関数に送ります。

**（1）メニュー**ID **のマッピング**

カスタム・メニューのメニュー項目 ID の定義には制約はありません。しかし，メニュー・メッセージを正しく送るために，メニュー項目 ID は HSM77016 内で一意の項目 ID を保証できるように変換されます。この変換プロセスは完全に隠されており，DLL サイドからのプログラミングを必要としません。DLL がカスタム・メニュー項目を更新，修正できるようにするには，変換された ID を使用しなければなりません。したがって，HSM77016 は，メニュー・テンプレート・リソースで記述されているオリジナルの ID と変換された ID とをパラメータとして CommandProc および UpdateCommandProc 関数に送ります。

**（2）メニュー・プロンプト・ストリング**

HSM77016 のステータス・バーには，メニューの各項目間でマウスを動かす際に便利なフィードバック機能が備わっています。メニュー項目を選択している際には，ステータス・バーにプロンプト・ストリングが自動的に表示されます。

ユーザが HSM77016 に追加したメニュー項目にプロンプト・ストリングを定義すると，ステータス・バーにヘルプを簡単に追加することができます。そのようなプロンプト・ストリングは，DLL のリソースのストリング・リソースとして定義されます。それらのストリングは，説明するコマンドと同じ ID（未変換のオリジナル ID）を持っていなければなりません。

保守／廃止

**（3）カスタム・メニューを** HSM77016 **に追加**

**カスタム・メニューを** HSM77016 **に追加する方法は次のとおりです。**

1. **リソース・ファイルを直接編集するために，リソース・エディタ（**Microsoft Developer Studio **リソース・エディタなど）またはプレーン・テキスト用エディタを使用してメニュー・リソースを作成します。**
2. **メニュー・プロンプトをストリング・リソースとして作成します。これはオプションです。**
3. **ユーザのメニューからのコマンド・メッセージを処理する，**DLL **の CommandProc コールバック関数（**function**）を定義します。**
4. **必要に応じてメニューを更新および修正する，**DLL **の UpdateCommandProc コールバック関数を定義します。これはオプションです。**
5. **MENUDESC 構造体に組み込みます。**
   - dwSize **メンバを，MENUDESC 構造体のバイト・サイズに設定します。**
   - hInst **メンバを，**DLL **のインスタンス・ハンドルに設定します。**
   - lpMenu **メンバを，メニュー・リソースのリソース識別子に設定します。**
   - lpCommandFunc **メンバを，CommandProc コールバック関数に設定します。**
   - lpUpdateCommandFunc **メンバを，**UpdateCommandProc **コールバック関数に設定します。**
     **UpdateCommandProc コールバック関数が用意されていない場合には** NULL **に設定します。**
6. **MENUDESC 構造体を UninstallUIExtension 関数に渡し，戻されたハンドルを記憶しておきます。**

**メニューは，必要でなくなったとき（通常** DllMain **内）には，必ず UninstallUIExtension 関数で削除してください。**

## 9.3.2 HSM77016 ウインドウの拡張

**ウインドウ拡張機能から，**HSM77016 **に新たな子ウインドウを追加することができます。そのような子ウインドウはほかのすべての** HSM77016 **ウインドウ（レジスタ，メモリなど）と同じ基本機能を持っています。これらのウインドウには，タイトル・バー，システム・メニュー，およびウインドウの最小化，最大化およびクローズするためのボタンが表示されます。**DLL **ウインドウを開く場合は，**HSM77016 **のウインドウ・メニューにあるメニュー・コマンドを使用します。各** DLL **ウインドウには，アクティブの場合にのみ使用できる関連メニューが用意されています。**

DLL **ウインドウの定義は，ダイアログ・テンプレート・リソースを使用して行います。このテンプレートでは，すべての種類のウインドウ・コントロール（エディット・コントロール，チェック・ボックスなど）とユーザ定義のコントロールが使用できます。ウインドウの共通コントロールのほかに，**HSM77016 **は，レジスタ，端子，およびポートを表示，編集する** ITEMLIST **という特殊なコントロールを定義しています。すべてのウインドウ・メッセージは，**DLL **内にある** DlgProc **関数に送られます。**

保守／廃止

**(1) ITEMLIST コントロール**

ITEMLIST コントロールは，グリッドのように描画されるセルのレジスタ，端子，およびポートなどの HSM77016 データ項目のデータ値を表示するカスタム・ウインドウ・コントロールです。このコントロールでは，HSM77016 が扱えるすべてのデータ・フォーマット（2 進，10 進，16 進など）でデータ値を表示したり，エディット・バーを使用してそれらの値を編集することができます。さらに，縦に配置され，グリッド・ラインで区切られた列にデータ項目を入力することで，複数のデータ項目を表示することもできます。それぞれのセルの列は選択できますが，アンカを持つことができることのは１つのセルのみです。アンカを持っているセルのみが編集できます。

ITEMLIST コントロールのクラス名は，ITEMLIST です。このコントロールは，作成時に，ウインドウ・テキスト・タイトル・テキストから HSM77016 のデータ項目の名前リストを検索します。ウインドウ・タイトルに 2 つ以上の項目名が含まれている場合，その名前は，REG R0＼nREG R1 のように“＼n”で区切ります。次に，ダイアログ・テンプレート内での ITEMLIST コントロール宣言の例をします。

```
CONTROL   "REG IP"，IDC_MYCTRL，"ITEMLIST"，WS_TABSTOP. 136，13，50，16
```

**(2) WM_REFRESHDISPLAY メッセージ**

WM_REFRESHDISPLAY メッセージは，すべての子ウインドウの画面更新を強制的に行う HSM77016 定義のメッセージです。このメッセージは，次のイベント群の 1 つが発生したときにすべての HSM77016 の子ウインドウに送られます。

- RUN，STEP，または TRACE コマンドのあとでシミュレーションが停止した。
- プロセッサのリセット
- ファイルのロード
- モデルの変更
- 編集によってメモリ値またはレジスタ値が変わった。

**(3) カスタム子ウインドウを HSM77016 に追加**

カスタム子ウインドウを HSM77016 に追加する方法は次のとおりです。

1. リソース・エディタを使用して，ウインドウの内容を定義するダイアログ・リソースを作成します。ウインドウに関連メニューがない場合，ステップ 8 に進みます。
2. ウインドウとともに使用するメニュー・リソースを作成します。
3. ストリング・リソースとしてメニュー・プロンプトを作成します。これはオプションです。
4. 当該ウインドウのウインドウ・メッセージすべてを処理する DlgProc コールバック関数を定義します。
5. ユーザのメニューからのコマンド・メッセージを処理する，DLL の CommandProc コールバック関数を定義します。
6. 必要に応じてメニューを更新および修正する，DLL の UpdateCommandProc コールバック関数を定義します。これはオプションです。
7. ダイアログ・リソース・テンプレートの MENU ステートメントを使用してメニューをウインドウにアタッチします。
8. ウインドウとともに使用するアイコン・リソースを作成します。リソース識別子は，ウインドウと同じ ID を持っていなければなりません。これはオプションです。


9. **WINDOWDESC 構造体を組み込みます。**
   - dwSize **メンバを，WINDOWDESC 構造体のバイト・サイズに設定します。**
   - hInst **メンバを，**DLL **のインスタンス・ハンドルに設定します。**
   - lpTemplate **メンバを，ダイアログ・リソースのリソース識別子に設定します。**
   - lpDialogFunc **メンバを，DlgProc コールバック関数に設定します。**
   - lpszMenuText **メンバを，当該ウインドウを開くために** HSM77016 **のウインドウ・メニューに表示したいメニュー・コマンドのテキスト・ストリングに設定します。**
   - lpszMenuMessage **メンバを，メニュー・コマンドが選択されたときにステータス・バーにメニュー・プロンプトとして表示されるテキスト・ストリングに設定します。**
   - lpCommandFunc **および** lpUpdateCommandFunc **メンバを，ユーザが定義したコールバック関数に，あるいは必要ない場合には** NULL **に設定します。**
10. **WINDOWDESC 構造体を InstallUIExtension 関数に渡し，戻されたハンドルを記憶しておきます。**

**ウインドウは，必要でなくなったとき（通常** DllMain **内）には，必ず UninstallUIExtension 関数で削除してください。**

## 9.4 関数参照

### 9.4.1 ユーザ DLL によってエクスポートされる関数

**（**1**）**CommandProc

**CommandProc 関数は，**DLL **定義のコールバック関数です。この関数は，InstallUIExtension 関数で定義されたカスタム・メニュー拡張部分のコマンド・メッセージ（**WM_COMMAND**）で呼び出されます。**

- **構文**

```
int CommandProc(
  UINT nIDOrg,
  UINT nIDTrans
);
```

- **パラメータ**

| | |
|---|---|
| *nIDOrg* | DLL **リソースで指定されているように，メニュー項目，コントロール，またはアクセラレータのオリジナルの識別子を指定します。** |
| *nIDTrans* | **メニュー項目，コントロール，またはアクセラレータの変換後の識別子を指定します。** |

- **戻り値**

**コマンド・プロシージャはメッセージを処理する場合には** 0 **を戻します。処理しない場合には** 0 **以外の数値を戻します。**

保守／廃止

（2）CreateMemoryMappedArea

**CreateMemoryMappedArea 関数は，**HSM77016 **がロードしたモデル定義がユーザ** DLL **のメモリ・マップ領域を指定するときに呼び出されます。**

**• 構文**

```
HANDLE CreateMemoryMappedArea(
  DWORD dwAreaBase,     // メモリ・マップ領域のベース・アドレス
  WORD wSize,           // メモリ・マップ領域のサイズ
  LPCSTR pszOpt         // 初期化ストリング（オプション）
);
```

**• パラメータ**

| | |
|---|---|
| *dwAreaBase* | *dwAreaBase*　**モデル定義で定義されたメモリ・マップ領域のアドレス。アドレス** 0x00000-0x0FFFF **は** X **データ・メモリを，アドレス** 0x10000-0x1FFFF **は** Y **データ・メモリをそれぞれ示します。** |
| *wSize* | **モデル定義で定義されているメモリ・マップ領域のサイズ。** |
| *pszOpt* | **モデル定義で定義されている初期化ストリング（オプション）。**HSM77016 **は初期化ストリングをモデル定義からユーザ** DLL **に転送します。**HSM77016 **は，このストリングはその書式にも変換されません。ユーザ** DLL **は，このストリングを，任意の書式で自由に使用できます。** |

**• 戻り値**

**関数が正しく実行されると，戻り値は新たに作成されたメモリ・マップ領域のハンドルとなります。このメモリ・マップ領域を参照する際，**HSM77016 **は関数の呼び出しにこのハンドラをユーザ** DLL **に適用します。**

**関数が正しく実行されなかった場合，戻り値は** 0 **になります。**

**• 備考**

**メモリ・マップ領域は常に，**X **データ・メモリまたは** Y **データ・メモリ空間内に収まっています。**HSM77016 **は，**X **データ・メモリで始まり，**Y **データ・メモリで終わるメモリ・マップ領域を作成することはありません。**

**例：次の行を含むモデルがロードされます。**

```
XEXT1=0x9800,0x0400,USRDLL.DLL,XXX
```

HSM77016 **は** USRDLL.DLL **をロードし，当該** DLL **の CreatMemoryMappedArea**（0x9800, 0x0400, "XXX"） **を呼び出します。**

保守／廃止

(3) DestroyMemoryMappedArea

**DestroyMemoryMappedArea 関数が呼び出されるのは，CreatMemoryMappedArea 関数によって作成されたメモリ・マップ領域が必要でなくなった場合です。**

**• 構文**

```
int DestroyMemoryMappedArea(
    HANDLE hndlArea          // メモリ・マップ領域のハンドル
);
```

**• パラメータ**

| | |
|---|---|
| *hndlArea* | **破棄されるメモリ・マップ領域のハンドル。このハンドルは，CreatMemoryMappedArea 関数によって戻されます。** |

**• 戻り値**

**関数が正しく実行された場合，戻り値は** 0 **です。**

**関数が正しく実行されなかった場合，戻り値はエラー・コードです。**

(4) DialogProc

**DialogProc 関数はアプリケーション定義のコールバック関数で，モードまたはモードレス・ダイアログ・ボックスに送られたメッセージを処理します。**

**• 構文**

```
BOOL CALLBACK DialogProc(
    HWND hwndDlg,            // ダイアログ・ボックスへのハンドル
    UINT uMsg,               // メッセージ
    WPARAM wParam,           // 最初のメッセージ・パートナー
    LPARAM lParam            // 2 番目のメッセージ・パートナー
);
```

**• パラメータ**

| | |
|---|---|
| *hwndDlg* | **ダイアログ・ボックスを識別します。** |
| *uMsg* | **メッセージを指定します。** |
| *wParam* | **メッセージ独自の追加情報を指定します。** |
| *lParam* | **メッセージ独自の追加情報を指定します。** |

**• 戻り値**

WM_INITDIALOG **メッセージへの応答が禁じられているので，ダイアログ・ボックス・プロシージャはメッセージを処理する場合には** 0 **以外の値を戻し，処理しない場合には** 0 **を戻します。**WM_INITDIALOG **メッセージに応答して，ダイアログ・ボックス・プロシージャはダイアログ・ボックスのコントロール群の** 1 **つにフォーカスを設定するために SetFocus 関数を呼び出す場合には** 0 **を戻します。それ以外の場合には** 0 **以外の値を戻し，その場合システムはフォーカスが与えられるダイアログ・ボックスの最初のコントロールにフォーカスを設定します。**

保守／廃止

(5) EventProc

**EventProc 関数は，**DLL **定義の関数であり，CreateXxxEventCallback 関数に対して指定したイベントが発生したときに** HSM77016 **によって呼び出されます。EventProc 関数は CreateXxxEventCallback 関数へと移行します。EventProc 名は，ユーザ** DLL **から供給される関数名のプレース・ホルダです。実際の名前はエクスポートする必要があります。**

**• 構文**

```
int EventProc(
    HEVENT hEvent,          // イベント・ハンドル
    DWORD dwEventData,      // 使用しません
    PVOID pData             // ユーザ・データ
);
```

**• パラメータ**

*hEvent*　**コールバック関数を呼び出すイベントのイベント・ハンドル。このハンドルは，このイベントを作成した CreateXxxEventCallback 関数によって戻されます。**

*dwEventData*　**このパラメータはイベント特有のものです。その内容は，コールバック関数を呼び出したイベントをどの CreateXxxEventCallback 関数で作成したかによって異なります。**

| 関　数 | *dwEventData* の内容 |
|---|---|
| CreateTimeEventCallback | 使用しません。常に 0 に設定されます。 |
| CreateStepEventCallback | 使用しません。常に 0 に設定されます。 |
| CreateItemEventCallback | CreateItemEventCallback 関数に渡される項目ハンドル |
| CreateMemRdEventCallback | イベントを引き起こしたメモリ・アドレス |
| CreateMemWrEventCallback | イベントを引き起こしたメモリ・アドレス |

*pData*　**ユーザが供給したデータ。このデータは，当該イベントを引き起こした CreateXxxEventCallback 関数の** pData **引数で設定されます。このパラメータは，CreateXxxEventCallback 関数からの追加情報をコールバック関数に渡すのに使用できます。**

**• 戻り値**

**関数は，次のコード群の** 1 **つを戻すことがあります。**

**EVT_STOP_EVENT_NOTIFICATION**

HSM77016 **は，たとえイベントが発生してもコールバック関数をふたたび呼び出し直すことはありません。ここで注意していただきたいのは，コールバック関数は呼び出されませんが，CreateXxxEventCallback 関数によって作成されたイベント・オブジェクトは依然として存在し，必要でなくなった場合には DestroyEventCallback 関数で破棄しなければならないということです。DestroyEventCallback 関数は，イベント・コールバック関数内から使用してはいけません。EnableEventCallback 関数は，イベント・コールバック関数から** EVT_STOP_EVENT_NOTIFICATION **を戻すことによっていったんディスエーブルされた，コールバック関数の呼び出しを再度イネーブルするのに使用できます。**

保守／廃止

**EVT_CONTINUE_EVENT_NOTIFICATION**

HSM77016 は，イベントが再度発生した場合にはコールバック関数を呼び出します。

**EVT_BREAK_EXECUTION**

HSM77016 はシミュレーションをブレークし，コントロールをユーザに戻します。

(6) ItemDataProc

**ItemDataProc 関数は** DLL **拡張部分によって定義された関数であり，ユーザ** DLL **が作成した項目が** HSM77016 **によって読み出されるか書き込まれるときに** HSM77016 **によって呼び出されます。ユーザ** DLL **が作成した項目を読み出すための ItemDataProc 関数は，**GREATEITEM **構造体の** lpfctReadItem **メンバの CreateItem 関数に移行します。また，ユーザ** DLL **作成の項目を書き込むための関数は，**CREATEITEM **構造体の** lpfctWriteItem **メンバである** CreateItem **関数に移行します。ItemDataProc 名は，ユーザ** DLL **が供給する関数名のプレース・ホルダです。実際の名前はエクスポートする必要があります。**

- **構文**

```
int ItemDataProc(
    HITEM hItem,              //アイテム・ハンドル
    LPITEMDATA lpItemData,    //
    PVOID pData               // ユーザ・データ
);
```

- **パラメータ**

| | |
|---|---|
| *hItem* | **読み出されるまたは書き込まれる項目の項目ハンドル。項目ハンドルは CreateItem 関数によって戻されます。** |
| *lpItemData* | **項目データを受信または供給する** ITEMDATA **構造体を指します。** |
| *pData* | **ユーザから供給されるデータ。このデータは項目作成時に** CREATEITEM **構造体の** pData **メンバで設定されます。** |

- **戻り値**

**関数が正しく実行されたときには** 0 **を，エラーが発生したときには** 0 **以外の値を戻します。**

(7) MakeSnapshotMemoryMappedArea

**MakeSnapshotMemoryMappedArea 関数が呼び出されるのは，**HSM77016 **がユーザ** DLL **に対して，**SIM **ファイルへの永久格納またはバックトレース・バッファへの一時格納のいずれかについて内部状態をスナップショットするように要求するときです。**

- **構文**

```
int MakeSnapshotMemoryMappedArea(
    HANDLE hndlArea,                  // メモリ・マップ領域のハンドル
    LPMEMMAPSNAPSHOT lpSnapShot       // スナップショット・バッファ・
                                      // データを持つ構造体への
                                      // ポインタ
);
```


• パラメータ

| | |
|---|---|
| *hndlArea* | スナップショットするメモリ・マップ領域のハンドル。ハンドルは，CreatMemoryMappedArea 関数から戻されたハンドルです。 |
| *lpSnapShot* | スナップショット動作およびスナップショット動作に使用するバッファ・メモリに関する情報を含んでいる MEMMAPSNAPSHOT 構造体へのポインタ。 |

• 戻り値

*lpSnapShot* 引数で指定したバッファがスナップショット・バッファに使用できるほどの容量を持っていて，かつ関数がその内部状態をスナップショットすることに成功した場合は，戻り値は *lpSnapShot* 引数で指定したバッファのスナップショット動作に使用するバイト数になります。

*lpSnapShot* 引数で指定したバッファがスナップショット動作に使用できるほどの容量を持っていない場合，戻り値はスナップショットの動作に必要なバイト数（負数）です。

スナップショット動作をサポートしていない関数では，戻り値は 0 となります。

（8）ModifyReqProc

**ModifyREqProc 関数は，**DLL **拡張部分によって定義された関数であり，ユーザ** DLL **によって作成された項目が ItemDataModified コールを生成すべきかどうかを示すために** HSM77016 **に呼び出されます。**

**ModifyREqProc 関数はオプションであり，使用されるときは，**CREATEITEM **構造体の** lpfctModifyReq **メンバの CreateItem に移行する必要があります。ModifyREqProc 名はユーザ** DLL **が供給する関数名のプレース・ホルダです。実際の名前はエクスポートする必要があります。**

• 構文

```
int ModifyReqProc(
  HITEM hItem,          // 項目ハンドル
  BOOL bGenerate,       //
  PVOID pData           // ユーザ・データ
);
```

• パラメータ

| | |
|---|---|
| *hItem* | 読み出されるかあるいは書き込まれる項目の項目ハンドル。項目ハンドルは，CreateItem 関数によって戻されます。 |
| *bGenerate* | ユーザ DLL が ItemDataModified コールを生成する場合には真となり，ItemDataModified コールを生成する必要のない場合には偽となります。 |
| *pData* | ユーザに供給されるデータ。このデータは，項目を作成時に CREATEITEM 構造体の pData データで設定されます。 |

• 戻り値

この関数が正しく実行されたときには 0 を，エラーが発生したときには 0 以外の値を戻します。

保守／廃止

（9）ReadWriteMemoryMappedElement

**ReadWriteMemoryMappedElement 関数が呼び出されるのは，ユーザ** DLL **のメモリ・マップ領域へのシミュレーションか，ユーザ・インタフェース読み出しまたは書き込みアクセスが生じたときです。**

**• 構文**

```
int ReadWriteMemoryMappedElement(
    HANDLE hndlArea,              // メモリ・マップ領域のハンドル
    LPMEMMAPRW lpRwStruct         // リード／ライト・データを持つ
                                  // 構造体へのポインタ
);
```

**• パラメータ**

*hndlArea*　**読み出されるメモリ・マップ領域のハンドル。このハンドルは，CreatMemoryMappedArea 関数によって戻されます。**

*lpRwStruct*　**読み出されるかあるいは書き込まれるメモリ・マップ・エレメントに関する情報を含んでいる** MEMMAPRW **構造体へのポインタ。**

MMRWMDE_WRITE **フラグが** MEMMAPRW **構造体の wMode で設定されているとき，データはメモリ・マップ領域に書き込まれます。**

MMRWMDE_UNDEFDATA **フラグが wMode で設定されていない場合，データは** MEMMAPRW **構造体の wMode に含まれます。それ以外では不定データがメモリに書き込まれます。**

MMRWMDE_WRITE **フラグが** MEMMAPRW **構造体の wMode で設定されていないとき，データはメモリ・マップ領域から読み出され，それに応じてこの関数は，wData と** MMRWMDE_UNDEFDATA **フラグを wMode に書き込みます。**

**• 戻り値**

**関数が正しく実行された場合，戻り値は 0 です。**

**関数が正しく実行されなかった場合，戻り値はエラー・コードです。**

（10）RestoreSnapshotMemoryMappedArea

**RestoreSnapshotMemoryMappedArea 関数は，MakeSnapshotMemoryMappedArea 関数と対をなすものです。**HSM77016 **がユーザ** DLL **に対して，その内部状態を MakeSnapshotMemoryMappedArea 関数によって作成されたスナップショットから復帰するとき，RestoreSnapshotMemoryMappedArea 関数が呼び出されます。**

**• 構文**

```
int RestoreSnapshotMemoryMappedArea(
    HANDLE hndlArea,                  // メモリ・マップ領域のハンドル
    LPMEMMAPSNAPSHOT lpSnapShot       // スナップショット・バッファ・データを
                                      // 持つ構造体へのポインタ
);
```

保守／廃止

• パラメータ

| | |
|---|---|
| *hndlArea* | スナップショットから復元される，メモリ・マップ領域のハンドル。このハンドルは CreatMemoryMappedArea 関数によって戻されます。 |
| *lpSnapShot* | スナップショット動作およびスナップショット動作に使用するバッファ・メモリに関する情報を含んでいる MEMMAPSNAPSHOT 構造体へのポインタ。 |

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（11）UpdateCommandProc

**UpdateCommandProc 関数は，**DLL **の定義済みコールバック関数であり，**InstallUIExtension **関数で定義されたカスタム・メニュー拡張部分の** initmenu **メッセージ（**WM_INITMENU**）で呼び出されます。これにより** DLL **はメニューを，表示される前に全体を変えることなく修正することができます。**

UpdateCommandProc **関数は各メニュー項目に対して呼び出されます。**

• 構文

```
int UpdateCommandProc(
  UINT nIDOrg,
  UINT nIDTrans,
  UINT nIndex,
  HMENU hMenu,
  HMENU hSubMenu
);
```

• パラメータ

| | |
|---|---|
| *nIDOrg* | DLL リソースで規定されているメニュー項目，コントロール，またはアクセラレータのオリジナルの識別子を指定します。 |
| *nIDTrans* | メニュー項目，コントロール，またはアクセラレータの変換後の識別子を指定します。 |
| *nIndex* | メニュー項目のインデクスを識別します。 |
| *hMenu* | 初期化されるメニューを識別します。 |
| *hSubMenu* | ドロップダウン・メニューまたはサブメニューを識別します。 |

• 戻り値

コマンド・プロシージャは，メッセージを処理する場合には 0 を戻し，処理しない場合には 0 以外の値を戻します。

保守／廃止

## 9.4.2　HSM77016 によってエクスポートされる関数

（1）CreateItem

**CreateItem 関数は，**HSM77016 **環境で使用される新規項目を作成します。この関数に渡される** CREATEITEM **構造体での設定に基づいて，新規項目は，レジスタ，端子，ポート，または変数エレメントとして登録されます。構造体はさらに，項目データが読み出されるかあるいは書き込まれる場合に** HSM77016 **によって呼び出される** 2 **つのコールバック関数をそれぞれ指定します。**

**• 構文**

```
HITEM CreateItem(
    LPCREATEITEM lpCreateItemStruct      // 項目作成構造体への
                                         //ポインタ
);
```

**• パラメータ**

*lpCreateItemStruct*　**項目作成構造体へのポインタ。**

**• 戻り値**

**関数が正しく実行された場合，戻り値は新規項目のハンドルです。項目のデータは，GetItem および SetItem 関数で読み出したり，書き込んだりすることができます。項目が必要でなくなった場合，DestroyItem 関数でハンドルを破棄する必要があります。**

**関数が正しく実行されなかった場合，戻り値は** NULL **です。**

（2）CreateExecEventCallback

**CreateExecEventCallback 関数は，指定のインストラクション・メモリ・ロケーションが実行される場合にユーザ供給のコールバック関数を呼び出すイベントを作成します。**

**• 構文**

```
HEVENT CreateExecEventCallback(
    DWORD dwMemAddr,          // インストラクション・メモリ・アドレス
    EVENTCALLBACKPROC pfct,   // コールバック関数
    PVOID pData               // ユーザ・データ
);
```

**• パラメータ**

| | |
|---|---|
| *dwMemAddr* | **監視されるインストラクション・メモリ・ロケーションのアドレス。プログラム実行がこのロケーションに達すると，***pfct* **によって供給されたユーザ関数は，このロケーションの命令が実行される前に呼び出されます。** |
| *pfct* | **この関数によって作成されたイベントで呼び出されるコールバック関数を指します。コールバック関数についての詳細は，**9.4.1（5）EventProc **を参照してください。** |
| *pData* | **ユーザに供給されるデータ。このデータはコールバック関数に渡されます。それ以外で** HSM77016 **によって使用されることはありません。このパラメータは，コールバック関数に追加情報を渡すのに使用されます。** |

保守／廃止

• 戻り値

関数が正しく実行された場合，戻り値は新たに作成したオブジェクトのハンドルです。
関数が正しく実行されなかった場合，戻り値は NULL です。

（3）CreateItemEventCallback

CreateItemEventCallback 関数は，指定した項目が変更されたときにユーザ供給のコールバック関数を呼び出すイベントを作成します。

• 構文

```
HEVENT CreateItemEventCallback(
  HITEM hItem,                  // イベント項目ハンドラ
  EVENTCALLBACKPROC pfct,       // コールバック関数
  PVOID pData                   // ユーザ・データ
);
```

• パラメータ

*hItem*　この関数によって作成されたイベントで使用される項目のハンドル。hItem で指定した項目が変更されると，pfct から供給されたユーザ関数が呼び出されます。項目ハンドル hItem は，CreateItem または FindItem 関数によって戻されたハンドルでなければなりません。

*pfct*　この関数によって作成されたイベントで呼び出されるコールバック関数を指します。コールバック関数についての詳細は，9.4.1（5）EventProc 関数を参照してください。

*pData*　ユーザから供給されるデータ。このデータはコールバック関数に渡されます。そうでない場合は，HSM77016 によって使用されることはありません。このパラメータは，コールバック関数に追加情報を渡すのに使用されます。

• 戻り値

関数が正しく実行された場合，戻り値は新規に作成したオブジェクトのハンドルです。
関数が正しく実行されなかった場合，戻り値は NULL です。

（4）CreateMemRdEventCallback

CreateMemRdEventCallback 関数は，μPD77016 プログラムを実行することによって，指定のメモリ範囲でのメモリ読み出しアクセスが発生したときに，ユーザ供給のコールバック関数を呼び出すイベントを作成します。

• 構文

```
HEVENT CreateMemRdEventCallback(
  DWORD dwMemAddr,              // イベント・メモリ・ブロック開始アドレス
  DWORD dwMemSize,              // イベント・メモリ・ブロック・サイズ
  EVENTCALLBACKPROC pfct,       // コールバック関数
  PVOID pData                   // ユーザ・データ
);
```

保守／廃止

• パラメータ

*dwMemAddr*　この関数によって作成されたイベントで使用するメモリ・ブロックの開始アドレス。メモリ・アドレスはワードで表されます。X データ・メモリ・アドレスの範囲は 0x00000-0x0FFFF，Y データ，メモリ・アドレスの範囲は 0x10000-0x1FFFF です。

*dwMemSize*　この関数によって作成されたイベントで使用するメモリ・ブロックのサイズ。μPD77016 プログラムを実行することによって dwMemAddr と wMemSize が指定したメモリ･ブロックでメモリ読み出しアクセスが発生すると，pfct から供給されたユーザ関数が呼び出されます。項目ハンドル hItem は，CreateItem または FindItem 関数によって戻されたハンドルでなければなりません。

*pfct*　この関数によって作成されたイベントに呼び出されるコールバック関数を指します。コールバック関数についての詳細は，9.4.1（5）EventProc を参照してください。

*pData*　ユーザから供給されるデータ。このデータは，コールバック関数に渡されます。それ以外で HSM77016 によって使用されることはありません。このパラメータは，コールバック関数に追加情報を渡すのに使用されます。

• 戻り値

関数が正しく実行された場合，戻り値は新たに作成したオブジェクトのハンドルです。
関数が正しく実行されなかった場合，戻り値は NULL です。

• 備考

*dwMemAddr* と *dwMemSize* が指定したメモリ・ブロックは，X または Y データ・メモリ空間内のいずれかに収まっていなければなりません。X データ・メモリで始まり，Y データ・メモリで終わるメモリ・ブロック（たとえば，dwMemAddr = 0x00000 と dwMemSize = 0x20000）を指定することはできません。

（5）CreateMemWrEventCallback

**CreateMemWrEventCallback 関数は，**μPD77016 **プログラムを実行することによって指定のメモリ範囲でメモリ書き込みアクセスが発生したときに，ユーザ供給のコールバック関数を呼び出すイベントを作成します。**

• 構文

```
HEVENT CreateMemWrEventCallback(
    DWORD dwMemAddr,          // イベント・メモリ・ブロック開始アドレス
    DWORD dwMemSize,          // イベント・メモリ・ブロック・サイズ
    EVENTCALLBACKPROC pfct,   // コールバック関数
    PVOID pData               // ユーザ・データ
);
```

保守／廃止

• パラメータ

*dwMemAddr*　この関数によって作成されたイベントで使用されるメモリ・ブロックの開始アドレス。メモリ・アドレスはワードで表されます。X データ・メモリ・アドレスの範囲は 0x00000-0x0FFFF，Y データ・メモリ・アドレスの範囲は 0x10000-0x1FFFF です。

*dwMemSize*　この関数によって作成されたイベントで使用されるメモリ・ブロックのサイズ。μPD77016 プログラムを実行することによって，dwMemAddr と dwMemSize 指定のメモリ・ブロックでメモリ書き込みアクセスが発生すると，pfct によって指定されたユーザ関数が呼び出されます。項目ハンドル hItem は，CreateItem または FindItem 関数が戻したハンドルでなければなりません。

*pfct*　この関数によって作成されたイベントによって呼び出されるコールバック関数を指します。コールバック関数についての詳細は，9.4.1（5）EventProc を参照してください。

*pData*　ユーザから供給されるデータ。このデータは，コールバック関数に渡されます。それ以外で HSM77016 によって使用されることはありません。このパラメータは，コールバック関数に追加情報を渡すのに使用できます。

• 戻り値

関数が正しく実行された場合，戻り値は新たに作成したオブジェクトのハンドルです。
関数が正しく実行されなかった場合，戻り値は NULL です。

• 備考

*dwMemAddr* と *dwMemSize* によって指定されたメモリ・ブロックは，X または Y データ・メモリ空間内のいずれかに収まっていなければなりません。X データ・メモリで始まり，Y データ・メモリで終わるメモリ・ブロック（たとえば，dwMemAddr = 0x00000 と dwMemSize = 0x20000）を指定することはできません。

（6）CreateStepEventCallback

CreateStepEventCallback 関数は，指定のシミュレーション・ステップ数が経過したときにユーザ供給のコールバック関数を呼び出すイベントを作成します。

• 構文

```
HEVENT CreateStepEventCallback(
  DWORD dwStep,                 // イベント・ステップ数
  EVENTCALLBACKPROC pfct,       // コールバック関数
  PVOID pData                   // ユーザ・データ
);
```

• パラメータ

*dwStep*　この関数によって作成されたイベントで使用されるシミュレーション・ステップの数。dwStep で指定されたシミュレーション・ステップが実行されたときに，pfct から供給されたユーザ関数が呼び出されます。

*pfct*　この関数によって作成されたイベントによって呼び出されるコールバック関数を指します。コールバック関数についての詳細は，9.4.1（5）EventProc を参照してください。

保守／廃止

*pData*　　ユーザから供給されるデータ。このデータはコールバック関数に渡されます。それ以外で HSM77016 によって使用されることはありません。このパラメータは，コールバック関数に追加情報を渡すのに使用できます。

• 戻り値

関数が正しく実行された場合，戻り値は新たに作成したオブジェクトのハンドルです。
関数が正しく実行されなかった場合，戻り値は NULL です。

（7）CreateTimeEventCallback

**CreateTimeEventCallback 関数は，指定のシミュレーション時間が経過したときに，ユーザ供給のコールバック関数を呼び出すイベントを作成します。**

• 構文

```
HEVENT CreateTimeEventCallback(
  DWORD dwTime,                 // ナノ秒単位のイベント時間
  EVENTCALLBACKPROC pfct,       // コールバック関数
  PVOID pData                   // ユーザ・データ
);
```

• パラメータ

*dwTime*　　この関数によって作成されたイベントで使用されるシミュレーション時間遅延。dwTime で指定したシミュレーション時間が経過したときに，pfct で供給されたユーザ関数が呼び出されます。時間はナノ秒で指定します。

*pfct*　　この関数によって作成されたイベントによって呼び出されるコールバック関数を指します。 コールバック関数についての詳細は，9.4.1（5）EventProc を参照してください。

*pData*　　ユーザから供給されるデータ。このデータはコールバック関数に渡されます。それ以外で HSM77016 によって使用されることはありません。このパラメータは，コールバック関数に追加の情報を渡すのに使用されます。

• 戻り値

関数が正しく実行された場合，戻り値は新たに作成したオブジェクトのハンドルです。
関数が正しく実行されなかった場合，戻り値は NULL です。

（8）DestroyEventCallback

**DestroyEventCallback 関数は，Create*Xxx*EventCallback 関数によって作成されたイベントを破棄します。いったんイベントが破棄されると，そのハンドルはもはや有効ではなく，ユーザ供給のコールバック関数は呼び出されることはありません。**

• 構文

```
Int DestroyEventCallback(
  HEVENT hndl                   // イベント・ハンドル
);
```


• パラメータ

*hndl*　　破棄されるイベントのハンドル。ハンドルは，このイベントを作成した Create*Xxx*EventCallback 関数によって戻されます。

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。
関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（9）DestroyItem

DestroyItem 関数は，CreateItem によって戻されたユーザ定義の項目ハンドルを破棄します。CreateItem 関数によって戻された項目ハンドラは，すべて DestroyItem 関数で破棄する必要があります。

• 構文

```
int DestroyItem(
  HITEM hItem              // 項目ハンドル
);
```

• パラメータ

*hItem*　　破棄される項目のハンドル

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。
関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（10）EnableEventCallback

EnableEventCallback 関数は，Create*Xxx*EventCallback 関数が作成したイベントをイネーブルまたはディスエーブルします。ディスエーブルされたイベントは対応するコールバック・プロシージャを呼び出すことはありません。

• 構文

```
int EnableEventCallback(
  HEVENT hndl,             // 項目ハンドル
  BOOL bEnable             // フラグをイネーブル／ディスエーブルする
);
```

• パラメータ

*hndl*　　イネーブルまたはディスエーブルされる項目のハンドル

*bEnable*　　イベントをイネーブルするかあるいはディスエーブルするかを指定します。このパラメータが TRUE の場合，イベントはイネーブルされます。パラメータが FALSE の場合，イベントはディスエーブルされます。

保守／廃止

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（11）FindItem

FindItem 関数は，式ストリングからの項目のハンドルを検索します。式ストリングは，HSM77016 の Watch ウインドウで使用できればどのストリングでもかまいません。

• 構文

```
HITEM FindItem(
    LPCTSTR lpszName        // 式ストリングでのポインタ
);
```

• パラメータ

*lpszName*　　式ストリングへのポインタ

• 戻り値

関数が正しく実行された場合，戻り値は項目のハンドルです。項目のデータは，GetItem および SetItem 関数で読み出しと書き込みが可能です。項目が必要でなくなった場合，そのハンドルは ReleaseItem 関数で破棄しなければなりません。

関数が正しく実行されなかった場合，戻り値は NULL です。

• 備考

式が正しいかぎり，その式の有効なハンドルを戻します。式が算術演算を含んでいる場合，GetItem が戻した値は演算の結果です。SetItem は通常，式が算術演算を含んでいる場合には失敗します。

式が未知のシンボルを含んでいても FindItem 関数は正しく実行され，式の有効なハンドルを戻します。これらのシンボルは，あとのポイントで，GetItem 関数の使用に先立って定義することができます。正しく実行されなかった場合，この関数はエラー・コードを戻します。

（12）GetBuildVersion

GetBuildVersion 関数は，ユーザ DLL をロードした Atair DSP のビルド・バージョンを戻します。

• 構文

```
DWORD GetBuildVersion();
```

• 戻り値

戻り値は，上位ワードの上位バイトにあるメージャ・バージョン，上位ワードの下位バイトにあるマイナ・バージョン，および下位ワードのビルド・ステップを含んでいます。たとえばリリース 2.1 では，メージャ・バージョン 2，マイナ・バージョン 1，ビルド・ステップ 5 を示す 0x02010005 を戻します。

保守／廃止

（13）GetExecMode

**GetExecMode 関数は，**HSM77016 **の現在のシミュレーション状態を戻します。**

- **構文**

**DWORD GetExecMode();**

- **戻り値**

**戻り値は次のフラグ群の** 1 **つです。**

**シミュレーションが現在停止している場合，次のフラグ群の** 1 **つが戻されます。**

**EXECMODE_IDLE**

**シミュレーションがまだ開始されていません。**

**EXECMODE_FAILURE**

**このフラグは，**HSM77016 **から戻されていません。**

**EXECMODE_STOPPED**

**シミュレーションが停止しています。**

**EXECMODE_BREAK**

**シミュレーションが，ブレークポイントまたはマニュアル・ブレーク・コマンドで停止しています。シミュレーションが現在実行されている場合，次のフラグ群の** 1 **つが戻されます。**

**EXECMODE_WAITINGFORRESET**

**このフラグは，**HSM77016 **から戻されません。**

**EXECMODE_BOOTING**

**このフラグは，**HSM77016 **から戻されません。**

**EXECMODE_RUNNING**

**シミュレーションが実行中です。**

**EXECMODE_HALT**

**シミュレーションが実行中であり，**HALT **命令が現在処理されています。シミュレーションは** HALT **モードを解除する割り込みを待っています。**

**EXECMODE_STOP**

**シミュレーションが実行中であり，**STOP **命令が現在処理されています。シミュレーションは** STOP **モードを解除するリセット信号を待っています。**

（14）GetExecParam

**GetExecParam 関数は，現在ロードされているモデル，プロセッサのタイプ，およびクロック周波数などの現在実行しているシミュレーションの構成に関する情報を戻します。**

- **構文**

```
int GetExecParam(
    LPEXECSTATE lpExecState        // 実行しているシミュレーションの構成に
                                   // 関する情報を受けとるためのポインタ
);
```

保守／廃止

• パラメータ

*lpExecState*　　実行構成状態構造体へのポインタ。

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（15）GetItem

GetItem 関数は，HSM77016 内の 1 項目に関する現在の値および状態を戻します。

CreateItem 関数か FindItem 関数によって戻された項目であれば，どの項目でも使用できます。

• 構文

```
int GetItem(
  HITEM hItem,                  // 項目ハンドル
  LPITEMDATA pItemData,         // 項目データ構造体へのポインタ
);
```

• パラメータ

*hItem*　　検索の対象となる値を持つ項目のハンドル。

*pItemData*　　項目データ構造体へのポインタ。ITEMDATA 構造体の dwFlags のフラグは，どのデータが検索されるのかを示します。次に示すフラグを任意に組み合わせて使用できます。

ITEMDATA_VALUE

liValue および dwItemState の項目に関する現在の値と状態を戻します。dwItemState は，ビット 0-15，16-31，32-47，および 48-63 の不定データをそれぞれ示すフラグ ITEMSTATE_UNDEF_L，ITEMSTATE_UNDEF_H，ITEMSTATE_UNDEF_E，および ITEMSTATE_UNDEF_X の任意の組み合わせを含んでいます。また，項目がハイ・インピーダンス状態をサポートしている場合には，フラグ ITEMSTATE_HIZ も含まれます。

ITEMDATA_BITS

dwNumberOfBits の項目の有効なデータ・ビットの数を戻します。

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

• 備考

GetItemProperty 関数は，項目がハイ・インピーダンス状態をサポートしているかどうかを調べるのに使用できます。

保守／廃止

（16）GetItemProperty

**GetItemProperty 関数は，FindItem 関数が検索した項目，あるいは CreateItem 関数が作成した項目に関する情報を戻します。**

**• 構文**

```
int GetItemProperty(
    HITEM hItem,                          // 項目ハンドル
    LPITEMPROPERTY lpItemProperty         // points to an
                                          // ITEMPROPERTY 構造体への
                                          // ポインタ
);
```

**• パラメータ**

*hItem*　**照会の対象となる項目のハンドル。項目ハンドルは，FindItem または CreateItem 関数によって戻されます。**

*lpItemProperty*　**項目情報を検索する** ITEMPROPERTY **を指します。dwMask フィールドは，どの情報が検索されるのかを示します。**

**• 戻り値**

**関数が正しく実行され，かつ** ITEMPROPERTY **構造体が dwMask の** ITEMPROPMASK_NAME **フラグをセットしていて dwBufferLength が** 0 **の場合，戻り値は項目名を保持するのに必要なバッファ・サイズです。**

**関数が正しく実行され，かつ** ITEMPROPERTY **構造体が dwMask の** ITEMPROPMASK_NAME **フラグをセットしていて dwBufferLength が** 0 **でない場合，戻り値は** 0 **となります。**

**関数が正しく実行されなかった場合，戻り値はエラー・コードです。**

（17）GetMemory

GetMemory **関数は，シミュレーション・メモリのブロックを読み出します。この関数は，データ・メモリとインストラクション・メモリの両方のブロックを読み出すことができます。**

**• 構文**

```
int GetMemory(
    LPMEMDATA lpMemData          //  MEMDATA 構造体への
                                 //  ポインタ
);
```

**• パラメータ**

*lpMemData*　**読み出されるメモリ・ブロックを指定している** MEMDATA **構造体を指します。pData メンバは，メモリ・セル・データを検索するためにバッファを指さなければなりません。**

保守／廃止

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。エラーがメモリ・アクセス動作によって引き起こされた場合，失敗したアクセスに対応する MEMCELLDATA エレメントの dwState フィールドが，MEMCELLDATA_ERROR フラグを設定することによってそのエラーを示します。

（18）GetTool

GetTool 関数は，どの Atair DSP ツールがユーザ DLL をロードしたのかを戻します。

• 構文

```
DWORD GetTool();
```

• 戻り値

戻り値は，TOOL_HSM77016 です。

（19）InstallUIExtension

InstallUIExtension 関数は，HSM77016 環境で使用するユーザ・インタフェース拡張項目をインストールします。この関数に渡される dwType と lpDescriptor の設定に基づいて，新たな拡張項目がウインドウ拡張部分またはメニュー拡張部分として登録されます。

• 構文

```
HEXTENSION InstallUIExtension(
DWORD dwType,           // 拡張対象の種類
LPVOID lpDescriptor     // 記述構造体へのポインタ
);
```

• パラメータ

*dwType*　　*dwType* ユーザ・インタフェース拡張部分の種類を指定します。このパラメータは，次の値のうちの 1 つです。

| 値 | 意　味 |
|---|---|
| UIEXTTYP_WINDOW | lpDescriptor は，ウインドウ拡張部分をインストールする WINDOWDESC 構造体へのポインタです。 |
| UIEXTTYP_HELPCTX | lpDescriptor は，CREATEITEM 構造体で使用するコンテキスト・ヘルプ識別子を得るための HELPCTXDESC 構造体へのポインタです。戻されたハンドルは，UninstallUIExtension 関数で破棄してはなりません。 |
| UIEXTTYP_MENU | lpDescriptor は，メニュー拡張部分をインストールするための MENUDESC へのポインタです。 |

*lpDescriptor*　　拡張部分構造体へのポインタ。

• 戻り値

関数が正しく実行された場合，戻り値はユーザ・インタフェース拡張部分のハンドルです。この拡張部分がもはや必要でない場合，そのハンドルは UninstallUIExtension 関数で破棄しなければなりません。
関数が正しく実行されなかった場合，戻り値は NULL です。

（20）ItemDataModified

ItemDataModified 関数は，CreateItem 関数によって作成された項目が変更するときに呼び出す必要があります。

• 構文

```
int ItemDataModified(
  HITEM hItem              // 項目ハンドル
);
```

• パラメータ

| | |
|---|---|
| *hItem* | 変更された項目のハンドル。項目ハンドルは，CreateItem 関数から戻されます。 |

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。
関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（21）MemoryDataModified

MemoryDataModified 関数は，ユーザ DLL のメモリ・マップ・エレメントが変更するときに呼び出す必要があります。

• 構文

```
int MemoryDataModified(
  DWORD dwAddress          // メモリ・マップ・エレメントのアドレス
);
```

• パラメータ

| | |
|---|---|
| *dwAddress* | 変更になったアドレスまたはメモリ・マップ・エレメント。アドレス 0x00000-0x0FFFF は X データ・メモリを，アドレス 0x10000-0x1FFFF は Y データ・メモリを指定します。 |

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。
関数が正しく実行されなかった場合，戻り値はエラー・コードです。

• 備考

大きなメモリ・ブロックが変更になった場合，すべての X データ・メモリが変更になったことを示すのに dwAddress を MDMF_ALLX に，すべての Y データ・メモリが変更になったことを示すのに MDMF_ALLY に設定します。しかし，これらのフラグの使用については，HSM77016 がすべてのメモリ・データ，さらにはユーザ DLL に含まれていないデータさえも再度読み出す原因となるので，注意してください。

保守／廃止

（22）OutputLogEntry

OutputLogEntry 関数はログ・エントリを作成します。このログ・エントリは HSM77016 の Log ウインドウで表示できます。

• 構文

```
HITEM OutputLogEntry(
  LPCTSTR lpszEventText,        // ログ・エントリ・メッセージ・テキスト
  DWORD dwFlags                 // ログ・エントリ・フラグ
);
```

• パラメータ

*lpszEventText*　ログ・エントリ・テキスト・ストリングへのポインタ。

*dwFlags*　ログ・エントリ・クラスを示しているログ・エントリ・フラグ。次のフラグ群のいずれかとすることができます。

**LOGCLASS_ERROR**
　エラー・ログ・エントリ

**LOGCLASS_QUESTION**
　クエスチョン・ログ・エントリ

**LOGCLASS_WARNING**
　警告ログ・エントリ

**LOGCLASS_INFORMATIONAL**
　情報ログ・エントリ

**LOGCLASS_EXECUTE**
　実行ログ・エントリ

**LOGCLASS_DUMP**
　ダンプ・ログ・エントリ

**LOGCLASS_DUMP_SUPL**
　補足ダンプ・ログ・エントリ。このクラスのエントリは，LOGCLASS_DUMP スタイルを持つ別のエントリに続く必要があり，したがって Log ウインドウ内にアイコンまたはタイム・スタンプ・フィールドを持ちません。

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

（23）OutputMessageBox

OutputMessageBox 関数は，エラー，警告，および情報メッセージの出力に使用されます。現在のシミュレーション状態によっては，メッセージはユーザに与えられるか，イベント・ログに記録されるか，あるいはすべてスキップされます。

保守／廃止

• 構文

```
int OutputMessageBox(
    LPCTSTR lpszMessageTemplate,        // メッセージ・テキスト・テンプレート
    ...                                 // オプション引数
);
```

• パラメータ

*lpszMessageTemplate*　メッセージ・テキスト・テンプレートへのポインタ。使用可能なフォーマット・オプションについては，**備考**を参照してください。

• 戻り値

関数が正しく実行された場合，戻り値は次の値のいずれかです。

| 値 | 意　味 |
|---|---|
| IDABORT | Abort ボタンが選択された。 |
| IDCANCEL | Cancel ボタンが選択された。 |
| IDIGNORE | Ignore ボタンが選択された。 |
| IDNO | No ボタンが選択された。 |
| IDOK | OK ボタンが選択された。 |
| IDRETRY | Retry ボタンが選択された。 |
| IDYES | Yes ボタンが選択された。 |

関数が正しく実行されなかった場合，戻り値は 0 です。

• 備考

*lpszMessageTemplate* によって示されるストリングは次のフォーマットを持っています。

[*type*][**$**]*message text*

*type*　Windows の **MessageBox** 関数に渡されるタイプを示す 10 進コード化数字。使用可能なオプションとそれらの数値については，Windows **MessageBox** 関数に関するマニュアルと WINUSER.H の宣言を参照してください。

**$**　$がある場合，*message text* で与えられたストリングは Windows FormatMessage 関数を使用してフォーマットされます。オプション引数は，Windows MessageBox 関数に渡されます。

$がない場合は，*message text* で与えられたストリングは標準 C printf 関数を使用してフォーマットされます。オプション引数は，printf 関数に渡されます。Windows wsprintf 関数と対照的に，使用された printf 関数は浮動小数点フォーマットをサポートしています。

*message text*　lpSource 引数ストリングとして FormatMessage に，またはフォーマット引数ストリングとして printf に渡されるメッセージ・テキスト。使用可能なフォーマット・オプションについては，それぞれの関数についての説明を参照してください。

変換後の最終ストリングは，350 文字を越えることはできません。

保守／廃止

（24）ReleaseItem

**ReleaseItem 関数は，FindItem 関数によって戻された項目ハンドルを破棄します。FindItem 関数によって戻された各項目ハンドルは，すべて ReleaseItem 関数で破棄する必要があります。**

**• 構文**

```
int FindItem(
    HITEM hItem                 // 項目ハンドル
);
```

**• パラメータ**

| | |
|---|---|
| *hItem* | **リリースされる項目のハンドル** |

**• 戻り値**

**関数が正しく実行された場合，戻り値は** 0 **です。**

**関数が正しく実行されなかった場合，戻り値はエラー・コードです。**

（25）SetItem

**SetItem 関数は，**HSM77016 **の項目の現在の値と状態を設定します。CreateItem または FindItem 関数によって戻されたすべての項目に有効です。**

**• 構文**

```
int SetItem(
    HITEM hItem,                // 項目ハンドル
    LPITEMDATA pItemData        // 項目データ構造体へのポインタ
);
```

**• パラメータ**

| | |
|---|---|
| *hItem* | Handle of the item whose value is to be set. |
| *pItemData* | **項目データ構造体へのポインタ。**ITEMDATA **構造体の dwFlags のフラグは，どのデータが設定されているかを示します。ITEMDATA_VALUE フラグを設定する必要があります。項目内に設定される値と状態は，liValue と dwItemState でそれぞれ示されます。dwItemState は，ビット** 0-15，16-31，32-47，**および** 48-63 **の不定データをそれぞれ示すフラグ ITEMSTATE_UNDEF_L，ITEMSTATE_UNDEF_H，ITEMSTATE_UNDEF_E，および** ITEMSTATE_UNDEF_X **の任意の組み合わせを含んでいます。また，項目がハイ・インピーダンス状態をサポートしている場合には，フラグ ITEMSTATE_HIZ も含まれます。ITEMDATA_BITS が dwFlags で設定されていない場合，liValue のデータは項目の有効ビット数まで切り捨てられたあと，項目に割り当てられます。ITEMDATA_BITS が dwFlags で設定されている場合，liValue のデータは dwNumberOfBits で示されるビット数に切り捨てられ，次に項目の有効ビット数まで符号拡張されたあと，項目に割り当てられます。** |

保守／廃止

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

• 備考

GetItemProperty 関数は，項目がハイ・インピーダンス状態をサポートしているかどうかを調べるのに使用できます。ITEMSTATE_HIZ が設定されていて，かつ項目がハイ・インピーダンス状態をサポートしていない場合，SetItem 関数は正しく実行されません。

（26）SetMemory

SetMemory 関数は，シミュレーション・メモリのブロックを書き込みます。この関数は，データ・メモリとインストラクション・メモリのブロックを書き込むのに使用できます。

• 構文

```
int SetMemory(
    LPMEMDATA lpMemData          // MEMDATA 構造体を指す
);
```

• パラメータ

*lpMemData*　書き込まれるメモリ・ブロックを指定する MEMDATA 構造体を指します。pData メンバは，メモリ・セル・データを含んでいるバッファを指す必要があります。

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。エラーがメモリ・アクセス動作によって引き起こされた場合，そのエラーは MEMCELLSTATE_ERROR フラグを設定することによって，失敗したアクセス動作に対応する MEMCELLDATA エレメントの dwState フィールドで示されます。

（27）UninstallUIExtension

UninstallUIExtension 関数は，InstallUIExtension 関数によって戻された定義済み拡張部分ハンドルを破棄します。InstallUIExtension 関数によって戻されたすべての拡張部分ハンドル（コンテキスト・ヘルプ・ハンドルは除く）は，UninstallUIExtension 関数によって破棄する必要があります。

• 構文

```
int UninstallUIExtension(
    HEXTENSION hExtension        // 拡張部分ハンドル
);
```

• パラメータ

*hExtension*　破棄される拡張部分項目のハンドル

保守／廃止

• 戻り値

関数が正しく実行された場合，戻り値は 0 です。

関数が正しく実行されなかった場合，戻り値はエラー・コードです。

## 9.4.3 データ構造体

（1）CREATEITEM

**CREATEITEM 構造体は，**HSM77016 **環境におけるユーザ定義の項目を作成するのに必要な情報を含んでいます。**

• 構文

```
typedef struct tagCREATEITEM {    /* ci */
        DWORD dwSize;
        LPCSTR lpszName;
        LPCSTR lpszDescription;
        WORD wClass;
        WORD wBitSize;
        ITEMDATAPROC lpfctReadItem;
        ITEMDATAPROC lpfctWriteItem;
        MODIFYREQPROC lpfctModifyReq;
        HHELPID hHelpId;
        PVOID pData;
        DWORD dwReserved;
} CREATEITEM, * LPCREATEITEM;
```

• メンバ

**dwSize**　メモリにある CREATEITEM 構造体のサイズを示します。

**lpszName**　新たな項目の識別子を含んでいます。このストリングは，接頭語 REG，PIN，または PORT を含んでいません。wClass で与えられる項目クラスに基づいて，それらの接頭語は自動的に適用されます。たとえば，REG XYZ を介してアクセスできるレジスタを作成したときには “ XYZ ” が lpszName に，“ CICLS_REG ” が wClass に指定されます。

**lpszDescription**　記号テーブル・ダイアログで表示される項目の説明的コメントを含んでいます。このメンバは，説明を表示させたくない場合には NULL とすることができます。

**wClass**　新たな項目のクラスを含んでいます。これは次の定数のいずれかである必要があります。

**CICLS_GENERIC**
　項目は総称項目です。この項目にアクセスする場合は，接頭語が必要ありません。

**CICLS_REG**
　項目はレジスタです。この項目にアクセスする場合は，接頭語 REG が必要です。

**CICLS_PIN**
　項目は端子です。この項目にアクセスする場合は，接頭語 PIN が必要です。

保守／廃止

CICLS_PORT
　項目はポートです。この項目にアクセスする場合は，接頭語 PORT が必要です。

**wBitSize**　項目の有効なビット数を含んでいます。この値は主に，表示のフォーマットに使用します。

**lpfctReadItem**　項目の値が読み出される場合に HSM77016 によって呼び出されるコールバック関数を含んでいます。

**lpfctWriteItem**　項目の値が書き込まれる場合に HSM77016 によって呼び出されるコールバック関数を含んでいます。このメンバは，読み出し専用項目を作成する場合に NULL とすることができます。

**lpfctModifyReq**　項目データの修正通知に関する要件が変更になった場合に HSM77016 に呼び出されるコールバック関数を含んでいます。この関数が使用される場合，HSM77016 はこのコールバック関数を呼び出して，DataItemModified コールが必要かどうかを示します。DataItemModified コールが必要であることを HSM77016 が示した場合，これらのコールは項目の値が変更になるたびに作成されます。DataItemModified コールが必要でないことを HSM77016 が示した場合，ユーザ DLL がそれらのコールをスキップして，パフォーマンスを向上させます。ここで注意していただきたいのは，HSM77016 がその要件をセッション中に何度か変更することがあるということです。このメンバは，当該機能がユーザ DLL によってサポートされていないことを示す場合に NULL とすることができます。この場合，項目の値が変更になるたびに DataItemModified コールを作成する必要があります。

**hHelpId**　この項目に表示されるコンテキスト依存ヘルプ・トピックを記述している InstallUIExtension 関数によって取得されるヘルプ ID を含んでいます。このメンバは，コンテキスト依存ヘルプが使用できないことを示す場合に NULL とすることができます。

**pData**　ユーザ供給のデータを含んでいます。このデータは，項目読み出し，書き込みコールバック関数に渡されますが，それ以外では HSM77016 によって使用されることはありません。

**dwReserved**　0 に設定する必要があります。

### （2）EXECSTATE

EXECSTATE 構造体は，カスタム・メニュー拡張部分に関する情報を含んでいます。

- **構文**

```
typedef struct tagMENUDESC {
        DWORD dwSize;
        DWORD dwFlags;
        LPTSTR lpszModelFile;
        DWORD dwModelFileBufferLength;
        LPTSTR lpszModel;
        DWORD dwModelBufferLength;
        DWORD dwProcFlags;
        DWORD dwCyclePeriod;
}  EXECSTATE, * LPEXECSTATE;
```

保守／廃止

• メンバ

**dwSize**　この構造体のサイズをバイトで示します。

**dwFlags**　どの情報が検索されるのか示します。次のフラグのうち 1 つまたは複数のフラグを設定できます。

**EXECSTATE_MODELFILE**
lpszModeFile によって示されるバッファに入っている現行モデル・ファイル名を戻します。

**EXECSTATE_MODEL**
lpszModel によって示されるバッファに入っている現行モデル名を戻します。

**EXECSTATE_CPUTYPE**
dwProcFlags の現行 CPU タイプを戻します。

**EXECSTATE_CYCLEPERIOD**
dwCyclePeriod の現行 CPU クロック・サイクルを戻します。

**lpszModelFile**　現行モデル名が検索されるバッファへのポインタ

**dwModelFileBufferLength**　lpszModeFile によって指定されるバッファの長さを含んでいます。

**lpszModel**　現行モデル名が検索されるバッファへのポインタ

**dwModelBufferLength**　lpszModel によって指定されるバッファの長さを含んでいます。

**DwProcFlags**　現行 CPU タイプ。次のフラグのいずれかが設定されます。

**EXECSTATE_UNKNOWN**
CPU タイプが不明です。このフラグは，モデル定義上の問題を示します。

**EXECSTATE_UPD77016**
CPU タイプは$\mu$PD77016 またはそれと互換のタイプです。$\mu$PD77017 に特有の関数と命令はディスエーブルされます。

**EXECSTATE_UPD77017**
CPU タイプは$\mu$PD77017 またはそれと互換のタイプです。このタイプは，モデルが$\mu$PD77015，77018，77019 の場合でも同様に戻されます。

**dwCyclePeriod**　ナノ秒（$10^{-9}$秒）で表される現行 CPU クロック・サイクル。

（3）HELPCTXDESC

HELPCTXDESC 構造体は，コンテキスト依存ヘルプに関する情報を含んでいます。

• 構文

```
typedef struct tagHELPCTXDESC {        /* id */
  DWORD dwSize;
  LPCTSTR lpszHelp;
  DWORD dwData;
} HELPCTXDESC, * LPHELPCTXDESC;
```

• メンバ

**dwSize**　メモリにあるこの構造体のサイズを示します。

**lpszHelp**　ヘルプ情報を含んでいるヘルプ・ファイルのファイル名を示します。

**dwData**　lpszHelp メンバで指定されているヘルプ・ファイル内の識別子を含んでいます。

保守／廃止

(4) ITEMDATA

ITEMDATA 構造体は，項目のデータと状態に関する情報を含んでいます。

• 構文

```
typedef struct tagITEMDATA {  /* id */
  DWORD dwSize;
  DWORD dwFlags;
  DWORD dwNumberOfBits;
  LARGE_INTEGER liValue;
  DWORD dwItemState;
}   ITEMDATA, * LPITEMDATA;
```

• メンバ

**dwSize**　メモリにあるこの構造体のサイズを示します。

**dwFlags**　どのデータが検索または設定されるのかを示します。どのフラグがどの関数で有効なのかに関する情報については，9.4　**関数参照**を参照してください。

**dwNumberOfBits**　dwValueLo および dwValueHi の有効なデータ・ビット数を含んでいます。このフィールドが使用される場合，ITEMDATA_BITS フラグを dwFlags で設定する必要があります。

**liValue**　項目データのデータ値を含んでいます。このフィールドが使用される場合，ITEMDATA_VALUE フラグを dwFlags で設定する必要があります。

**wItemState**　項目の状態フラグを含んでいます。このフィールドが使用される場合，ITEMDATA_VALUE フラグを dwFlags で設定する必要があります。このフィールドは，dwValueLo と dwValueHi のビット 0-15，16-31，32-47，および 48-63 の不定データをそれぞれ示すフラグ ITEMSTATE_UNDEF_L，ITEMSTATE_UNDEF_H，ITEMSTATE_UNDEF_E，および ITEMSTATE_UNDEF_X の任意の組み合わせを含んでいます。また，項目がハイ・インピーダンス状態をサポートしている場合には，このフィールドはフラグ ITEMSTATE_HIZ を含むことができます。

• 備考

GetItemProperty 関数を使って，その項目がハイ・インピーダンス状態をサポートしているかどうかを調べることができます。

保守／廃止

(5) ITEMPROPERTY

**ITEMPROPERTY 構造体は，項目に関する情報を受け取ります。**

- **構文**

```
typedef struct tagITEMPROPERTY {       // ip
  DWORD dwSize;
  DWORD dwMask;
  DWORD dwFlags;
  DWORD dwNumberOfBits;
  LPTSTR lpszName;
  DWORD dwBufferLength;
}   ITEMPROPERTY, * LPITEMPROPERTY;
```

- **メンバ**

**dwSize**　メモリにあるこの構造体のサイズを示します。

**dwMask**　どのデータが検索されるかを示します，次のフラグの組み合わせを設定できます。

**ITEMPROPMASK_BITS**
項目の有効なデータ・ビット数は，dwNumberOfBits で設定されます。

**ITEMPROPMASK_NAME**
項目名は，lpszName で指し示されるバッファで設定されます。

**ITEMPROPMASK_CAPABILITY**
項目の容量は dwFlags で設定されます。

**dwFlags**　項目の容量を受け取ります。次のフラグが設定できます。

**ITEMPROP_UNDEF_ABLE**
項目は，不定値を保持できます。項目データを設定または検索するとき，ITEMSTATE_UNDEF_*x* フラグが ITEMDATA 構造体の wItemstate 内に設定できます。

**ITEMPROP_HIZ_ABLE**
項目はハイ・インピーダンス状態をサポートしています。項目データを設定または検索するとき，ITEMSTATE_HIZ フラグは，ITEMDATA 構造体の wItemstate 内に設定できます。

**ITEMPROP_READ_ONLY**
項目は読み出し専用アクセスになっています。書き込み動作を行おうとすると，エラー・コードが戻されます。

**dwNumberOfBits**　項目の有効なデータ・ビット数を受け取ります。このフィールドが使用される場合，ITEMPROPMASK_BITS フラグを dwMask で設定する必要があります。

**lpszName**　項目名を受け取るバッファを指します。ITEMPROPMASK_NAME フラグは，このフィールドが使用される場合には dwMask で設定しなければなりません。

**dwBufferLength**　lpszName で与えられるバッファの長さを示します。

保守／廃止

(6) MEMCELLDATA

**MEMCELLDATA 構造体は，メモリ・セルの内容に関する情報を含んでいます。この構造体は，MEMCELLDATA 構造体の配列を作成する MEMDATA 構造体によってのみ使用されます。GetMemory および SetMemory 関数は，メモリ・ブロックの読み出しと書き込みに MEMDATA 構造体を使用します。**

**• 構文**

```
typedef struct tagMEMCELLDATA {        // mcd
  DWORD dwValue;
  DWORD dwState;
}  MEMCELLDATA, * LPMEMCELLDATA;
```

**• メンバ**

**dwValue**　**メモリ・セル値を含んでいます。**

**dwState**　**メモリ・セルの状態を含んでいます。次のフラグは SetMemory 関数の入力時に設定可能です。**

**MEMCELLSTATE_UNDEF**

**このフラグが設定されると，メモリ・セルは不定値に設定されます。**GetMemory **および** SetMemory **関数が終了したとき，次のフラグが設定可能になります。**

**MEMCELLSTATE_UNDEF**

**このフラグが設定されると，メモリ・セルは不定値に設定されます。**

**MEMCELLSTATE_ERROR**

**このフラグが設定されると，メモリ・アクセス・エラーが発生します。GetMemory および SetMemory 関数はエラー・コードを戻しますが，このフラグは，エラーを引き起こした特定のメモリ・セルを識別するのに使用できます。**

(7) MEMDATA

**MEMDATA 構造体は，メモリ・セルのブロックへの読み出しまたは書き込みアクセスに関する情報を含んでいます。GetMemory および SetMemory 関数は，メモリ・ブロックの読み出しまたは書き込みに MEMDATA 構造体を使用します。**

**• 構文**

```
typedef struct tagMEMDATA {  // md
  DWORD dwSize;
  DWORD dwFlags;
  DWORD dwMemAddr;
  DWORD dwMemSize;
  LPMEMCELLDATA pData;
}  MEMDATA, * LPMEMDATA;
```

保守／廃止

• メンバ

| | |
|---|---|
| dwSize | この構造体のサイズをバイトで示します。 |
| dwFlags | 現在使用されていません。0 に設定されています。 |
| dwMemAddr | メモリ・セル・ブロックの開始アドレスを含んでいます。X データ・メモリ・アドレスの範囲は 0x00000-0x0FFFF です。Y データ・メモリ・アドレスの範囲は 0x10000-0x1FFFF です。インストラクション・メモリ・アドレスの範囲は 0x20000-0x2FFFF です。 |
| dwMemSize | メモリ・セル・ブロックのサイズをメモリ・セル単位で含んでいます。 |
| pData | メモリ・セルの内容を保持または受けとる MEMCELLDATA 構造体の配列を指します。配列は，少なくとも dwMemSize エレメントを含んでいなければなりません。 |

(8) MEMMAPRW

MEMMAPRW 構造体は，ユーザ DLL のメモリ・マップ・エレメントへの読み出しまたは書き込みアクセスに関する情報を含んでいます。ReadWriteMemoryMappedElement 関数はこの構造体を使用して，読み出しまたは書き込みされたデータ・エレメントを獲得，設定します。またアクセスが疑似メモリ・アクセス命令（μPD77016 プログラムにおけるメモリ・アクセス命令）によって引き起こされたか，あるいはアクセスがユーザ・インタフェース動作（たとえば，データ・メモリ・ウインドウ更新）に引き起こされた場合，自身の内部処理を調整します。

• 構文

```
typedef struct tagMMRW {       // mmrw
  WORD    wMode;
  WORD    wRelAddr;
  WORD    wData;
}   MEMMAPRW, * LPMEMMAPRW;
```

• メンバ

**wMode**　メモリ・アクセス動作を示します。次のフラグが設定可能です。

**MMRWMDE_WRITE**

このフラグがセットされた場合，メモリ書き込み動作が発生します。wMode の wData フィールドと MMRWMDE_UNDEFDATA フラグが，ユーザ DLL のメモリ・マップ・エレメントに書き込まれるデータを示します。

このフラグがセットされていない場合，メモリ読み出し動作が発生します。ユーザ DLL のメモリ・マップ・エレメントの現在の値と状態を示す場合，wMode の wData フィールドと MMRWMDE_UNDEFDATA フラグ・ユーザ関数によってフィルする必要があります。

**MMRWMDE_CORE**

このフラグがセットされた場合，命令シミュレーション（例：μPD77016 プログラム・シミュレーション）によってメモリ・アクセスが起こります。ユーザ DLL はこのフラグを無視できます。

**MMRWMDE_SUPERVISOR**

このフラグがセットされた場合，ユーザ DLL はほかの読み出し専用値（いずれかの妥当な値）に書き込みアクセスを行えるようにします。ユーザ DLL はこのフラグを無視できます。

**MMRWMDE_NOVALIDATE**

**このフラグがセットされた場合，ユーザ DLL は，メモリ・マップ・エレメントに書き込みを行うときに余分なデータの妥当性検査をスキップします。**

**MMRWMDE_UNDEFDATA**

**このフラグがセットされた場合，不定データがメモリ・マップ・エレメントから読み出されるか，そのエレメントに書き込まれます。MMRWMDE_WRITE フラグの説明を参照してください。**

**wRelAddr**　**メモリ・マップ領域の基底アドレスに関連する，アドレスまたはアクセスされたメモリ・マップ・エレメントを含んでいます。**

**wData**　**アクセスされたメモリ・マップ・エレメントのデータ値を含んでいます。**

**（**9**）**MEMMAPSNAPSHOT

**MEMMAPSNAPSHOT 構造体は，メモリ・マップ領域のスナップショット動作に関する情報を含んでいます。**

- **構文**

```
typedef struct tagMMSS {       // mmss
  DWORD  dwMode;
  DWORD  dwBufferSize;
  LPBYTE  lpBuffer;
}   MEMMAPSNAPSHOT, * LPMEMMAPSNAPSHOT;
```

- **メンバ**

**dwMode**　**スナップショット動作の種類を示します。**

**dwBufferSize**　**メモリ・マップ領域の基底アドレスに関連する，アドレスまたはアクセスされたメモリ・マップ・エレメントを含んでいます。**

**lpBuffer**　**アクセスされたメモリ・マップ・エレメントのデータ値を含んでいます。**

**（**10**）**MENUDESC

**MENUDESC 構造体は，カスタム・メニュー拡張部分に関する情報を含んでいます。**

- **構文**

```
typedef struct tagMENUDESC {
  DWORD dwSize;
  DWORD dwFlags;
  HINSTANCE hInst;
  LPCTSTR lpMenu;
  CMDPROC lpCommandFunc;
  UPDCMDPROC lpUpdateCommandFunc;
  LPVOID lpReserved;
} MENUDESC, * LPMENUDESC;
```

保守／廃止

• メンバ

**dwSize**　この構造体のサイズをバイトで示します。

**dwFlags**　現在使用されていません。0 に設定してください。

**hInst**　メニュー・リソースを含んでいるモジュールのインスタンスを識別します。

**lpMenu**　メニュー・リソースを識別します。このパラメータは，メニュー・リソースの名前を指定する NULL で終わる文字ストリングのポインタか，あるいはメニューのリソース識別子を指定する整数値かのいずれかです。パラメータがリソース識別子を指定する場合，その上位ワードは 0 で，下位ワードは識別子である必要があります。ユーザがこの値を作成する場合，MAKEINTRESOURCE マクロを使用できます。

**lpCommandFunc**　メニュー・コマンド・プロシージャを指します。

**lpUpdateCommandFunc**　メニュー・コマンド更新プロシージャを指します。

**lpReserved**　現在使用されていません。0 に設定してください。

(11) WINDOWDESC

**WINDOWDESC 構造体は，カスタム・ウインドウ拡張部分に関する情報を含んでいます。**

• 構文

```
typedef struct tagWINDOWDESC {
  DWORD dwSize;
  DWORD dwFlags
  HINSTANCE hInst;
  LPCTSTR lpTemplate;
  DLGPROC lpDialogFunc;
  LPCTSTR lpszMenuText;
  LPCTSTR lpszMenuMessage;
  CMDPROC lpCommandFunc;
  UPDCMDPROC lpUpdateCommandFunc;
  LPVOID lpReserved
} WINDOWDESC, * LPWINDOWDESC;
```

• メンバ

**dwSize**　この構造体のサイズをバイトで示します。

**dwFlags**　現在使用されていません。0 に設定してください。

**hInst**　ダイアログ・テンプレート・リソースを含んでいるモジュールのインスタンスを識別します。

**lpTemplate**　ダイアログ・ボックス・テンプレートを識別します。このパラメータは，ダイアログ・ボックス・テンプレートの名前を指定する，NULL で終わる文字ストリングへのポインタか，あるいはダイアログ・ボックス・テンプレートのリソース識別子を指定する整数値のいずれかです。パラメータがリソース識別子を指定する場合，その上位ワードは 0 で，下位ワードは識別子である必要があります。この値をユーザが算出する場合には MAKEINTRESOURCE マクロを使用できます。

**lpDialogFunc**　ダイアログ・ボックス・プロシージャを指します。


**lpszMenuText**　ウインドウを開くためのメニュー項目テキストを含んでいる，NULL で終わる文字ストリングへのポインタ。このメニュー項目は，HSM77016 の Window メニューに追加されます。

**lpszMenuMessage**　ウインドウを開くためのメニューが選択されたときに HSM77016 のステータス・バーに置かれるストリングを含んでいる，NULL で終わる文字ストリングへのポインタ。

**lpCommandFunc**　メニュー・コマンド・プロシージャを指します。

**lpUpdateCommandFunc**　メニュー・コマンド更新プロシージャを指します。

**lpReserved**　現在使用されていません。0 に設定してください。

## 9.5 サンプル

次に示すサンプルは，HSM77016 パッケージとともに提供されます。これらのサンプルは，HSM77016 アプリケーション・インタフェースの性能を実証するために用意されているものです。詳しくは，対応の HSM77016 がインストールされている EXAMPLE フォルダにある README.TXT を参照してください。

### 9.5.1 Buffer View サンプル

Buffer View DLL は，HSM77016 のアプリケーション拡張部分です。この DLL は，ユーザ指定のメモリ範囲の内容を図で表示します。このサンプルで，次の HSM77016 アプリケーション・インタフェース機能が実証されます。

- CreateItemEventCallback
- DestroyEventCallback
- EnableEventCallback
- FindItem
- GetItem
- InstallUIExtension
- OutputMessageBox
- ReleaseItem
- UninstallUIExtension

保守／廃止

### 9.5.2 データ・モニタ・サンプル

データ・モニタ DLL は，HSM77016 のアプリケーション拡張部分です。この DLL は，有効な HSM77016 の式の変更を図で記録し,モニタ領域に表示します。このサンプルで,次の HSM77016 アプリケーション･インタフェース機能が実証されます。

- CreateItemEventCallback
- DestroyEventCallback
- EnableEventCallback
- FindItem
- GetItem
- InstallUIExtension
- ReleaseItem
- UninstallUIExtension

### 9.5.3 メモリ・バンク・サンプル

メモリ・バンク DLL は，メモリ・マップ・デバイスをシミュレートする HSM77016 のアプリケーション拡張部分です。デバイスは，サポートされているシミュレーション・モデル・ファイルで指定されているメモリ・バンクです。このサンプルで，次の HSM77016 アプリケーション・インタフェース機能が実証されます。

- CreateItem
- CreateMemoryMappedArea
- DestroyItem
- DestroyMemoryMappedArea
- InstallUIExtension
- ItemDataModified
- MakeSnapshotMemoryMappedArea
- MemoryDataModified
- OutputMessageBox
- ReadWriteMemoryMappedElement
- RestoreSnapshotMemoryMappedArea
- UninstallUIExtension

保守／廃止

### 9.5.4　タイマ・サンプル

**タイマ** DLL **は，**HSM77016 **のアプリケーション拡張部分です。この** DLL **は** DLL **がロードされてからの時間を秒単位でカウントするタイマ変数から構成されており，**HSM77016 **とロードされているモデル・ファイルの状態の情報を検索できるようになっています。このサンプルで，次の** HSM77016 **アプリケーション・インタフェース機能が実証されます。**

- CreateItem
- DestroyItem
- GetBuildVersion
- GetExecMode
- GetExecParam
- InstallUIExtension
- ItemDataModified
- UninstallUIExtension

〔メ　モ〕

# 第10章　プロファイリング

## 10.1　プロファイリングとは？

プロファイリングは，プログラム・コードの改良，プログラムのボトルネックの検出を行い，その結果プログラム・パフォーマンスを向上させることです。ボトルネック検出の観点から次の点が重要視されます。

- どの箇所でプログラムは時間を費やしたか？
- どのコードが最も使用されたか？

**注意　プロファイリングは，バグを発見する手法ではありません。プロファイリングは，ディバグ済みのプログラムに適用されます。プログラムのプロファイリングは，直線的で順方向的なものではなくインタラクティブなものです。そのためプロイファイル・パラメータを変更すると，プロファイリングの結果が異なったものとなり，その結果を異なった意味に解釈するおそれがでてきます。**

### 10.1.1　プログラムをプロファイリングし，改良するためのステップ

- プログラムをセットアップします。
- プロファイリング・マーカを設定し，プログラムを実行しながらプロファイリング・データを収集します。
- 収集したデータを Statistic ウインドウで分析します。
- 統計データをレポート・ファイルに保存します。
- プログラムを修正します。

プログラム修正後，プロファイリングを繰り返して，修正でプログラムのパフォーマンスが向上したかを確認します。

## 10.2　プロファイリング・マーカ

マーカは，プロファイリング・セッションを組み立てたり，プロファイリング情報を得るために使用します。プロファイリングの統計情報の重要性と可用性は，マーカを適切に設定するかどうかに大きく依存しています。マーカは次の位置で設定できます。

- Instruction Memory ウインドウのアドレス
- Module ウインドウのカーソル位置
- Statistic ウインドウ

保守／廃止

### 10.2.1　マーカ・タイプ

プロファイラでは，次のマーカ・タイプが使用できます。

- Label マーカ

  共通プロファイリング・マーカは，プログラム実行中にマーカ・アドレスがヒットした回数，および次のマーカがヒットするまでの経過時間を記録します。

- Entry および Exit マーカ

  Entry および Exit マーカは，組み合わせてプロファイリング・サブルーチン内で使用します（図 10－1　Entry マーカおよび Exit マーカの使用方法を参照）。Hit Entry マーカと Exit マーカを実行する際は，アクティブ・マーカ・スタックのオーバフローまたはアンダフローを防ぐために互いに一致している必要があります。アクティブ・マーカ・スタックのサイズは，Tools メニューの Options...コマンドで設定できます。

- Start マーカ

  実行がこのマーカに達したときにプロファイリングをオンにします。プロファイリングがすでに実行されているとき，Start マーカは無効です。

- Stop マーカ

  実行がこのマーカに達したときにプロファイリングをオフにします。

- プロファイリング・マーカによって記録されるデータ

- マーカは，プログラム・パフォーマンスの向上の観点から異なる結論をもたらす 2 種類のプロファイリング・データを記録します。

  - Hit Count データ

    Hit Count プロファイリングは，プログラム実行中にプロファイリング・マーカがヒットした回数を記録します。ヒット・カウント・データは，実行されたプログラム・コードがいくつあるか，特定のサブルーチンが呼ばれたか，またはプログラムが何回ループするかを調べるのに使われます。

  - Time Count データ

    Time Count プロファイリングは，複数のマーカ間でコードを実行するのに要した時間を記録します。

次の表に，マーカ・タイプ別に記録データを示します。

表10－1　記録マーカ・データ

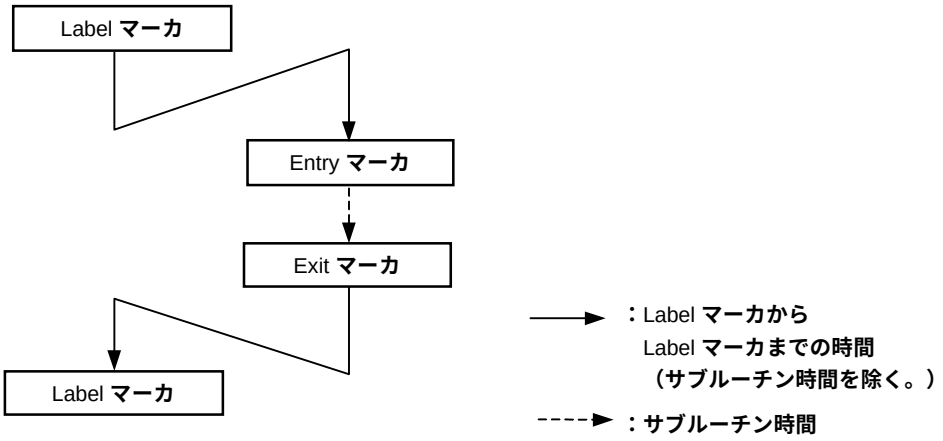
| マーカ名 | ヒット・カウント・データ | タイム・カウント・データ |
|---|---|---|
| Label マーカ | √ | √ |
| Entry マーカ | √ | √ |
| Exit マーカ | √ | － |
| Start マーカ | － | － |
| Stop マーカ | － | － |

保守／廃止

## 10.2.2　プロファイリング・サブルーチン

Entry マーカと Exit マーカは，図 10－1 に示すようにサブルーチンをプロファイリングするのに使用します。

**図 10－1　Entry マーカおよび Exit マーカの使用例**




Entry マーカは，先行 Label マーカの値をスタックに置きます。Exit マーカは，その Time Count 値をスタックに追加し，合計値を先行 Label マーカに戻します。その結果，最初の Label マーカは次の Label マーカまでの時間を測定します。このときサブルーチンの実行時間は無視されます。サブルーチンの実行時間は Entry マーカに割り当てられています。

図 10－2 は，Module ウインドウでの，ネストした 2 つのサブルーチンのプロファイリングに使用するマーカ設定を示しています。

**図 10－2　サブルーチンのプロファイリング**

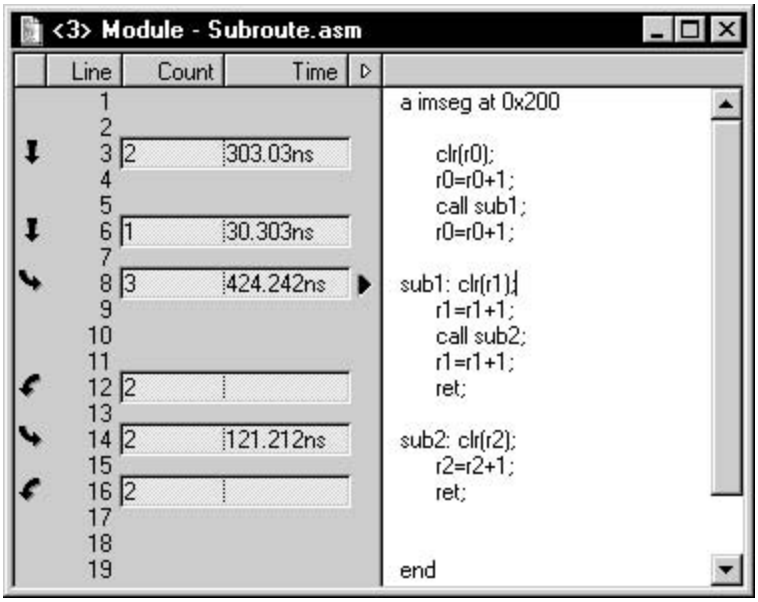



図 10－2 に示しているメイン・プログラムは，サブルーチン sub2 を呼び出すサブルーチン sub1 を呼び出します。マーカは，次の表に示している行で設定されます。

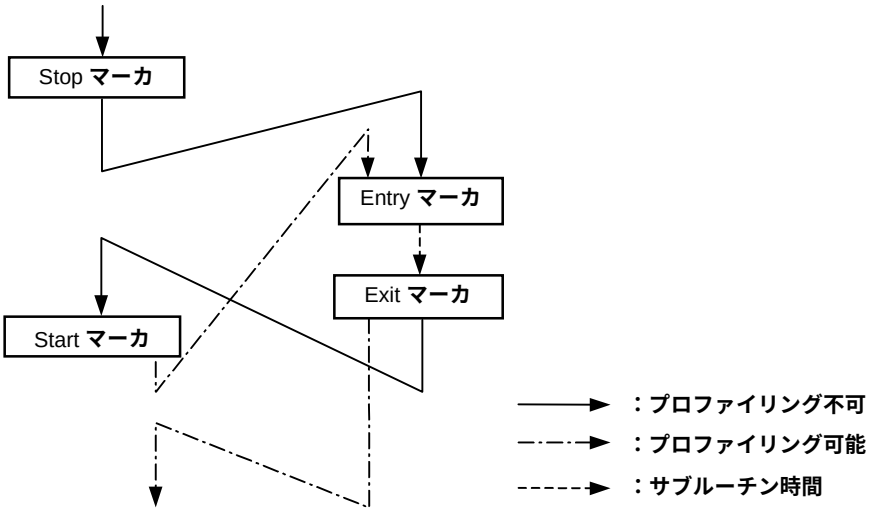
| 行 | マーカ |
|---|---|
| 3 | メイン・プログラムの Entry ポイントでの最初の Label マーカ |
| 6 | メイン・プログラムの終了ポイントでの 2 番目の Label マーカ |
| 8 | sub1 の Entry ポイントでの Entry マーカ |
| 12 | sub1 の終了ポイントでの Exit マーカ |
| 14 | sub2 の Entry ポイントでの Entry マーカ |
| 16 | sub2 の終了ポイントでの Exit マーカ |

プロファイリングの結果，最初の Label マーカがメイン・プログラムの実行時間を記録します。このときサブルーチンの実行時間は無視されます。sub1 の実行時間は最初の Entry マーカによって，sub2 の実行時間は 2 番目の Entry マーカによってそれぞれ記録されます。

### 10.2.3　Start マーカと Stop マーカの使用方法

図 10－3 は，メイン・プログラムにおける 2 つの異なるポイントから呼び出されるサブルーチンをプロファイリングする際の Start マーカと Stop マーカの使用例を示しています。Start マーカと Stop マーカは，プロファイリングされるプログラム・セクションの指定に使用されます。

図 10－3　Start マーカと Stop マーカの使用例




Stop マーカは，サブルーチンが最初に呼び出される前にプロファイリングをディスエーブルします。実行が呼び出しポイントに戻り，Start マーカに達すると，プロファイリングがイネーブルされます。サブルーチンが 2 番目の呼び出しポイントから呼び出されると，Entry マーカと Exit マーカが起動されます。

## 10.3　プロファイリング結果の分析

どのプロイファイル・データを収集するかを決定し，それに応じてプログラムを調整してプログラムを実行したあとでプロファイリング結果を検討するのが適切な方法です。Statistic ウインドウはプロファイル・セッションの要約を示し，プロファイリング結果の概観を示します。

- 指定されたマーカのタイプ
- 対応するマーカが設定されたアドレス
- マーカがヒットした回数
- 次のマーカまでの経過時間
- 指定されたマーカの情報

Statistic ウインドウに表示された収集データを評価すると，ソース・コードのどの部分を変えたらプログラム・パフォーマンスが向上するかがわかります。通常，ソース・コード中の次の部分（ルーチン）が検討されることになります。ただし，このようなポイントは最適化のためのおおよその指標にすぎません。

- 実行時間の大部分を占める部分
- 使用頻度の高い部分
- 実行時間 vs. 実行カウント（1 回の呼び出し値あたりの時間）の比が高い部分

## 10.4　プロファイリング・レポート

プロファイリング・セッション中に Statistic ウインドウで収集したデータは，レポート・ファイルに保存できます。レポート・ファイルとは，次の情報を含んでいる，テキスト・フォーマットでのファイルのことです。
レポート・ファイル例については，4.5　**レポート・ファイル**を参照してください。

- ファイルのヘッダ
  レポート作成の際に使用したプログラムの名前とバージョン，レポート・ファイルのパスと名前，レポートの作成日時。
- プログラム統計情報
  レポートに関連するリンク・ファイルのパスと名前。
- プロファイル統計情報
  マーカ総数，時間合計値およびヒット・カウント総回数，データ並べ替え順序，Statistic ウインドウで収集したデータを含んでいるデータ・テーブル。
- タイプの凡例
  データ・テーブルのマーカ・タイプ・シンボルの説明。

## 10.5　プロファイリング・データの保存

プロファイリング・データ（マーカ設定）は，Tools メニューの Options...コマンドで保存，復帰できます。設定ファイルをロードすると，現在のマーカすべてが削除されます。

〔メ　モ〕

保守／廃止

# 第11章　レファレンス

## 11.1　インライン・アセンブラと WB77016 アセンブラ

命令は，ニモニック形式で Instruction Memory ウインドウに入力できます。それらの命令は内蔵インライン・アセンブラでアセンブルされます。使用するニモニック言語は，次に示す制約を持つ WB77016（μPD77016 ワークベンチ・リロケータブル・アセンブラ）で使用されるものと同様です。

- μPD77016 命令以外の疑似命令またはステートメントは使用できません。
- セグメントやシンボルは定義できません。
- ループ命令のアセンブラ構文をコード化するには次のとおりです。アセンブラ構文もサポートされています。

```
loop loop_count {
.
.
.
}

-or-

loop loop count { [start_address] …end_address }
```

## 11.2　未定義値の概念

より精度が高く，優れたシミュレーション結果を得るために，シミュレーションは“未定義”という値を追加して 2 進数の世界を拡大しています。この値は，プログラムで決めることができないすべての値を表しています。

シミュレーションの場合，すべての値には，“未定義”としてフラグを立てる追加ビットがあります。このビットは値全体を指しています。したがって，どの値も定義部分および未定義部分の双方（つまりビット）を含むことはできません。ただし，汎用レジスタ R0-R7 は例外で，次の 3 つの定義済みフラグを持っています。

- 上位 8 ビット用の拡張部（R0E など）
- 中位 16 ビット用の高位部（R0H など）
- 下位 16 ビット用の低位部（R0L など）

したがって，これらのレジスタは定義済み部分と未定義部分を含むことができます。しかし，レジスタが未定義部分を含んでいる場合，Split 8'16'16 数値形式オプションが選択されていないかぎり未定義として表示されます。

シミュレーション結果が，たまたま初期化されていない値（しかし，参照された値）に依存しないように未定義値が加えられています。最初，すべての値は，自動的に実行されるチップ・リセットによって定数値に初期化されているものを除いて，未定義と仮定されます。また，すべてのメモリ領域も入力時に未定義となっています。

**(1) 未定義ソース**

**ハードウエア・チップには未定義値がないので（すべてのビット・セルは** “0” **か** “1” **です），未定義値の発生原因は次のソース群のいずれかになります。**

- **シミュレーション新規開始**

  HSM77016 **起動時または** File **メニューから** New **コマンドを選択後，ハードウエアのリセットに影響を受けない各メモリ領域およびチップ・レジスタは未定義に設定されます。**
- **ユーザ**

  **命令ポインタ（**IP**），ステータス・レジスタ（**SR**），および内蔵** I/O **デバイス・ステータス・レジスタを除いて，各値は未定義に設定できます。**

**(2) 未定義値の特殊動作**

**未定義に設定するとき,ユーザに対して警告を出す値もあります。この警告が出た場合,ユーザはシミュレーションを未定義値で続行するかブレークするかを選択できます。**

## 11.3　C 言語の式

C **言語の式のフォーマットは次の属性から構成されています。**

- **式表示形式は，式データ・タイプから自動的に得られます。**
- C **言語変数および** C **言語式にタイプの情報が使用できます。**
- **ポインタ式は，**16 **進形式で表示されます。**

### 11.3.1　パーサの切り替え

C **言語またはアセンブラの式を正しく評価するため，該当するパーサを選択する必要があります。パーサの切り替えは** Options **メニューの** Language **コマンドのサブコマンドをチェックすることで行います。ロードされているリンク・ファイルが** C **ディバグ情報を含んでいる場合のみ，**C **パーサが使用できます。**

**パーサの切り替えは，**C **言語変数とアセンブラ予約語を区別する場合に必要です。**C **プログラムが** HSM77016 **の変数でもある変数** r0 **を宣言していると仮定した場合，**C **パーサが選択されると** r0 **は** C **変数と解釈されます。したがって，**r0 **が，たとえば** Watch **ウインドウ・エントリとして参照される場合，対応する** C **変数の値が評価されます。アセンブラ・パーサに切り替えると，汎用レジスタ** R0 **の値が表示されます。**

**しかし，**C **パーサが選択されている場合に完全な（省略していない）入力構文を使用すれば，**HSM77016 **は** HSM77016 **変数の評価を強制的に行うことができます(たとえば,**Watch **ウインドウ･エントリの** r0 **ではなく** regr0 **を指定すると，**HSM77016 **は汎用レジスタ** R0 **の値を表示します)。**

保守／廃止

# 11.4　メモリ範囲の入力

HSM77016 ダイアログにはプロンプトを表示して必要なメモリ範囲の入力を指示するものがありますが，メモリ範囲の入力は Memory Range 入力フィールドに直接行うか，この入力をサポートしているダイアログを使用して（Select...ボタンをクリックして）行います（この場合，構文に関する知識は不要です）。これらのダイアログは次に示す動作中に表示されます。

- ディスティネーション・ソースとディスティネーション・ターゲットを指定するデータ・ファイルの，インポート，およびエクスポート（File メニューの Import...コマンド，Export...コマンドで始動されます。）
- メモリ検索およびフィル動作（Memory メニューの Search...コマンドおよび Fill...コマンドで始動されます。）

## 11.4.1　メモリ範囲フィールドの入力

**（1）構文**

*start_address*:*memory_space*,*block_size*

**（2）パラメータ**

| | |
|---|---|
| *start_address* | メモリ範囲の開始アドレス |
| *memory_space* | インストラクション・メモリの場合は I<br>X データ・メモリの場合は X<br>Y データ・メモリの場合は Y |
| *block_size* | メモリ・ブロックの長さ（メモリ・ワード数） |

**（3）備考**

これらのエントリの場合，大文字と小文字の区別はされません。

**（4）例**

次に示す例は，Y データ・メモリのアドレス 0x200 から始まる 4 つのメモリ・ワードを指定しています。したがって，メモリ範囲は Y データ・メモリ 0x200～0x203（両方を含む）となります。

0x200:Y,4

保守／廃止

## 11.5　数値形式の構文

### 11.5.1　10 進数（基数が 10）

**(1) 構文**

[ **+** | **-** ] [ **0T** | **0t** ] *decimal_digit...*

**(2) 備考**

**接頭語ストリング** 0T **または** 0t **はオプションです。負の数は** 2 **の補数で表されます。**

**(3) 例**

**有効：**　0t1000, 2345

**無効：**　0t8FD2, 3DH

### 11.5.2　16 進数（基数が 16）

**(1) 構文**

[ **+** | **-** ] **0X** | **0x** *hexadecimal_digit...*

**(2) 備考**

**接頭語ストリング** 0X **または** 0x **が必要です。負の数は** 2 **の補数で表されます。**

**(3) 例**

**有効：**　0xEA00, 0xD000

**無効：**　0x8FG2, D000

### 11.5.3　8 進数（基数が 8）

**(1) 構文**

[ **+** | **-** ] **0Q** | **0q** | **0** *octal_digit...*

**(2) 備考**

**接頭語ストリング** 0Q, 0q **または** 0 **が必要です。負の数は** 2 **の補数で表されます。**

**(3) 例**

**有効：**　0q7777, 03742

**無効：**　0q7890, 39FA

保守／廃止

### 11.5.4　2 進数（基数が 2）

**(1) 構文**

[ **+** | **-** ] **0Y** | **0y** *binary_digit...*

[ **+** | **-** ] **0B** | **0b** *binary_digit...*

**(2) 備考**

**接頭語ストリング** 0Y/0B **または** 0y/0b **が必要です。負の数は** 2 **の補数で表されます。**

**(3) 例**

**有効：**　0y1000

0b1000

**無効：**　0y1120, 2100

0b1120, 2100

### 11.5.5　固定小数点数

**(1) 構文**

[ **+** | **-** ] integer_part.decimal_part

[ **E** [ **+** | **-** ] exponent ]

[ **[** #_of_bits_before_dot . #_of_bits_after_dot **]** |

**[** #_of_bits_after_dot **]** ]

**(2) 備考**

**固定小数点数は** 2 **進の** 2 **の補数形式に変換され，**“10 **進”小数点がビット数***#_of_bits_after_dot* **のすぐ左に置かれます。***#_of_bits_before_dot* **有効ビット位置は** 10 **進小数点の左であり，***#_of_bits_after_dot* **有効ビット位置は** 10 **進小数点の右です。固定小数点数の範囲は任意とすることができます。それを越えるビットは，通知なしで無視されます。**

**(3) 例**

2 **の補数での** 3.25 **の** 2 **進表現は**...011.0100...**です（**16 **ビットの結果，**10 **進小数点前の有効ビットが太字で印刷されています)。**

**有効：**　**3.25 [9.7]**　**0000 0001 1**010 0000

**3.25 [1.15]**　**1**010 0000 0000 0000　(!)

**3.25 [11]**　**0001 1**010 0000 0000

**無効：**　**3[9.7], 4.0[4.1], 0.2[18]**

保守／廃止

### 11.5.6　40 ビット E-H-L 分割数

**(1) 構文**

E_value ' H_value ' L_value

**(2) 備考**

この形式は，汎用ワーク・レジスタ R0-R7 の拡張部分（E，ビット 39-32），上位部分（H，ビット 31-16）および下位部分（L，ビット 15-0）を直接入力できるように設計されています。E_value，H_value，および L_value は整数とすることができます。

**(3) 例**

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 有効： | **0x12 ' 0 ' 0** | 0001 | 0010 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| | **0xFF ' 0x30 ' 1** | 1111 | 1111 | 0000 | 0000 | 0011 | 0000 | 0000 | 0000 | 0000 | 0001 |
| | **??? ' 0t10 ' ???** | ???? | ???? | 0000 | 0000 | 0000 | 1010 | ???? | ???? | ???? | ???? |
| 無効 | **0 ' 3 + 4 ' 3-1** | | | | | | | | | | |

### 11.5.7　ニモニック

内蔵μPD77016 逆アセンブラは，数字をニモニック形式で出力するのに使用します。内蔵μPD77016 インライン・アセンブラは入力を変換するのに使用します。

### 11.5.8　C 言語式

C 言語式のフォーマットは，次の属性から構成されています。

- 式表示形式は，式データ・タイプから自動的に得られます。
- C 言語変数および C 言語式にタイプ情報が使用できます。
- ポインタ式は，16 進形式で表示されます。

### 11.5.9　状態表示形式

この表示形式は，ビット・フィールドにのみ適用されます。この表示形式を選択した場合，影響を受けたビット・フィールドはビット状態を示します。チェックされたビット・フィールドは，位置調整されたテキストのステートメントが肯定されたことを意味します。空のビット・フィールドは，位置調整されたテキストのステートメントが否定されたことを意味します。

**備考**　状態表示形式は，値表示形式と同じ意味を持つことはできません（ビットはその設定に関係なくアクティブとすることができます）。

保守／廃止

### 11.5.10　値表示形式

この表示形式は，ビット・フィールドにのみ適用されます。この表示形式が選択された場合，影響を受けたビット・フィールドはビット値を持ちます。設定されたビットは 1 で示され，設定されていないビットは 0 で示されます。

**備考**　値表示形式は，状態表示形式と異なる意味を持つことがあります（ビットはその設定に関係なくアクティブとすることができます）。

### 11.5.11　未定義値

任意の数の疑問符は未定義値として機能します。未定義値が出力された場合，3 つの疑問符（???）が使用されます。

**（1）構文**

**?**...

**（2）備考**

ほかの文字が検出された場合，その文字はシンボル名として解釈されます。

**（3）例**

**有効：**　???,?,???????

**無効：**　?X,??PRINT??

### 11.5.12　ハイ・インピーダンス値

任意の数の@はハイ・インピーダンス値として機能します。ハイ・インピーダンス値が出力された場合，3 つの@（@@@）が使用されます。

**（1）構文**

@...

**（2）備考**

ほかの文字が検出された場合，その文字はシンボル名と解釈されます。ハイ・インピーダンス値は，双方向端子およびポートなどのハイ・インピーダンスを持つオブジェクトとともにのみ使用できます。

**（3）例**

**有効：**　@@@, @, @@@@@@@@

**無効：**　@X, @@PRINT@@

保守／廃止

## 11.6　演算子

次の演算子が HSM77016 上で使用できます。

| 優先順位 | 演算子 | 名　　称 |
|---|---|---|
| 高 | ( ... ) | グループ |
| | * | 間接 |
| | ~ | ビット単位で NOT（すべてのビットをトグルする） |
| | - | 単項マイナス |
| | + | 単項プラス |
| | * | 乗算 |
| | / | 除算 |
| | % | 剰余 |
| | + | 加算 |
| | - | 減算 |
| | << | 左シフト |
| | >> | 右シフト |
| | < | より小 |
| | <= | 以下 |
| | > | より大 |
| | >= | 以上 |
| | == | 等しい |
| | != | 不当 |
| | & | ビット単位で AND |
| | ^ | ビット単位で排他 OR |
| | \| | ビット単位で OR |
| | && | 論理 AND |
| 低 | \|\| | 論理 OR |

**注意　比較演算子（=, <, >, ...）は，条件が真の場合には 1 を戻し，そうでない場合には 0 を戻します。論理演算子は，非ゼロ定義値を真と見なします。この結果とすべての演算子は，40 ビット整数とともに機能します。数字の代わりにレジスタも指定できます。40 ビット未満のレジスタが指定された場合，それらのレジスタは 40 ビットまで符号拡張されます。すべての演算子は未定義値にも正しく機能します。間接演算子（*）は，指定のアドレスでメモリ・セルのアドレスを戻します。**

```
*address : memory_space
```

address はメモリ・セルのアドレスです，

memory_space は，インストラクション・メモリの場合には I，X データ・メモリの場合には X，Y データ・メモリの場合には Y です。

保守／廃止

### 11.6.1　数字の処理

HSM77016 の数字処理では，データ値の実際のビット幅が考慮されます。また，演算子（式）を使用する評価は，符号付き 40 ビット幅の結果として処理されます。11.7　変　　数には，適正な入力数字形式を選択する際の，それぞれの式入力に関する予想結果を実現する場合に考慮される実際のビット幅が含まれます。

## ★ 11.7　変　　数

次に示すアルファベット順のリストは，変数の概要を示しています。これらの変数の内容は次のとおりです。

- 変数の省略名
- 変数名
- 変数のサイズまたはタイプ，およびビット・フィールドの場合にはビット数
- 入力構文（タイミング・ファイル，Evaluate And Modify Variable ダイアログおよび Watch ウインドウの変数を指定するのに使用）
- ビット・フィールドの場合にはビット値の意味

**注意**　すべての HSM77016 変数は，最初に使用する前に初期化しなければなりません。初期化しない場合は，メッセージがシミュレーションを中断し，それぞれの変数の初期化を指示します。シミュレーションによって現在使用されている変数はロックされ，ほかの箇所で使用できません。変数の入力構文では，大文字と小文字は区別されません。

保守／廃止

HSM77016 の端子変数

| 略　　号 | 名　　称 | 入出力 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|---|
| CLKIN | クロック入力 | 入力 | 1 | PIN CLKIN | |
| CLKOUT | クロック出力 | 入力 | 1 | PIN CLKOUT | |
| CSTOP | クロック停止 | 入力 | 1 | PIN CSTOP | |
| HALTS | ホールト出力 | 出力 | 1 | PIN HALT | |
| HCS | チップ選択入力 | 入力 | 1 | PIN HCS | 1=ハイ・レベル |
| HRD | ホスト読み出し入力 | 入力 | 1 | PIN HRD | |
| HRE | ホスト読み出しイネーブル出力 | 出力 | 1 | PIN HRE | |
| HWE | ホスト書き込みイネーブル出力 | 出力 | 1 | PIN HWE | |
| HWR | ホスト書き込み入力 | 入力 | 1 | PIN HWR | |
| INT1 | マスク可能外部割り込み入力 | 入力 | 1 | PIN INT1 | |
| INT2 | | | 1 | PIN INT2 | |
| INT3 | | | 1 | PIN INT3 | |
| INT4 | | | 1 | PIN INT4 | |
| INT5 | | | 1 | PIN INT5 | |
| INT6 | | | 1 | PIN INT6 | |
| INT7 | | | 1 | PIN INT7 | |
| INT8 | | | 1 | PIN INT8 | |
| INT9 | | | 1 | PIN INT9 | |
| INT10 | | | 1 | PIN INT10 | |
| P0 | 汎用入出力端子 | 入出力 | 1 | PIN PIO | |
| P1 | | | 1 | PIN PI1 | |
| P2 | | | 1 | PIN PIO2 | |
| P3 | | | 1 | PIN PIO3 | |
| P4 | | | 1 | PIN PIO4 | |
| P5 | | | 1 | PIN PIO5 | |
| P6 | | | 1 | PIN PIO6 | |
| P7 | | | 1 | PIN PIO7 | |
| P8 | | | 1 | PIN PIO8 | |
| P9 | | | 1 | PIN PIO9 | |
| P10 | | | 1 | PIN P10 | |
| P11 | | | 1 | PIN P11 | |
| P12 | | | 1 | PIN P12 | |
| P13 | | | 1 | PIN P13 | |
| P14 | | | 1 | PIN P14 | |
| P15 | | | 1 | PIN P15 | |
| P16 | | | 1 | PIN P16 | |
| P17 | | | 1 | PIN P17 | |
| P18 | | | 1 | PIN P18 | |
| P19 | | | 1 | PIN P19 | |
| PLL0 | PLL 設定 | 入力 | 1 | PIN PLL0 | |
| PLL1 | | 入力 | 1 | PIN PLL1 | |
| PLL2 | | 入力 | 1 | PIN PLL2 | |

保守／廃止

HSM77016 **の端子変数**

| 略　　号 | 名　　称 | タイプ | サイズ（ビット数.） | 構　　文 | 備　　考 |
|---|---|---|---|---|---|
| RESET<br>RESETS | リセット信号入力<br>リセット信号出力 | 入力<br>出力 | 1<br>1 | PIN RESET<br>PIN RESETS | |
| RFS | シリアル入力イネーブル | 入出力 | 1 | PIN RFS | スレーブ・モード |
| SCK1 | シリアル 1 クロック入力 | 入出力 | 1 | PIN SCK1 | |
| SCK2 | シリアル 2 クロック入力 | 入力 | 1 | PIN SCK2 | |
| SI1 | シリアル・データ入力 1 | 入力 | 1 | PIN SI1 | |
| SI2 | シリアル・データ入力 2 | 入力 | 1 | PIN SI2 | |
| SIAK1 | シリアル入力 1 アクノリッジ | 出力 | 1 | PIN SIAK1 | |
| SIAK2 | シリアル入力 2 アクノリッジ | 出力 | 1 | PIN SIAK2 | |
| SIEN1 | シリアル入力 1 イネーブル | 入出力 | 1 | PIN SIEN1 | |
| SIEN2 | シリアル入力 2 イネーブル | 入力 | 1 | PIN SIEN2 | |
| SO1 | シリアル・データ出力 1 | 出力／ハイ・インピーダンス | 1 | PIN SO1 | |
| SO2 | シリアル・データ出力 2 | 出力／ハイ・インピーダンス | 1 | PIN SO2 | |
| SOEN1 | シリアル出力 1 イネーブル | 入出力 | 1 | PIN SOEN1 | |
| SOEN2 | シリアル出力 2 イネーブル | 入出力 | 1 | PIN SOEN2 | |
| SORQ1 | シリアル出力 1 要求 | 出力 | 1 | PIN SORQ1 | |
| SORQ2 | シリアル出力 2 要求 | 出力 | 1 | PIN SORQ2 | |
| STOPS | ストップ信号出力 | 出力 | 1 | PIN STOPS | |
| TFS | シリアル出力イネーブル | 入出力 | 1 | PIN TFS | スレーブ・モード |
| TSDR | シリアル・データ入力 | 入力 | 1 | PIN TSDR | スレーブ・モード |
| TSDT | シリアル・データ出力 | 出力／ハイ・インピーダンス | 1 | PIN TSDT | スレーブ・モード |
| WAIT | ウエイト信号入力 | 入力 | 1 | PIN WAIT | |

HSM77016 **ポート変数**

| 略　　号 | 名　　称 | サイズ（ビット数.） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| HA | ホスト・アクセス | 2 | PORT HA | 入力ポート |
| HD | ホスト・データ・バス | 8 | PORT HD | 入出力またはハイ・インピーダンス・ポート |
| SI1 | シリアル・データ入力 1 | 16 | PORT SI1 | 入力ポート |
| SI2 | シリアルデータ入力 2 | 16 | PORT SI2 | 入力ポート |
| SO1 | シリアル・データ出力 1 | 16 | PORT SO1 | 出力またはハイ・インピーダンス・ポート |
| SO2 | シリアル・データ出力 2 | 16 | PORT SO2 | 出力またはハイ・インピーダンス・ポート |
| TSDR | シリアル・データ入力 | 16 | PORT TSDR | 入力ポート |
| TSDT | シリアル・データ出力 | 16 | PORT TSDT | 出力またはハイ・インピーダンス・ポート |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| CLKCNTL | | 16 | REG CLKCNTL | |
| CKOEN | | 1 (5) | REG CLKCNTL.CKOEN | |
| LOCK | | 1 (8) | REG CLKCNTL.LOCK | |
| ON | | 1 (7) | REG CLKCNTL.ON | |
| SEL | | 1 (6) | REG CLKCNTL.SEL | |
| DDMACOM | データ DMA コマンド | 16 | REG DDMACOM | |
| MODE | | 2 (0,1) | REG DDMACOM.MODE | |
| XY | | 1 (2) | REG DDMACOM.XY | |
| DDMASTS | データ DMA 状態 | 16 | REG DDMASTS | |
| BI | | 1 (15) | REG DDMASTS.BI | |
| MODE | | 2 (0,1) | REG DDMASTS.MODE | |
| XY | | 1 (2) | REG DDMASTS.XY | |
| DEADD | データ DMA 外部開始アドレス | 16 | REG DEADD | |
| DIADD | データ DMA 内部開始アドレス | 16 | REG DIADD | |
| DMOD | データ DMA 外部モジュロ・アドレス | 16 | REG DMOD | |
| DMX | モジュロ・レジスタ | 16 | [REG] DMX | |
| DMY | | | [REG] DMY | |
| DN0 | インデクス・レジスタ | 16 | [REG] DN0 | |
| DN1 | | | [REG] DN1 | |
| DN2 | | | [REG] DN2 | |
| DN3 | | | [REG] DN3 | |
| DN4 | | | [REG] DN4 | |
| DN5 | | | [REG] DN5 | |
| DN6 | | | [REG] DN6 | |
| DN7 | | | [REG] DN7 | |
| DOFF1 | データ DMA 外部オフセット・アドレス | 16 | REG DOFF1 | |
| DOFF2 | | | REG DOFF2 | |
| DP0 | データ・ポインタ | 16 | [REG] DP0 | |
| DP1 | | | [REG] DP1 | |
| DP2 | | | [REG] DP2 | |
| DP3 | | | [REG] DP3 | |
| DP4 | | | [REG] DP4 | |
| DP5 | | | [REG] DP5 | |
| DP6 | | | [REG] DP6 | |
| DP7 | | | [REG] DP7 | |
| DWCOUNT | データ DMA ワード・カウント | 16 | REG DWCOUNT | |
| WC1 | | 8 (0,1,2,3,4,5,6,7) | REG DWCOUNT.WC1 | |
| WC2 | | 8(8,9,10,11,12,13, 14,15) | REG DWCOUNT.WC2 | |

保守／廃止

HSM77016 **のレジスタ変数**

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| DWTR | データ・ウエイト・サイクル | 16 | REG DWTR | 00=ウエイト・サイクルなし<br>01=1 ウエイト・サイクル<br>10=3 ウエイト・サイクル<br>11=7 ウエイト・サイクル |
| B | データ・ウエイト・サイクル | 2 (2,3) | REG DTWR.B | |
| C | 領域 | 2 (4,5) | REG DTWR.C | |
| D | | 2 (6,7) | REG DTWR.D | |
| F | | 2 (10,11) | REG DTWR.F | |
| G | | 2 (12,13) | REG DTWR.G | |
| H | | 2 (14,15) | REG DTWR.H | |
| EIR | 割り込みイネーブル・スタック | 16 | [REG] EIR | |
| EI | 割り込みイネーブル・フラグ | 1 (15) | [REG] EIR.EI | 0=イネーブル<br>1=ディスエーブル |
| EP | | 1 (14) | [REG] EIR.EP | |
| EB | | 1 (13) | [REG] EIR.EB | |
| E3 | | 1 (12) | [REG] EIR.E3 | |
| E4 | | 1 (11) | [REG] EIR.E4 | |
| E5 | | 1 (10) | [REG] EIR.E5 | |
| E6 | | 1 (9) | [REG] EIR.E6 | |
| E7 | | 1 (8) | [REG] EIR.E7 | |
| E8 | | 1 (7) | [REG] EIR.E8 | |
| E9 | | 1 (6) | [REG] EIR.E9 | |
| E10 | | 1 (5) | [REG] EIR.E10 | |
| E11 | | 1 (4) | [REG] EIR.E11 | |
| E12 | | 1 (3) | [REG] EIR.E12 | |
| E13 | | 1 (2) | [REG] EIR.E13 | |
| E14 | | 1 (1) | [REG] EIR.E14 | |
| E15 | | 1 (0) | [REG] EIR.E15 | |
| ESR | エラー状態 | 16 | [REG] ESR | 0=エラーなし<br>1=エラー |
| BAC | バス・アクセス・エラー | 1 (0) | [REG] ESR.BAC | |
| LSE | ループ・スタック・エラー | 1 (1) | [REG] ESR.LSE | |
| STE | スタック・エラー | 1 (2) | [REG] ESR.STE | |
| OVF | オーバフロー・エラー | 1 (3) | [REG] ESR.OVF | |
| EWTR | 外部直接アクセス・メモリ・ | 16 | REG EWTR | |
| EC | ウエイト・サイクル制御 | 4 (0,1,2,3) | REG EWTR.EC | |
| ED | | 4 (4,5,6,7) | REG EWTR.ED | |
| EG | | 4 (8,9,10,11) | REG EWTR.EG | |
| EH | | 4 (12,13,14,15) | REG EWTR.EH | |
| HDTIN | ホスト・データ入力 | 16 | REG HDTIN | |
| HDTOUT | ホスト・データ出力 | 16 | REG HDTOUT | |
| HPCD | ホスト・ポート・コマンド・ | 16 | REG HPCD | |
| B | データ | 3 (8,9,10) | REG HPCD.B | |
| BE | | 1 (15) | REG HPCD.BE | |
| IO | | 1 (12) | REG HPCD.IO | |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| M0 | ホスト・ポート・コマンド・ | 1 (0) | REG HPCD.M0 | |
| M1 | データ | 1 (1) | REG HPCD.M1 | |
| M2 | | 1 (2) | REG HPCD.M2 | |
| M3 | | 1 (3) | REG HPCD.M3 | |
| M4 | | 1 (4) | REG HPCD.M4 | |
| M5 | | 1 (5) | REG HPCD.M5 | |
| M6 | | 1 (6) | REG HPCD.M6 | |
| M7 | | 1 (7) | REG HPCD.M7 | |
| ME | | 1 (13) | REG HPCD.ME | |
| PSR | | 1 (14) | REG HPCD.PSR | |
| HPDR | ホスト・ポート・データ | 16 | REG HPDR | |
| HOST | ホスト状態 | 16 | REG HST | |
| HAWE | アクセス・ウエイト・イネーブル | 1 (10) | REG HST.HAWE | 0=ディスエーブル |
| HLER | ロード・エラー | 1 (2) | REG HST.HLER | 0=エラーなし |
| HPF | | 1 (12) | REG HST.HPF | |
| HPM | | 1 (11) | REG HST.HPM | |
| HREF | ホスト読み出しイネーブル | 1 (1) | REG HST.HREF | 1=イネーブル |
| HREM | HRE マスク | 1 (9) | REG HST.HREM | 1=マスク |
| HRER | 読み出しエラー | 1 (5) | REG HST.HRER | |
| HSER | 格納エラー | 1 (3) | REG HST.HSER | 1=エラー |
| HWEF | ホスト書き込みイネーブル | 1 (0) | REG HST.HWEF | 0=ディスエーブル |
| HWEM | HWE マスク | 1 (8) | REG HST.HWEM | 0=マスクなし |
| HWER | 書き込みエラー | 1 (4) | REG HST.HWER | |
| UF0 | ユーザ・フラグ | 1 (6) | REG HST.UF0 | |
| UF1 | | 1 (7) | REG HST.UF1 | |
| ICR | 割り込みコントロール | 16 | REG ICR | |
| I5 | | 1 (4) | REG ICR.I5 | |
| I6 | | 1 (5) | REG ICR.I6 | |
| I7 | | 1 (6) | REG ICR.I7 | |
| I8 | | 1 (7) | REG ICR.I8 | |
| I9 | | 1 (8) | REG ICR.I9 | |
| I10 | | 1 (9) | REG ICR.I10 | |
| IDMACOM | 命令 DMA コマンド | 16 | REG IDMACOM | |
| ADDR | | 9(7,8,9,10,11,12,13,14,15) | REG IDMACOM.ADDR | |
| BC | | 7 (0,1,2,3,4,5,6) | REG IDMACOM.BC | |
| IDMASTS | 命令 DMA 状態 | 16 | REG IDMASTS | |
| BI | | 1 (15) | REG IDMASTS.BI | |
| IMR | 割り込みマスク | 16 | REG IMR | |
| IM1 | | 1 (0) | REG IMR.IM1 | |
| IM2 | | 1 (1) | REG IMR.IM2 | |
| IM3 | | 1 (2) | REG IMR.IM3 | |
| IM4 | | 1 (3) | REG IMR.IM4 | |

保守／廃止

HSM77016 **のレジスタ変数**

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| IM5 | **割り込みマスク** | 1 (4) | REG IMR.IM5 | |
| IM6 | | 1 (5) | REG IMR.IM6 | |
| IM7 | | 1 (6) | REG IMR.IM7 | |
| IM8 | | 1 (7) | REG IMR.IM8 | |
| IM9 | | 1 (8) | REG IMR.IM9 | |
| IM10 | | 1 (9) | REG IMR.IM10 | |
| IM11 | | 1 (10) | REG IMR.IM11 | |
| IM12 | | 1 (11) | REG IMR.IM12 | |
| IP | **命令ポインタ** | 16 | [REG] IP | |
| ISR | **割り込み状態** | 16 | REG ISR | |
| IS1 | | 1 (0) | REG ISR.IS1 | |
| IS2 | | 1 (1) | REG ISR.IS2 | |
| IS3 | | 1 (2) | REG ISR.IS3 | |
| IS4 | | 1 (3) | REG ISR.IS4 | |
| IS5 | | 1 (4) | REG ISR.IS5 | |
| IS6 | | 1 (5) | REG ISR.IS6 | |
| IS7 | | 1 (6) | REG ISR.IS7 | |
| IS8 | | 1 (7) | REG ISR.IS8 | |
| IS9 | | 1 (8) | REG ISR.IS9 | |
| IS10 | | 1 (9) | REG ISR.IS10 | |
| IS11 | | 1 (10) | REG ISR.IS11 | |
| IS12 | | 1 (11) | REG ISR.IS12 | |
| IWTR | **命令ウエイト・サイクル** | 16 | REG IWTR | 00=**ウエイト・サイクルなし**<br>01=1 **ウエイト・サイクル**<br>10=3 **ウエイト・サイクル**<br>11=7 **ウエイト・サイクル** |
| B | **命令ウエイト・サイクル領域** | 2 (2,3) | REG IWTR.B | |
| C | | 2 (4,5) | REG IWTR.C | |
| D | | 2 (6,7) | REG IWTR.D | |
| IA | | 4 (0,1,2,3) | REG IWTR.IA | $\mu$PD77116 **のみ** |
| IB | | 4 (4,5,6,7) | REG IWTR.IB | |
| IC | | 4 (8,9,10,11) | REG IWTR.IC | |
| ID | | 4 (12,13,14,15) | REG IWTR.ID | |
| LC | **ループ・カウンタ** | 16 | REG LC | |
| LOOP | **ループ・フラグ** | 1 (15) | REG LC.LOOP | 0=**ループ内部**<br>1=**ループ外部** |
| LEA | **ループ終了アドレス** | 16 | [REG] LEA | |
| LSA | **ループ開始アドレス** | 16 | [REG] LSA | |
| LSP | **ループ・スタック・ポインタ** | 16 | [REG] LSP | |
| LSR1 | LSA **プッシュ／ポップ** | 16 | [REG] LSR1 | |
| LSR11 | LSA **スタック** | | [REG] LSR11 | |
| LSR12 | | | [REG] LSR12 | |
| LSR13 | | | [REG] LSR13 | |
| LSR14 | | | [REG] LSR14 | |

保守／廃止

HSM77016 **のレジスタ変数**

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| LSR2 | LEA **プッシュ／ポップ** | 16 | [REG] LSR2 | |
| LSR21 | LEA **スタック** | | [REG] LSR21 | |
| LSR22 | | | [REG] LSR22 | |
| LSR23 | | | [REG] LSR23 | |
| LSR24 | | | [REG] LSR24 | |
| LSR3 | LC **プッシュ／ポップ** | 16 | [REG] LSR3 | |
| LSR31 | LC **スタック** | | [REG] LSR31 | |
| LSR32 | | | [REG] LSR32 | |
| LSR33 | | | [REG] LSR33 | |
| LSR34 | | | [REG] LSR34 | |
| MISR | **メモリ割り込み状態** | 16 | REG MISR | |
| DAC | | 1 (2) | REG MISR.DAC | |
| EDAC | | 1 (10) | REG MISR.EDAC | |
| EIAC | | 1 (9) | REG MISR.EIAC | |
| EMHC | | 1 (8) | REG MISR.EMHC | |
| IAC | | 1 (1) | REG MISR.IAC | |
| MHC | | 1 (0) | REG MISR.MHC | |
| PBCNTL1 | **周辺バッファ・コントロール** | 16 | REG PBCNTL1 | |
| IBM | 1 | 1 (3) | REG PBCNTL1.IBM | |
| EIN | | 1 (1) | REG PBCNTL1.IEN | |
| IIEN | | 1 (0) | REG PBCNTL1.IIEN | |
| ISEL | | 1 (2) | REG PBCNTL1.ISEL | |
| OBM | | 1 (7) | REG PBCNTL1.OBM | |
| OEN | | 1 (5) | REG PBCNTL1.OEN | |
| OIEN | | 1 (4) | REG PBCNTL1.OIEN | |
| OSEL | | 1 (6) | REG PBCNTL1.OSEL | |
| PSEL | | 2 (8,9) | REG PBCNTL1.PSEL | |
| PBCNTL2 | **周辺バッファ・コントロール** | 16 | REG PBCNTL2 | |
| IBM | 2 | 1 (3) | REG PBCNTL2.IBM | |
| EIN | | 1 (1) | REG PBCNTL2.IEN | |
| IIEN | | 1 (0) | REG PBCNTL2.IIEN | |
| ISEL | | 1 (2) | REG PBCNTL2.ISEL | |
| OBM | | 1 (7) | REG PBCNTL2.OBM | |
| OEN | | 1 (5) | REG PBCNTL2.OEN | |
| OIEN | | 1 (4) | REG PBCNTL2.OIEN | |
| OSEL | | 1 (6) | REG PBCNTL2.OSEL | |
| PSEL | | 2 (8,9) | REG PBCNTL2.PSEL | |
| PBIAC1 | **周辺バッファ入力アクセ** | 16 | REG PBIAC1 | |
| PBIAC2 | **ス・カウンタ値** 1&2 | | REG PBIAC2 | |
| PBIACI1 | **周辺バッファ出力アクセ** | 16 | REG PBIACI1 | |
| PBIACI2 | **ス・カウンタ初期値** 1&2 | | REG PBIACI2 | |
| PBIDP1 | **周辺バッファ入力データ・** | 16 | REG PBIDP1 | |
| PBIDP2 | **ポインタ** 1&2 | | REG PBIDP2 | |

保守／廃止

HSM77016 **のレジスタ変数**

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| PBISTP1<br>PBISTP2 | **周辺バッファ入力ストップ・アドレス** 1&2 | 16 | REG PBISTP1<br>REG PBISTP2 | |
| PBISTR1<br>PBISTR2 | **周辺バッファ入力開始アドレス** 1&2 | 16 | REG PBISTR1<br>REG PBISTR2 | |
| PBOAC1<br>PBOAC2 | **周辺バッファ出力アクセス・カウンタ値** 1&2 | 16 | REG PBOAC1<br>REG PBOAC2 | |
| PBOACI1<br>PBOACI2 | **周辺バッファ出力アクセス・カウンタ初期値** 1&2 | 16 | REG PBOACI1<br>REG PBOACI2 | |
| PBODP1<br>PBODP2 | **周辺バッファ出力データ・ポインタ** 1&2 | 16 | REG PBODP1<br>REG PBODP2 | |
| PBOSTP1<br>PBOSTP2 | **周辺バッファ出力ストップ・アドレス** 1&2 | 16 | REG PBOSTP1<br>REG PBOSTP2 | |
| PBOSTR1<br>PBOSTR2 | **周辺バッファ出力開始アドレス** 1&2 | 16 | REG PBOSTR1<br>REG PBOSTR2 | |
| PCD | **ポート・コマンド** | 16 | REG PCD | |
| B | | 2 (8,9) | REG PCD.B | $\mu$PD77116 **のみ** |
| B0 | **ビット操作** | 1 (8) | REG PCD.B0 | B1 B0 |
| B1 | **出力ポート** | 1 (9) | REG PCD.B1 | 00　= P0<br>01　= P1<br>10　= P2<br>11　= P3 |
| BE | **ビット操作イネーブル** | 1 (15) | REG PCD.BE | 0=**ディスエーブル**<br>1=**イネーブル** |
| IO | **入出力** | 1 (12) | REG PCD.IO | 0=**入力**<br>1=**出力** |
| M0 | **モード選択ポート** | 1 (0) | REG PCD.M0 | 0=P0 **選択解除** |
| M1 | | 1 (1) | REG PCD.M1 | 1=P0 **選択** |
| M2 | | 1 (2) | REG PCD.M2 | |
| M3 | | 1 (3) | REG PCD.M3 | |
| ME | **モード設定イネーブル** | 1 (13) | REG PCD.ME | 0=**ディスエーブル**<br>1=**イネーブル** |
| PSR | **ポート・セット／リセット** | 1 (14) | y | y<br>1=**設定** |
| PDR | **ポート・データ・レジスタ** | 16 | REG PDR | |
| PDT | **ポート・データ・レジスタ** | 16 | REG PDT | |
| PDTIN | **ポート・データ入力** | 16 | REG PDTIN | |
| PDTOUT | **ポート・データ出力** | 16 | REG PDTOUT | |
| R0 | **汎用レジスタ** | 40 | [REG] R0 | |
| R0E | **汎用レジスタ** R0 **の一部** | 8 | [REG] R0E | R0 **のビット** 32-39 |
| R0EH | | 24 | [REG] R0EH | R0 **のビット** 16-39 |
| R0H | | 16 | [REG] R0H | R0 **のビット** 16-31 |
| R0HL | | 32 | [REG] R0HL | R0 **のビット** 0-31 |
| R0L | | 16 | [REG] R0L | R0 **のビット** 0-15 |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| R1 | 汎用レジスタ | 40 | [REG] R1 | |
| R1E | 汎用レジスタ R1 の一部 | 8 | [REG] R1E | R1 のビット 32-39 |
| R1EH | | 24 | [REG] R1EH | R1 のビット 16-39 |
| R1H | | 16 | [REG] R1H | R1 のビット 16-31 |
| R1HL | | 32 | [REG] R1HL | R1 のビット 0-31 |
| R1L | | 16 | [REG] R1L | R1 のビット 0-15 |
| R2 | 汎用レジスタ | 40 | [REG] R2 | |
| R2E | 汎用レジスタ R2 の一部 | 8 | [REG] R2E | R2 のビット 32-39 |
| R2EH | | 24 | [REG] R2EH | R2 のビット 16-39 |
| R2H | | 16 | [REG] R2H | R2 のビット 16-31 |
| R2HL | | 32 | [REG] R2HL | R2 のビット 0-31 |
| R2L | | 16 | [REG] R2L | R2 のビット 0-15 |
| R3 | 汎用レジスタ | 40 | [REG] R3 | |
| R3E | 汎用レジスタ R3 の一部 | 8 | [REG] R3E | R3 のビット 32-39 |
| R3EH | | 24 | [REG] R3EH | R3 のビット 16-39 |
| R3H | | 16 | [REG] R3H | R3 のビット 16-31 |
| R3HL | | 32 | [REG] R3HL | R3 のビット 0-31 |
| R3L | | 16 | [REG] R3L | R3 のビット 0-15 |
| R4 | 汎用レジスタ | 40 | [REG] R4 | |
| R4E | 汎用レジスタ R4 の一部 | 8 | [REG] R4E | R4 のビット 32-39 |
| R4EH | | 24 | [REG] R4EH | R4 のビット 16-39 |
| R4H | | 16 | [REG] R4H | R4 のビット 16-31 |
| R4HL | | 32 | [REG] R4HL | R4 のビット 0-31 |
| R4L | | 16 | [REG] R4L | R4 のビット 0-15 |
| R5 | 汎用レジスタ | 40 | [REG] R5 | |
| R5E | 汎用レジスタ R5 の一部 | 8 | [REG] R5E | R5 のビット 32-39 |
| R5EH | | 24 | [REG] R5EH | R5 のビット 16-39 |
| R5H | | 16 | [REG] R5H | R5 のビット 16-31 |
| R5HL | | 32 | [REG] R5HL | R5 のビット 0-31 |
| R5L | | 16 | [REG] R5L | R5 のビット 0-15 |
| R6 | 汎用レジスタ | 40 | [REG] R6 | |
| R6E | 汎用レジスタ R6 の一部 | 8 | [REG] R6E | R6 のビット 32-39 |
| R6EH | | 24 | [REG] R6EH | R6 ビット 16-39 |
| R6H | | 16 | [REG] R6H | R6 ビット 16-31 |
| R6HL | | 32 | [REG] R6HL | R6 ビット 0-31 |
| R6L | | 16 | [REG] R6L | R6 ビット 0-15 |
| R7 | 汎用レジスタ | 40 | [REG] R7 | |
| R7E | 汎用レジスタ R7 の一部 | 8 | [REG] R7E | R7 のビット 32-39 |
| R7EH | | 24 | [REG] R7EH | R7 ビット 16-39 |
| R7H | | 16 | [REG] R7H | R7 ビット 16-31 |
| R7HL | | 32 | [REG] R7HL | R7 ビット 0-31 |
| R7L | | 16 | [REG] R7L | R7 ビット 0-15 |

保守／廃止

HSM77016 **のレジスタ変数**

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| RC | リピート・カウンタ | 16 | [REG] RC | |
| REP | リピート・フラグ | 1 (15) | [REG] RC.REP | 0=イン・リピート<br>1=アウト・オブ・リピート |
| SDT1IN | シリアル・データ・イン | 16 | REG SDT1IN | |
| SDT1OUT | シリアル・データ・アウト | 16 | REG SDT1OUT | |
| SDT2IN | シリアル・データ・イン | 16 | REG SDT2IN | |
| SDT2OUT | シリアル・データ・アウト | 16 | REG SDT2OUT | |
| SP | スタック・ポインタ | 16 | [REG] SP | |
| SPCD1 | シリアル・ポート・コマンド・データ 1 | 16 | | |
| B | | 2 (8,9) | REG SPCD1.B | |
| BE | | 1 (15) | REG SPCD1.BE | |
| IO | | 1 (12) | REG SPCD1.IO | |
| M0 | | 1 (0) | REG SPCD1.M0 | |
| M1 | | 1 (1) | REG SPCD1.M1 | |
| M2 | | 1 (2) | REG SPCD1.M2 | |
| M3 | | 1 (3) | REG SPCD1.M3 | |
| ME | | 1 (13) | REG SPCD1.ME | |
| PSR | | 1 (14) | REG SPCD1.PSR | |
| SPCD2 | シリアル・ポート・コマンド・データ 2 | 16 | | |
| B | | 2 (8,9) | REG SPCD2.B | |
| BE | | 1 (15) | REG SPCD2.BE | |
| IO | | 1 (12) | REG SPCD2.IO | |
| M0 | | 1 (0) | REG SPCD2.M0 | |
| M1 | | 1 (1) | REG SPCD2.M1 | |
| M2 | | 1 (2) | REG SPCD2.M2 | |
| M3 | | 1 (3) | REG SPCD2.M3 | |
| ME | | 1 (13) | REG SPCD2.ME | |
| PSR | | 1 (14) | REG SPCD2.PSR | |
| SPDR1 | シリアル・ポート・データ | 16 | REG SPDR1 | |
| SPDR2 | | 16 | REG SPDR2 | |
| SPDT1IN | シリアル・ポート・データ入力 | 16 | REG SPDT1IN | |
| SPDT2IN | | 16 | REG SPDT2IN | |
| SPDT1OUT | シリアル・ポート・データ出力 | 16 | REG SPDT1OUT | |
| SPDT2OUT | | 16 | REG SPDT2OUT | |
| SR | ステータス・レジスタ | 16 | [REG] SR | |
| EB | | 1 (13) | [REG] SR.EB | $\mu$PD77016 のみ |
| EI | | 1 (15) | [REG] SR.EI | |
| EP | | 1 (14) | [REG] SR.EP | |
| HI | ホスト入力割り込みマスク | 1 (8) | [REG] SR.HI | 0=イネーブル |
| HO | ホスト出力割り込みマスク | 1 (9) | [REG] SR.HO | 1=ディスエーブル |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| INT1 | 外部割り込みマスク・ | 1 (0) | [REG] SR.INT1 | 0=イネーブル |
| INT2 | フラグ | 1 (1) | [REG] SR.INT2 | 1=ディスエーブル |
| INT3 | | 1 (2) | [REG] SR.INT3 | |
| INT4 | | 1 (3) | [REG] SR.INT4 | |
| INT1 | | 1 (4) | [REG] SR.INT1 | μPD77116 のみ |
| INT2 | | 1 (1) | [REG] SR.INT2 | |
| INT3 | | 1 (6) | [REG] SR.INT3 | |
| INT4 | | 1 (7) | [REG] SR.INT4 | |
| INT5 | | 1 (8) | [REG] SR.INT5 | |
| INT6 | | 1 (9) | [REG] SR.INT6 | |
| INT7 | | 1 (10) | [REG] SR.INT7 | |
| INT8 | | 1 (11) | [REG] SR.INT8 | |
| INT9 | | 1 (12) | [REG] SR.INT9 | |
| INT10 | | 1 (13) | [REG] SR.INT10 | |
| INT11 | | 1 (14) | [REG] SR.INT11 | |
| INT12 | | 1 (15) | [REG] SR.INT12 | |
| LF | ループ・フラグ | 1 (12) | [REG] SRLF | 0=ループ内部<br>1=ループ外部 |
| SI1 | シリアル入力割り込み | 1 (4) | [REG] SR.SI1 | 0=イネーブル |
| SI2 | マスク | 1 (6) | [REG] SR.SI2 | 1=ディスエーブル |
| SO1 | シリアル出力割り込み | 1 (5) | [REG] SR.SO1 | |
| SO2 | マスク | 1 (7) | [REG] SR.SO2 | |
| SST1 | シリアル・ステータス・レジスタ | 16 | REG SST1 | |
| SIBL | シリアル入力ビット長 | 1 (12) | REG SST1.SIBL | 0=16 ビット<br>1=8 ビット |
| SICM | シリアル入力・ホールド要求 | 1 (9) | REG SST1.SICM | 0=ホールド<br>1=通常 |
| SIEF | SIAK 出力イネーブル | 1 (8) | REG SST1.SIEF | 0=ディスエーブル<br>1=イネーブル |
| SITF | シリアル入力伝送 | 1 (14) | REG SST1.SITF | 0=MSB ファースト<br>1=LSB ファースト |
| SLEF | SDT ロード・イネーブル | 1 (0) | REG SST1.SLEF | 0=ディスエーブル<br>1=イネーブル |
| SLER | SDT ロード・エラー | 1 (2) | REG SST1.SLER | 0=エラーなし<br>1=エラー |
| SLWE | SDT ロード・ウエイト・イネーブル | 1 (10) | REG SST1.SLWE | 0=ディスエーブル<br>1=イネーブル |
| SOBL | シリアル出力ビット長 | 1 (13) | REG SST1.SOBL | 0=16 ビット<br>1=8 ビット |
| SOTF | シリアル出力伝送 | 1 (15) | REG SST1.SOTF | 0=MSB ファースト<br>1=LSB ファースト |
| SSEF | SDT 格納イネーブル | 1 (1) | REG SST1.SSEF | 0=ディスエーブル<br>1=イネーブル |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| SSER | SDT 格納エラー | 1 (3) | REG SST1.SSER | 0=エラーなし<br>1=エラー |
| SSWE | SDT 格納ウエイト･イネーブル | 1 (11) | REG SST1.SSWE | 0=ディスエーブル<br>1=イネーブル |
| SST2 | シリアル・ステータス・レジスタ | 16 | REG SST2 | |
| SIBL | シリアル入力ビット長 | 1 (12) | REG SST2.SIBL | 0=16 ビット<br>1=8 ビット |
| SICM | シリアル入力ホールド要求 | 1 (9) | REG SST2.SICM | 0=ホールド<br>1=通常 |
| SIEF | SIAK 出力イネーブル | 1 (8) | REG SST2.SIEF | 0=ディスエーブル<br>1=イネーブル |
| SITF | シリアル入力伝送 | 1 (14) | REG SST2.SITF | 0=MSB ファースト<br>1=LSB ファースト |
| SLEF | SDT ロード・イネーブル | 1 (0) | REG SST2.SLEF | 0=ディスエーブル<br>1=イネーブル |
| SLER | SDT ロード・エラー | 1 (2) | REG SST2.SLER | 0=エラーなし<br>1=エラー |
| SLWE | SDT ロード・ウエイト・イネーブル | 1 (10) | REG SST2.SLWE | 0=ディスエーブル<br>1=イネーブル |
| SOBL | シリアル出力ビット長 | 1 (13) | REG SST2.SOBL | 0=16 ビット<br>1=8 ビット |
| SOTF | シリアル出力伝送 | 1 (15) | REG SST2.SOTF | 0=MSB ファースト<br>1=LSB ファースト |
| SSEF | SDT 格納イネーブル | 1 (1) | REG SST2.SSEF | 0=ディスエーブル<br>1=イネーブル |
| SSER | SDT 格納エラー | 1 (3) | REG SST2.SSER | 0=エラーなし<br>1=エラー |
| SSWE | SDT 格納ウエイト･イネーブル | 1 (11) | REG SST2.SSWE | 0=ディスエーブル<br>1=イネーブル |
| STK | スタック・レジスタの内容 | 16 | REG STK | |
| STK1 | スタック・レジスタ | | REG STK1 | |
| STK2 | | | REG STK2 | |
| STK3 | | | REG STK3 | |
| STK4 | | | REG STK4 | |
| STK5 | | | REG STK5 | |
| STK6 | | | REG STK6 | |
| STK7 | | | REG STK7 | |
| STK8 | | | REG STK8 | |
| STK9 | | | REG STK9 | |
| STK10 | | | REG STK10 | |
| STK11 | | | REG STK11 | |
| STK12 | | | REG STK12 | |
| STK13 | | | REG STK13 | |
| STK14 | | | REG STK14 | |
| STK15 | | | REG STK15 | |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| TCR1 | タイマ・カウント・レジスタ | 16 | REG TCR1 | |
| TCR2 | | | REG TCR2 | |
| TCSR1 | タイマ制御ステータス 1 | 16 | REG TCSR1 | |
| TCLKPS | | 3 (0,1,2) | REG TCSR1.TCLKPS | |
| TCLKSEL | | 2 (6,7) | REG TCSR1.TCLKSEL | |
| TEN | | 1 (8) | REG TCSR1.TEN | |
| TIE | | 1 (9) | REG TCSR1.TIE | |
| TUD | | 1 (10) | REG TCSR1.TUD | |
| TCSR2 | タイマ制御ステータス 2 | 16 | REG TCSR2 | |
| TCLKPS | | 3 (0,1,2) | REG TCSR2.TCLKPS | |
| TCLKSEL | | 2 (6,7) | REG TCSR2.TCLKSEL | |
| TEN | | 1 (8) | REG TCSR2.TEN | |
| TIE | | 1 (9) | REG TCSR2.TIE | |
| TUD | | 1 (10) | REG TCSR2.TUD | |
| TIR1 | タイマ初期化レジスタ 1 | 16 | REG TIR1 | |
| TIR2 | タイマ初期化レジスタ 2 | 16 | REG TIR2 | |
| TISR | タイマ割り込みステータス | 16 | REG TISR | |
| TI1 | | 1 (0) | REG TISR.TI1 | |
| TI2 | | 1 (1) | REG TISR.TI2 | |
| TSARSH | TSA 受信スロット上位ワード | 16 | REG TSARSH | |
| RS16 | | 1 (0) | REG TSARSH.RS16 | |
| RS17 | | 1 (1) | REG TSARSH.RS17 | |
| RS18 | | 1 (2) | REG TSARSH.RS18 | |
| RS19 | | 1 (3) | REG TSARSH.RS19 | |
| RS20 | | 1 (4) | REG TSARSH.RS20 | |
| RS21 | | 1 (5) | REG TSARSH.RS21 | |
| RS22 | | 1 (6) | REG TSARSH.RS22 | |
| RS23 | | 1 (7) | REG TSARSH.RS23 | |
| RS24 | | 1 (8) | REG TSARSH.RS24 | |
| RS25 | | 1 (9) | REG TSARSH.RS25 | |
| RS26 | | 1 (10) | REG TSARSH.RS26 | |
| RS27 | | 1 (11) | REG TSARSH.RS27 | |
| RS28 | | 1 (12) | REG TSARSH.RS28 | |
| RS29 | | 1 (13) | REG TSARSH.RS29 | |
| RS30 | | 1 (14) | REG TSARSH.RS30 | |
| RS31 | | 1 (15) | REG TSARSH.RS31 | |
| TSARSL | TSA 受信スロット下位ワード | 16 | REG TSARSL | |
| RS0 | | 1 (0) | REG TSARSL.RS0 | |
| RS1 | | 1 (1) | REG TSARSL.RS1 | |
| RS2 | | 1 (2) | REG TSARSL.RS2 | |
| RS3 | | 1 (3) | REG TSARSL.RS3 | |
| RS4 | | 1 (4) | REG TSARSL.RS4 | |
| RS5 | | 1 (5) | REG TSARSL.RS5 | |

保守／廃止

HSM77016 のレジスタ変数

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| RS6 | TSA 受信スロット下位ワード | 1 (6) | REG TSARSL.RS6 | |
| RS7 | | 1 (7) | REG TSARSL.RS7 | |
| RS8 | | 1 (8) | REG TSARSL.RS8 | |
| RS9 | | 1 (9) | REG TSARSL.RS9 | |
| RS10 | | 1 (10) | REG TSARSL.RS10 | |
| RS11 | | 1 (11) | REG TSARSL.RS11 | |
| RS12 | | 1 (12) | REG TSARSL.RS12 | |
| RS13 | | 1 (13) | REG TSARSL.RS13 | |
| RS14 | | 1 (14) | REG TSARSL.RS14 | |
| RS15 | | 1 (15) | REG TSARSL.RS15 | |
| TSATSH | TSA 送信スロット上位ワード | 16 | REG TSATSH | |
| TS16 | | 1 (0) | REG TSATSH.TS16 | |
| TS17 | | 1 (1) | REG TSATSH.TS17 | |
| TS18 | | 1 (2) | REG TSATSH.TS18 | |
| TS19 | | 1 (3) | REG TSATSH.TS19 | |
| TS20 | | 1 (4) | REG TSATSH.TS20 | |
| TS21 | | 1 (5) | REG TSATSH.TS21 | |
| TS22 | | 1 (6) | REG TSATSH.TS22 | |
| TS23 | | 1 (7) | REG TSATSH.TS23 | |
| TS24 | | 1 (8) | REG TSATSH.TS24 | |
| TS25 | | 1 (9) | REG TSATSH.TS25 | |
| TS26 | | 1 (10) | REG TSATSH.TS26 | |
| TS27 | | 1 (11) | REG TSATSH.TS27 | |
| TS28 | | 1 (12) | REG TSATSH.TS28 | |
| TS29 | | 1 (13) | REG TSATSH.TS29 | |
| TS30 | | 1 (14) | REG TSATSH.TS30 | |
| TS31 | | 1 (15) | REG TSATSH.TS31 | |
| TSATSL | TSA 送信スロット下位ワード | 16 | REG TSATSL | |
| TS0 | | 1 (0) | REG TSATSL.TS0 | |
| TS1 | | 1 (1) | REG TSATSL.TS1 | |
| TS2 | | 1 (2) | REG TSATSL.TS2 | |
| TS3 | | 1 (3) | REG TSATSL.TS3 | |
| TS4 | | 1 (4) | REG TSATSL.TS4 | |
| TS5 | | 1 (5) | REG TSATSL.TS5 | |
| TS6 | | 1 (6) | REG TSATSL.TS6 | |
| TS7 | | 1 (7) | REG TSATSL.TS7 | |
| TS8 | | 1 (8) | REG TSATSL.TS8 | |
| TS9 | | 1 (9) | REG TSATSL.TS9 | |
| TS10 | | 1 (10) | REG TSATSL.TS10 | |
| TS11 | | 1 (11) | REG TSATSL.TS11 | |
| TS12 | | 1 (12) | REG TSATSL.TS12 | |
| TS13 | | 1 (13) | REG TSATSL.TS13 | |
| TS14 | | 1 (14) | REG TSATSL.TS14 | |
| TS15 | | 1 (15) | REG TSATSL.TS15 | |

保守／廃止

HSM77016 **のレジスタ変数**

| 略　　号 | 名　　称 | サイズ（ビット数） | 構　　文 | 備　　考 |
|---|---|---|---|---|
| TSCR | TSA **コントロール・レジスタ** | 16 | REG TSCR | |
| MODE | | 1 (8) | REG TSCR.MODE | |
| MSTPS | | 3 (0,1,2) | REG TSCR.MSTPS | |
| SCKMD | | 1 (9) | REG TSCR.SCKMD | |
| SCKPS | | 2 (4,5) | REG TSCR.SCKPS | |
| TSAE | | 1 (10) | REG TSCR.TSAE | |
| TSSR | TSA **ステータス・レジスタ** | 16 | REG TSSR | |
| ERRRX | | 1 (2) | REG TSSR.ERRRX | |
| ERRTX | | 1 (6) | REG TSSR.ERRTX | |
| FSTRX | | 1 (0) | REG TSSR.FSTRX | |
| FSTTX | | 1 (4) | REG TSSR.FSTTX | |
| RSTERR | | 1 (2) | REG TSSR.RSTERR | |
| RSTERT | | 1 (6) | REG TSSR.RSTERT | |
| UPDRX | | 1 (0) | REG TSSR.UPDRX | |
| UPDTX | | 1 (4) | REG TSSR.UPDTX | |
| VALRX | | 1 (2) | REG TSSR.VALRX | |
| VALTX | | 1 (6) | REG TSSR.VALTX | |
| TSSU | TSA **セットアップ・レジスタ** | 16 | REG TSSU | |
| FLRX | | 5 (0,1,2,3,4) | REG TSSU.FLRX | |
| FLTX | | 5 (8,9,10,11,12) | REG TSSU.FLTX | |
| SIBLRX | | 1 (5) | REG TSSU.SIBLRX | |
| SOBLTX | | 1 (13) | REG TSSU.SOBLTX | |
| XAR | **ディバグ・レジスタ** | 16 | REG XAR | |
| XWTR | X-DMA **メモリ・ウエイト・** | 16 | REG XWTR | |
| DA | **サイクル・コントロール** | 4 (0,1,2,3) | REG XWTR.DA | |
| DB | | 4 (4,5,6,7) | REG XWTR.DB | |
| DC | | 4 (8,9,10,11) | REG XWTR.DC | |
| DD | | 4 (12,13,14,15) | REG XWTR.DD | |
| YAR | **ディバグ・レジスタ** | 16 | REG YAR | |
| YWTR | Y-DMA **メモリ・ウエイト・** | 16 | REG YWTR | |
| DA | **サイクル・コントロール** | 4 (0,1,2,3) | REG YWTR.DA | |
| DB | | 4 (4,5,6,7) | REG YWTR.DB | |
| DC | | 4 (8,9,10,11) | REG YWTR.DC | |
| DD | | 4 (12,13,14,15) | REG YWTR.DD | |

保守／廃止

## 11.7.1 時間およびカウント測定変数例

次の表は，命令サイクル = 30 ns のときの時間およびカウント測定変数例です。

表 11－1　時間およびカウント測定変数例

| 命　　令 | r0=r0+1 | JMP | COND JMP |
|---|---|---|---|
| CYCLE | 1 | 2 | 3 |
| STEP | 1 | 1 | 1 |
| TIME | 30 | 60 | 90 |

HSM77016 の時間およびカウント測定変数

| 略　　号 | サイズ（ビット数） | 構　　文 | 説　　明 |
|---|---|---|---|
| CYCLE | 64 | CYCLE | シミュレートした CLK サイクル・カウント |
| ?DDMA | | | |
| LENGTH | 16 | ?DDMA. LENGTH | 現在の DMA 動作で転送されるワードの総数 |
| STATUS | 16 | ?DDMA.Status | 現在の DMA 動作で転送されるワードの残数 |
| ?IDMA | | | |
| LENGTH | 16 | ?IDMA.LENGTH | 現在の DMA 動作で転送されるワードの総数 |
| QLENGTH | 16 | ?IDMA.QLENGTH | 待機 DMA 動作で転送されるワードの総数 |
| QSTART | 16 | ?IDMA.QSTART | 待機 DMA 動作の開始アドレス |
| START | 16 | ?IDMA.START | 現在の DMA 動作の開始アドレス |
| STATUS | 16 | ?IDMA.STATUS | 現在の DMA 動作で転送されるワードの残数 |
| TYPE | 16 | ?IDMA.TYPE | 現在の DMA 動作のタイプ |
| STEP | | STEP | シミュレートした命令カウント |
| ?ICACHE | | | |
| TAG0 | 16 | ?ICACHE.TAG0 | キャッシュ・メモリ・ブロック No.0 の基底アドレス |
| TAG1 | 16 | ?ICACHE.TAG1 | キャッシュ・メモリ・ブロック No.1 の基底アドレス |
| TAG2 | 16 | ?ICACHE.TAG2 | キャッシュ・メモリ・ブロック No.2 の基底アドレス |
| TAG3 | 16 | ?ICACHE.TAG3 | キャッシュ・メモリ・ブロック No.3 の基底アドレス |
| TAG4 | 16 | ?ICACHE.TAG4 | キャッシュ・メモリ・ブロック No.4 の基底アドレス |
| TAG5 | 16 | ?ICACHE.TAG5 | キャッシュ・メモリ・ブロック No.5 の基底アドレス |
| TAG6 | 16 | ?ICACHE.TAG6 | キャッシュ・メモリ・ブロック No.6 の基底アドレス |
| TAG7 | 16 | ?ICACHE.TAG7 | キャッシュ・メモリ・ブロック No.7 の基底アドレス |
| TAG8 | 16 | ?ICACHE.TAG8 | キャッシュ・メモリ・ブロック No.8 の基底アドレス |
| TAG9 | 16 | ?ICACHE.TAG9 | キャッシュ・メモリ・ブロック No.9 の基底アドレス |
| TAG10 | 16 | ?ICACHE.TAG10 | キャッシュ・メモリ・ブロック No.10 の基底アドレス |
| TAG11 | 16 | ?ICACHE.TAG11 | キャッシュ・メモリ・ブロック No.11 の基底アドレス |
| TAG12 | 16 | ?ICACHE.TAG12 | キャッシュ・メモリ・ブロック No.12 の基底アドレス |
| TAG13 | 16 | ?ICACHE.TAG13 | キャッシュ・メモリ・ブロック No.13 の基底アドレス |
| TAG14 | 16 | ?ICACHE.TAG14 | キャッシュ・メモリ・ブロック No.14 の基底アドレス |
| TAG15 | 16 | ?ICACHE.TAG15 | キャッシュ・メモリ・ブロック No.15 の基底アドレス |
| TIME | 64 | TIME | シミュレートした時間 |
| TIME_RESOLUTION | 64 | TIME_RESOLUTION | シミュレートした時間（1 秒あたり）の分解能 |

〔メ　モ〕

保守／廃止

# 付録A HSM77016 のキー

HSM77016 のメニュー・コマンドにアクセスする場合は，Alt キーとメニュー・コマンドに下線がついている文字のキーを押します。コマンドの中にはキーボードのショートカットが用意されているものがあり，それぞれ該当するメニュー・コマンドの右に表示されています。

### コマンド・ナビゲーション・キー（Scroll Lock ディスエーブル）

| キー | アンカの移動先 |
|---|---|
| PageUp | 画面を上に動かす（前の画面の同じ水平位置へ） |
| PageDown | 画面を下に動かす（次の画面の同じ水平位置へ） |
| Ctrl＋PageUp | 画面の始まり／画面を左に動かす（画面の最上部；または 1 画面左に動かす） |
| Ctrl＋PageDown | 画面の終わり／画面を右に動かす（画面の最下部；または 1 画面右に動かす） |
| Home | 行の始まり（現在行の左端位置[注]） |
| End | 行の終わり（現在行の右端位置[注]） |
| Ctrl＋Home | データの始まり（現在のフィールドまたは文書の左最上部） |
| Ctrl＋End | データの終わり（現在のフィールドまたは文書の右最下部） |

**注** Register, Breakpoint および Watch ウインドウ

### 共通ナビゲーション・キー（Scroll Lock イネーブル）

| キー | アンカの移動先 |
|---|---|
| PageUp | 画面を上に動かす（前の画面の同じ水平位置へ） |
| PageDown | 画面を下に動かす（次の画面の同じ水平位置へ） |

### 編集バー

| キー | アンカの移動先 | 機　能 |
|---|---|---|
| Ctrl＋0 | Zero ボタン | Zero ボタンの動作 |

### Breakpoint ウインドウで使用するキー

| キー | 機　能 |
|---|---|
| Insert | 新たなブレークポイントを追加します。 |
| Delete | ブレークポイント，選択された条件または操作を削除します。 |

### Watch ウインドウで使用するキー

| キー | 機　能 |
|---|---|
| Insert | 新たなウオッチを追加します。 |
| Delete | 選択されたウオッチを削除します。 |

### Statistic ウインドウで使用するキー

| キー | 機　能 |
|---|---|
| Delete | 選択されたマーカを削除します。 |

### メモリ，レジスタ，またはテキスト・ビューアの各ウインドウで使用するキー

| キー | 機　能 |
|---|---|
| Ctrl＋G | 指定のメモリ・アドレス，レジスタ，または行番号に移動します |

### Edit メニューのコマンド

| キー | メニュー項目 | 機　能 |
|---|---|---|
| Ctrl＋Z | Undo | ユーザのデータを変更した最後の操作を取り消します。 |
| Ctrl＋X | Cut | 選択されたデータを切り取り，クリップ・ボードにコピーします。 |
| Ctrl＋C | Copy | 選択されたデータをクリップ・ボードにコピーします。 |
| Ctrl＋V | Paste | データをクリップ・ボードから貼り付けます。 |

### Run メニューのコマンド

| キー | メニュー項目 | 機　能 |
|---|---|---|
| F9 | Run | シミュレーションを開始します。 |
| Ctrl＋F9 | Break | シミュレーションをブレークします。 |
| Ctrl＋F7 | Instruction Trace | 1 命令を実行します。 |
| F7 | Trace | サブルーチン・エントリで 1 ステップをシミュレートします。 |
| F8 | Step | サブルーチン・エントリなしで 1 ステップをシミュレートします。 |
| Ctrl＋Shift＋F7 | Animate \| Instruction Trace | 命令トレース・モードを実行します。 |
| Shift＋F7 | Animate \| Trace | トレース・モードを実行します。 |
| Shift＋F8 | Animate \| Step | ステップ・モードを実行します。 |
| F4 | To Cursor | カーソル・アドレスに達するまでシミュレーションを実行します。 |
| Ctrl＋F2 | Reset | シミュレートされたターゲット・プロセッサをリセットします。 |
| Alt＋F7 | Back Trace | 1 ソース行または命令を逆方向に実行します。 |

### ヘルプ・コマンド

| キー | 機　能 |
|---|---|
| F1 | HSM77016 の Help を表示します。 |
| Shift＋F1 | HSM77016 ユーザ・インタフェースの一部に関するヘルプを表示します。必要なオブジェクトをクリックします。ヘルプが使用できない場合は，HSM77016 ヘルプの内容が表示されます。 |
| Ctrl＋F1 | μPD77016 のヘルプを表示します。 |
| Shift＋Ctrl＋F1 | μPD77016 項目に関するヘルプを表示します。必要な項目をクリックします。ヘルプが使用できない場合は，オンライン・マニュアル・ヘルプの内容が表示されます。 |

保守／廃止

★ # 付録B　リリース・ノート

## B.1　バージョン 2.32

### (1) リリース日時

1999 年 6 月 28 日

### (2) バージョン 2.31 以降のバグ・フィックス

**(a) SPR No.238：REP 命令実行中に割り込みが発生する**

**影響を受ける製品：**HSM77016 2.31

**問題：次の例の場合，HSM77016 は，マークしている位置（REP 0x7fff）で割り込み処理を開始します。**

```
REP 0x7fff;NOP;REP 0x7fff;     <--- interrupt may occurNOP;REP 0x7fff;     <--- interrupt may occurNOP;...
```

**状況：この問題はフィックスされています。**

**(b) グラフィック・タイミング出力のタイミング・ファイル変数がコンテキストから外れる**

**問題：タイミング・ファイルが再ロードされたあとで，タイミング・ファイル変数のグラフィック・タイミング出力は，コンテキストから外れることがあります。グラフィック・タイミング出力表示を更新する場合は，ウォッチ式を再入力してください。**

**状況：この問題はフィックスされています。タイミング・ファイル変数は，タイミング・ファイルの再ロード後に正しく評価されます。**

## B.2　バージョン 2.31

### (1) リリース日時

1999 年 6 月 9 日

### (2) 強化と修正

- **リンク・ファイル・インポート**

  **インポートされるリンク・ファイル・エレメントの選択を行えるようにするダイアログを採用することで，リンク・ファイル・インポート機能を強化しています。**

- **グラフィック・タイミング出力**

  **タイム・マーカ表示を修正しています。現在のタイム・マーカがステップ，トレース，またはバック・トレースが実行される以前に表示領域内にある場合，ステップ，トレースまたはバック・トレース実行後も表示領域内に残ります。現在のタイム・マーカが表示領域外にある場合，表示の位置は，ステップ，トレースまたはバック・トレース実行後に最後に表示された位置になります。**

- **モデル・ファイル**

  μPD77110 **モデル・ファイル・クロック周波数を** 75MHz **から** 65MHz **に変更しました。**

- **サンプル・ファイル**

  **現行の** TSA **仕様に従って** TSA **サンプル・ファイルを修正しています。**TSA **サンプル・ファイル** 2 **と** 4 **は，現行の** TSA **仕様に合致していないために削除されています。**TSA **サンプル・ファイル** 3 **は** 2 **に改名されています。**7701x **サンプル・ファイルは，現行デバイス制約事項に従って修正されています。**

保守／廃止

### (3) バージョン 2.3 以降のバグ・フィックス

#### (a) SPR #230：外部インストラクション・メモリがない場合に割り込みが遅れる

影響を受ける製品：HSM77016 2.3

問題：外部インストラクション・メモリ空間のないモデルを使用した場合，JUMP，JPEG，CALL，CREG，RET，および RETI の各命令実行の終了時に割り込みは処理されず，次の順次命令の終了時に処理を開始します。しかし，次に続く順次命令がない場合（JMP$命令など）割り込みはずっと遅延する形になり，処理されません。

詳細：この問題は外部インストラクション・メモリ空間のないモデルを使用した場合に発生します。この問題は外部メモリが実際に備えられていないことではなく，外部インストラクション・メモリ拡張空間がないことによるものです。

外部インストラクション・メモリ空間がない場合，HSM77016 はより効率的な内部コーディングを使用します。しかし，このコードは割り込み要求を正しく処理できません。

状況：割り込み処理機能はフィックスされています。

#### (b) SPR #229: TSA ブロックの RFS または TFS 端子が入力状態であるにもかかわらず出力端子として認識される

影響を受ける製品：HSM77016 2.3 M2〜2.3

問題：TSA ブロックの RFS または TFS 端子は，入力端子であるにもかかわらず出力端子であることを示すことがあります。この問題は，次のイベントの場合に発生します。

a. TSA ブロックがマスタ・モードで構成されているときに（RFS と TFS は出力）
b. デバイスをリセットし，
c. TSA ブロックをスレーブ・モードで構成したとき（RFS と TFS は入力）

このときステップ c のあとで端子 RFS または TFS を変更しようとすると，“それらの端子は現在出力として正しく構成されており，変更できません” というエラー・メッセージが出されます。

状況：TSA ブロックの RFS と TFS 端子の問題はフィックスされています。

#### (c) SPR #228：TSA フレーム長が 32 のタイム・スロットに設定されている場合，すべてのタイム・スロットがインアクティブになる

影響を受ける製品：HSM77016 2.3

問題：TSA フレーム長が TSSU レジスタで 32 のタイム・スロットに設定されている場合，すべてのタイム・スロットは，構成が TSARS resp. TSATS レジスタに設定されていることに関係なくインアクティブになります。

状況：この問題はフィックスされています。

#### (d) Statistic ウインドウのサイズを調整するとウインドウのカラムが消える

問題：少なくとも 1 つのプロファイリング・マーカが設定されている状態でウインドウ幅を狭くして Statistic ウインドウを（再）オープンした場合，ウインドウを元のサイズに復元したとき Count，Time，Time/Count，および Notes の各カラムが消えることがあります。

状況：Statistic ウインドウ・カラムの表示はフィックスされています。

保守／廃止

## B.3 バージョン 2.3

### (1) リリース日時

1999 年 5 月 11 日

### (2) 強化と修正

- グラフィック・タイミング出力
- μPD77110, 77111, 77112, 77113, 77114, 77116 の各デバイスをフルサポート

### (3) バージョン 2.3 のベータ版 2 以降のバグ・フィックス

(a) SPR #220: Model Editor オンライン・ヘルプのトピック・タイトルが誤って表示される

影響を受ける製品：Model Editor （MDLDLG32.DLL），バージョン 2.3（ビルド 22），ヘルプ・データ・ベース・ビルド 1.4

問題：トピック“ Memory Selection Properties Dialog ”のトピック・タイトルに Memory Section Properties Dialog が読み出されません。

状況：ヘルプ・トピックと，対応するすべての参照事項を修正済みです。

(b) SPR #219：外部メモリ・ブロックに使用する DWTR レジスタのビットが使用できない

影響を受ける製品：HSM77016 2.3 ベータ版 2

問題：DWTR レジスタのビットのうち，ビット 15, 14, 7，および 6 以外のビットを設定することができず，そのためデータ・メモリの 0x8000-0xBFFF のウエイト・サイクルが指定できなくなっています。これは，特にμPD77110 外部データ・メモリ範囲に適用する問題です。

状況：外部メモリ・ブロック用の DWTR レジスタの各ビットはフィックスされています。

(c) SPR #217：新規のセッションを作成せずに既存のログ・セッションが上書きされる

影響を受ける製品：HSM77016 2.3 ベータ版～2.3 ベータ版 2

問題：アプリケーションは，各ログ・セッションを，オプション・ダイアログの Save log data to file 設定で指定されているファイルの新たなセッションに保存する必要がありますが，ここで新たなセッションを作成する代わりに常に同じセッションに書き込みを行ってしまいます。ログ・ファイルによっては，既存のログ・セッションへ上書きすることもしないこともありますが，“ 影響を受ける製品 ” でリストしているアプリケーションは 1 ファイル当たり 1 つのログしか持ちません。

状況：1 ログ・ファイル当たり個別のログ・セッションが作成されるよう修正済みです。

(d) SPR #214：端子 P0，P1，P2，および P3 の値がそれぞれ正しく更新されない

影響を受ける製品：HSM77016 － すべてのバージョン

問題：汎用 I/O 端子 P0，P1，P2，および P3 の値が，たとえばウオッチ式で使用されている状態で入力／出力モードが変更されたときなどに正しく更新されません。

状況：汎用 I/O 端子 P0，P1，P2，および P3 の値が正しく更新されるよう修正済みです。

**（e）**SPR #213**：**PDR **レジスタがリセット後に更新されない**

**影響を受ける製品：**HSM77016 2.3 **ベータ版** 2

**問題：リセット後，汎用レジスタ** I/O **ポート端子** P0，P1，P2，**および** P3 **は入力モードとなります。これらの端子の状態は** HSM77016 **の** PDR **レジスタに反映されます。しかしこのレジスタはリセット後にも更新されず，リセット前の値を示します。**

**状況：**PDR **レジスタはリセット後に正しく更新されるよう修正済みです。**

## B.4　バージョン 2.3 ベータ版 2

**（1）リリース日時**

1999 **年** 5 **月** 15 **日**

**（2）強化と修正**

**グラフィック・タイミング出力：グラフィック・タイミング出力が次の機能によって強化されています。**

- **グリッド機能**
- **グリッド・スナップ機能**
- **時間座標表示を含むタイム・カーソル，タイム・マーカ**
- **測定ツール**
- **浮動信号ツール・バー**
- **ユーザ・インタフェース・カスタマイズ機能**

**（3）バージョン** 2.3 **ベータ版以降のバグ・フィックス**

**（a）**SPR #207**：以前の汎用** I/O **ポート端子名が認識されない**

**影響を受ける製品：**HSM77016 2.3M1～2.3 **ベータ版**

**問題：汎用** I/O **ポート端子** PIO0，PIO1，PIO2，**および** PIO3 **の以前の端子名が** HSM77016 **に既知のシンボル名として受け入れられません。**

**状況：**HSM77016 **は，次の汎用** I/O **ポート端子の以前と現在とのシンボル名を受け入れるよう修正済みです。**PIO0，PIO1，PIO2，PIO3，**および** P0，P1，P2，P3。

**（b）**SPR #200**：**CreateMemRdEventCallback **および** CreateMemWrEventCallback API **関数が実行に失敗する**

**影響を受ける製品：**HSM77016 2.1 **ベータ版**～2.3 M2a

**問題：**X **または** Y **メモリ・ブロック（**0xFFFF **または** 0x1FFFF**）の最高位アドレスが関数パラメータに含まれている場合，**CreateMemRdEventCallback **および** CreateMemWrEventCallback API **関数は実行に失敗します。**

**状況：**X **または** Y **メモリ・ブロック（**0xFFFF **または** 0x1FFFF**）のアドレスが関数パラメータに含まれている場合でも，**API **関数** CreateMemRdEventCallback **と** CreateMemWrEventCallback **は正しく実行されるよう修正済みです。**

保守／廃止

(c) SPR #199：アプリケーションが不適切な DSP デバイスの項目名を使用する

影響を受ける製品：HSM77016 2.3 M2～2.3 ベータ版

問題：シミュレーションがμPD77110，77111，77112，77113 または 77114 モデルに基づいている場合，HSM77016 はμPD7701x デバイスではなくμPD77100 デバイスの項目名を提供します。

影響を受けるのは，ステータス・レジスタ（SR）割り込みマスク・フラグおよびそれぞれの割り込み要求フラグです。これらのフラグには，次のようにタグが誤って付けられています。

INT1，INT2，INT3，INT4，INT5，INT6，INT7，INT8，INT9，INT10，INT11，および INT12。

これらのタグは，Symbol Table ダイアログと CPU Register ウインドウで表示されます。タグ付けが誤っても，その影響を受けた項目の機能性は影響を受けません。しかし，項目のタグは，HSM77016 の式で使用されるときに与えられるのと同様に指定する必要があります。

状況：HSM77016 は，それぞれの DSP デバイスの項目名を使用します。

(d) SPR #198：ログ・データがテキスト・ファイルにエクスポートされるときアプリケーションが応答を停止する

影響を受ける製品：すべてのバージョン

問題：Log ウインドウが 512 のエントリ，または 512 の倍数のエントリを含んでいて，かつログ・データが（File メニューの Export...コマンドで）テキスト・ファイルにエクスポートされる場合，アプリケーションは応答を停止します。

状況：ログ・データ・エクスポートの問題はフィックスされています。

(e) SPR #197：μPD7701x ファミリにバス・アクセス・エラー・フラグが設定されない

影響を受ける製品：HSM77016 2.3 ベータ版

問題：禁じられているパラレル・データ・メモリ・アクセスの組み合わせが実行され，シミュレーションがμPD7701x ファミリに基づいているときには，エラー・ステータス・レジスタ（ESR）のビット 0（バス・アクセス・フラグ）は 1 に設定されません。

状況：バス・アクセス・フラグが正しく設定されるように修正済みです

(f) SPR #196：インストラクション・メモリ・ウインドウのスクロールで誤ったアドレス範囲が表示される

影響を受ける製品：HSM77016 2.3 ベータ版

問題：HSM77016 のインストラクション・メモリ・ウインドウでスクロールを行うと，希望するアドレス領域ではなく，現在の命令ポインタ・ロケーションのアドレス領域が表示されることがあります。この問題はインストラクション・メモリ・ウインドウのスクロールに関係しているのではなく，HSM77016 のアイドル処理によって引き起こされるものです。

状況：アイドル処理問題はフィックスされています。

保守／廃止

## B.5　バージョン 2.3 ベータ版

**(1) リリース日時**

1999 年 2 月 1 日

**(2) 強化と修正**

- グラフィック・タイミング出力グラフィック・タイミング出力の機能性は修正されています。
- バックトレース：バックトレース機能が追加されています。
- μPD77116 サポート：すべてのμPD77116 周辺装置のシミュレーションが含められていますが，最新のデバイス仕様に従っていません。
- C ソース・レベルのディバグ：CC77016（C コンパイラ）の C ソース・ディバグ機能が追加されています。

## B.6　バージョン 2.3 M2 a

**(1) リリース日時**

1998 年 5 月 5 日

**(2) バージョン 2.3 M2 以降のバグ・フィックス**

**(a) LT（より小さい）命令の評価が常に符号なし比較の結果を戻す**

影響を受ける製品：HSM77016 2.3 M2

問題：LT 命令（OP3=LT(OP1, OP2)）の評価が符号付き比較ではなく OP1 と OP2 の符号なし比較の結果を戻します。

状況：LT 命令の評価が符号付き比較の結果を戻すように修正済みです。

**(3) バージョン 2.3 M1 以降のバグ・フィックス**

**(a) SPR #173：40 個未満の固定小数点ビットを指定したときにタイミング・ファイル・エラーが表示される**

影響を受ける製品：HSM77016 2.0 ベータ版～2.3 M1

問題：固定小数点数のカンマの左右にあるビットの総数が 40 でないとき，HSM77016 はタイミング・ファイル・エラーを表示します（OUTPUT FORMAT.FIX[1.15] など）。

状況：タイミング・ファイル文 “OUTPUT FORMAT FIX[1.15]” は，特定のビット数を要求しないように修正されています。しかし，小数点の前の数字は無視されるので，上記のステートメントを “OUTPUT FORMAT FIX[15]” に変えるように指示する警告メッセージが出力されます。[xxx.yyy] の xxx と yyy の指定は，下位互換の場合にのみサポートされます。

**(b) SPR #172：タイミング・ファイル出力フォーマット・オプションを処理中に HSM77016 がクラッシュする**

影響を受ける製品：HSM77016 2.0 ベータ版～2.3 M1

問題：次のタイミング・ファイル・コマンドが処理されるときにシミュレータがクラッシュします。

OUTPUT FORMAT FIX[9.31]

状況：このバグはフィックスされています。

保守／廃止

**(c)** SPR #171**：グループが存在しない場合ブレークポイント・グループ動作がロードされない**

**影響を受ける製品：**HSM77016 2.0 **ベータ版～**2.3 M1

**問題：イネーブル／ディスエーブル・グループ動作を含むブレークポイントが（たとえば設定ファイルから）**HSM77016 **にロードされるときは，対応するブレークポイント・グループが存在している必要があります。これらが存在しない場合，イネーブル／ディスエーブル・グループ動作はロードされません。**

**状況：イネーブル／ディスエーブル・グループ動作が正しくロードされるよう修正されています。ブレークポイントがヒットしたときに指定のブレークポイント・グループが存在しない場合，イネーブル／ディスエーブル・グループ動作は無効となっています。**

**(d)** SPR #170**：長いファイル名を持つモジュールで設定されているブレークポイントが正しくロードされない**

**影響を受ける製品：**HSM77016 2.0 **ベータ版～**2.3 M1

**問題：長いファイル名を持つモジュールで設定されているブレークポイントを，設定ファイルなどから**HSM77016 **にロードしようとするとき，次のエラー・メッセージが** Breakpoint **ウインドウに表示されます。:“** Extra input after expression found.　This is most likely by a syntax error... **”（式が検出されたあとで余分な入力がなされました。これは，構文エラーによって引き起こされた可能性が非常に高いエラーです。）このブレークポイントのプロパティは，ファイル名のエントリが長いために編集できません。**

**状況：ブレークポイントが，長いファイル名を持つモジュール，もしくは特殊文字を含むモジュール内で設定されている場合，ファイル名はダブル・クォーテーション（“ ”）で囲まれます。**

**(e)** SPR #178**：**Model Editor **のシェル拡張部分に不要なエントリがある**

**影響を受ける製品：**Model Editor (MDLDLG32.DLL) 2.0**（ビルド** 9**）**

**問題：**DSP Selection **タブの** DSP **タイプ選択フィールドが，リストの終わりに** ‘xX’ **と読める不要なエントリを含んでいます。モデル・ファイルを** Windows **エクスプローラでダブルクリックすると，モデルのプロパティが** Model Editor **のシェル拡張部分に表示されます。**Windows **エクスプローラで新たなモデル・ファイルを作成し，**xX **プロセッサのクロック周波数を指定した場合，モデルを再オープンしたときに** DSP **選択フィールドの** xX **エントリが二重に作成されます。**

**状況：エントリ** xX **は，**Model Editor **の現行バージョンでは削除されています。**

## B.7　**バージョン** 2.3 M2

### **(1) リリース日時**

1998 **年** 4 **月** 27 **日**

### **(2) 強化と修正**

$\mu$PD77116 **のサポート：**$\mu$PD77116 **を限定サポートしています。**

TSA **およびタイマ・サポート：**TSA **およびタイマ・サポートが追加されています。**

保守／廃止

## B.8　バージョン 2.3 M1

**(1) リリース日時**

1998 年 1 月 7 日

**(2) 強化と修正**

μPD77116 のサポート：μPD77116 を限定サポートしています。

## B.9　バージョン 2.2 ベータ版 2

**(1) リリース日時**

1997 年 12 月 23 日

**(2) 強化と修正**

グラフィック・タイミング出力：グラフィック・タイミング出力の機能性が改善されています。

**(3) バージョン 2.2 ベータ版以降のバグ・フィックス**

グラフィック・タイミング出力：バッファを記録するグラフィック・タイミング出力の問題がフィックスされています。

## B.10　バージョン 2.2 ベータ版

**(1) リリース日時**

1997 年 10 月 3 日

**(2) バージョン 2.1 以降の強化点**

・グラフィック・タイミング出力機能：グラフィック・タイミング出力機能が Watch ウインドウの強化機能として実装されています。

・周辺装置：μPD7701x 周辺装置が個別に実装されています。

・取り消し（Undo）機能：ユーザ入力取り消しを含む取り消し機能が追加されています。

## B.11　バージョン 2.1

**(1) リリース日時**

1998 年 1 月 20 日 － 公式リリース

**(2) バージョン 2.1 ベータ版 5 以降のバグ・フィックス**

- シミュレーション：無効なエラー・メッセージを伴う割り込み禁止がフィックスされています。この問題は，ROM デバイス（μPD77015，77017，77018）を使用時，命令ポインタ値が ≥0x4000 のときに割り込みが同時に解除されたときに発生していました。
- API：配布 API サンプルにおいて誤動作を引き起こした HSM77016 のアプリケーション・プログラム・インタフェースがフィックスされています。
- Model Editor：Model Editor シェル拡張部分によって新たなモデルを作成したときに発生していた問題がフィックスされています。新たなモデルを作成し，そのモデル・ファイルをダブルクリックすると，モデル名が Model Editor シェル拡張部分のファイル名フィールドに正しく表示されるようになっています。
- Model Editor：Select Model ダイアログの Model Wizard ボタンをクリックしたときに発生していた例外がフィックスされています。


- Model Editor：ID77016 のディバグ対象デバイスの選択に関連する問題がフィックスされています。
  HSM77016 に提供されている Model Editor は，ID77016 および WB77016 と同様，HSM77016 とともに使用できます。
- Model Editor：MDLDLG32.DLL ダイナミック・リンク・ライブラリ，バージョン 2.0 ベータ版（ビルド 4），および HSM77016 パッケージに関して，Model Editor に関連するバグはすべてフィックスされています。

## B.12　バージョン 2.1 ベータ版 5

**(1) リリース日時**

1997 年 12 月 24 日 － この HSM77016 バージョンは，Media OS Plug-In バージョン 1.0 とともにに出荷されています。

## B.13　バージョン 2.1 ベータ版 4

**(1) リリース日時**

1997 年 8 月 11 日

**(2) 強化と修正**

- ブレークポイント
  新たなブレークポイント・タイプが追加されています。“ expression true ” ブレークポイントに加えて，“ expression changed ” ブレークポイントが指定できます。このブレークポイントは指定の式が変更されたときに起動します。“ expression changed ” 項目もブレークポイント条件として使用できます。
- ブレークポイント・インディケータ
  以前のブレークポイント・インディケータ（対応するインストラクション・メモリ・セルの独自の色）に加えて，新たなブレークポイント・インディケータが，対応するインストラクション・メモリ・アドレス，モジュール，またはタイミング・ファイル・ウインドウ行番号の左に示されます。
- API
  API インタフェースが強化されています(詳しくは,第 9 章　アプリケーション･プログラム･インタフェースまたは HSM77016 API Reference オンライン・マニュアルを参照)。

## B.14　バージョン 2.1 ベータ版 3

**(1) リリース日時**

1997 年 5 月 8 日

このバージョンは，Media OS Plug-In バージョン 0.9 アルファ版とともに出荷されています。

## B.15　バージョン 2.1 ベータ版 2

**(1) リリース日時**

1997 年 2 月 18 日

**(2) バージョン 2.1 以降のバグ・フィックス**

- シリアル・インタフェース
  シリアル・インタフェースが 8 ビット・モードで構成されている場合，HSM77016 はシリアル・データ・レジスタの上位 8 ビットを読み出します。

保守／廃止

- NOT 命令

  NOT 命令がシミュレートされ，NOT 命令のディスティネーション・レジスタがレジスタ・ウインドウまたは式で表示に使用されたときに発生していたプログラムの異常終了がフィックスされています。

## B.16　バージョン 2.1 ベータ版

### (1) リリース日時

1996 年 12 月 23 日

### (2) 強化と修正

HSM77016 のアプリケーション・インタフェースとモデル・ウィザードが実装されています。

### (3) バージョン 2.01 以降のバグ・フィックス

- Log File Viewer：Fixed Save As...コマンドが修正されています。Save As...コマンドが適用されても現在アクティブなログ・ファイルは除去されません。
- Log File Viewer：ログ・セッション・リネーム中に Delete キーが押されてもログ・セッションは除去されません。
- Log File Viewer：リネームのあとでログ・セッション群の並べ替えを行ったときに発生していた問題はフィックスされています。一般に並べ替えは高速で行われます。
- Log File Viewer：Purged Entries イベント消失の問題がフィックスされています。
- On-Line Help：ヘルプ・タイトルのμ文字が，u 文字と置き換えられています。
- On-Line Help：Rename Log Session ダイアログの Help ボタンがフィックスされています。
- On-Line Help：Model Definition File トピック・フォーマットがフィックスされています。
- On-Line Help：Target System Model Dialog トピックへの Contents タブ・リンクがフィックスされています。

## B.17　バージョン 2.02

### (1) リリース日時

1997 年 2 月 27 日

### (2) バージョン 2.01 以降のバグ・フィックス

- シリアル・インタフェースが 8 ビット・モードで構成されている場合，HSM77016 はシリアル・データ・レジスタの上位 8 ビットを読み出します。
- NOT 命令がシミュレートされ，NOT 命令のディスティネーション・レジスタがレジスタ・ウインドウまたは式で表示に使用されたときに発生していたプログラムの異常終了がフィックスされています。

保守／廃止

## B.18　バージョン 2.01

**(1) リリース日時**

1997 年 1 月 27 日

**(2) バージョン 2.0 以降のバグ・フィックス**

- NOT 命令：NOT 命令がシミュレートされ，NOT 命令のディスティネーション・レジスタがレジスタ・ウインドウまたは式で表示に使用されたときに発生していたプログラムの異常終了がフィックスされています。

## B.19　バージョン 2.0

**(1) リリース日時**

1996 年 7 月 5 日 － 第一公式リリース

**(2) 強化と修正**

- ログ管理：新規ログ・ファイル管理と専用ログ・ファイル・ビューアが含められています。

**(3) バージョン 2.0 最終ベータ版以降のバグ・フィックス**

- I/O デバイス：μPD7701x ファミリの内蔵 I/O デバイスへアクセスする際にウエイト・サイクルを挿入するプログラムが正しく動作するように修正されました。以前のバージョンではウエイト・サイクルを挿入することはできませんでした。
- ウエイト状態ジェネレータ：外部メモリにアクセスする際，モデル定義ファイルで指定されている分のウエイト・サイクルを挿入するようウエイト状態ジェネレータがプログラムされていない場合，エラー・メッセージを出力するよう修正されました。以前のバージョンでは，メモリへのアクセスが失敗し未定義データが読み出されてもエラー・メッセージは出力されませんでした。
- ウエイト状態ジェネレータ：外部メモリへの書き込み時，モデル定義ファイルで指定されている分のウエイト・サイクルを挿入するよう WAIT 状態ジェネレータがプログラムされていない場合に発生していたプログラム異常終了が修正されました。
- インライン・アセンブラ：インライン・アセンブラが “ REP1 ” のエラー・メッセージを出力することはありません。
- デバイスの制約：次に示す禁止されているコード・シーケンスは，入力またはファイルからロードされたときに検出され，拒否されます。
  - リピート・ターゲットとしての HALT
  - 間接アドレス指定に関して同一のデータ・ポインタを用いるロード／ストア命令に続くデータ・ポインタ・レジスタへの割り当て
  - ループ終了ステートメント前の 3 つの命令に含まれる JMP，CALL，RET，RETI，REP，LOOP，LPOP，HALT，STOP 命令

保守／廃止

## B.20　バージョン 2.0 最終ベータ版

**(1) リリース日時**

1996 年 2 月 5 日

**(2) 強化と修正**

- 高速シミュレーション：高パフォーマンス・シミュレーション・エンジン一式が実装されています。
- 未定義値：未定義値がフルサポートされています。
- 新規ログ・ファイル管理機能は含まれていません。イベント・ログの保存は File.Export 機能で行う必要があります。

**(3) 既知の問題**

- ログ管理：新規ログ・ファイル管理機能は含まれていません。イベント・ログの保存は File.Export 機能で行う必要があります。

## B.21　バージョン 2.0 ベータ版

**(1) リリース日時**

1995 年 9 月 1 日 － 初期リリース

## B.22　SM77016（μPD77016 シミュレータ）バージョン 3.xx の強化

### B.22.1　SM77016 のパフォーマンス

SM77016 のパフォーマンスは大きく改善されています。それぞれのシミュレーション目的とは別に，SM77016 バージョン 3.xx と比較して次の制約が適用されます。

- システム要件（特にメモリ）は SM77016 バージョン 3.xx の場合よりも高くなっています。
- ポート A および DA はサポートされていません。
- ほかの機能はすべて，SM77016 バージョン 3.xx と完全な互換性を持っています。

### B.22.2　ユーザ・インタフェース

HSM77016 のユーザ・インタフェースは，Windows 95 のユーザ・インタフェース・スタイルに適合しています。新たな機能について次に説明します。

**(1) 長いファイル名のサポート**

HSM77016 のファイルには任意のファイル名をつけることができます。ファイル名に 8 文字 ＋ 拡張子 3 文字の制限はなく，スペースを含めることもできます。

**(2) サウンドのサポート**

Windows 95 マルチメディア機能を使用して，次のシミュレーション・イベントにサウンドを関連づけることができます。

- ブレークポイント・ヒット
- シミュレーション・エラー
- シミュレーション警告

**（3）Close ボタン**

HSM77016 のアプリケーション・ウインドウならびに HSM77016 のデータ・ウインドウの右上隅に Close ボタン ☒ があります。このボタンをクリックすると，ウインドウを閉じたり，プログラムを終了したりすることができます。

**（4）マウス右ボタン**

Memory，Register および Watch の各ウインドウでマウスの右ボタンをクリックすると，Copy，Paste，Delete および Log の各機能を使用できます。

**（5）データ・ウインドウ・ヘッダ・コントロール**

Memory ウインドウのヘッダ・コントロールは次の動作に使用できます。

- Address カラム・ヘッダをクリックして任意のアドレスへ移動
- Pointer カラム・ヘッダをクリックしてポインタ・プロパティをチェック，修正
- Data カラム・ヘッダをクリックしてデータをフォーマット

テキスト・ウインドウのヘッダ・コントロールは次の動作に使用できます。

- Line カラム・ヘッダをクリックして任意の行へ移動
- Pointer カラム・ヘッダをクリックしてポインタ・プロパティをチェック，修正（Module ウインドウのみ）

**（6）Symbol Table ダイアログ**

データ配列をより明確で一様なものとするために，Symbol Table ダイアログがタブ付きダイアログに再設計されています。

**（7）Tools メニュー**

以前の Options メニューは，Tools メニューに変更されています。Tools メニューの各コマンドで Log File Viewer ツールの起動，シミュレーション・モデルの変更，およびシ HSM77016 設定の修正を行えます。バージョン 3.xx での Options メニュー・エントリは，Tools メニューの Options...コマンドからのタブ付きメニューでアクセスできます。

## B.22.3　ログ機能

ログ処理には次の機能があります。

- 1 ログ・ファイルに対して複数のログ・セッションが作成可能
- ログ・ファイルとセッションを補足アプリケーション Log Viewer ツールによって管理
- ログ・セッション・データをテキスト・ファイルにエクスポート可能
- ログ・イベントの範囲を拡大
- 格納ログ・データの量をユーザが制限可能
- Log ウインドウにデータ・カラムとヘッダ行を付属

## B.22.4　Log Viewer ツール

Log Viewer とは，HSM77016 で作成されたログ・ファイルとログ・セッションを管理およびチェックするためのツールです。Log Viewer アプリケーションは，Tools メニューの Log Viewer...コマンドから起動します。

保守／廃止

## B.22.5　ソース・レベルのディバグ

**(1) ソース・コードの表示**

ソース・コードは Module ウインドウで表示したり，Instruction Memory ウインドウの命令コードと組み合わせることができます。シミュレーション中にソース・コードを表示するには，ロードされているリンク・ファイルがディバグ情報を含んでいる必要があります。

**(2) ブレークポイントとプロファイリング・マーカ**

ブレークポイントとプロファイリング・マーカは，Module ウインドウのソース行で，または Instruction Memory ウインドウの命令アドレスで設定できます。しかし，マーカとブレークポイントの切り替えを行うときに Module ウインドウと Instruction Memory ウインドウ間にはいくつかの依存関係が存在します。詳しくは，5.1　**メモリ・ウインドウ**と 5.7.2　Module **ウインドウ**を参照してください。

## B.22.6　タイミング・ファイル

**(1) タイミング・ファイルの処理**

タイミング・ファイル・ウインドウがアクティブなデータ・ウインドウであるとき，Run メニューのコマンドが適用されると，タイミング・ファイルは行ごとに処理されます。タイミング・ファイル・ウインドウ以外のウインドウがアクティブの場合には，Run メニュー・コマンドはシミュレーションに適用され，タイミング・ファイルはゼロ・シミュレーション・タイムで実行されます。

**(2) タイミング・ファイルのブレークポイント**

タイミング・ファイルのブレークポイントは，ソース・レベルのブレークポイントに従ってタイミング・ファイル行で切り替えできます。

タイミング・ファイルのブレークポイントは，各ブレークポイントに複数のブレークポイント条件と各種動作を付属させることができる Breakpoint ウインドウにリストされています。

タイミング・ファイル・ブレーク・コマンドで設定された非明示的タイミング・ファイル・ブレークポイントも Breakpoint ウインドウにリストされています。非明示的タイミング・ファイル・ブレークポイントは読み出し専用なので，Breakpoint ウインドウから削除できません。削除は，タイミング・ファイル・コードの編集によってのみ可能です。

## B.22.7　シミュレーションのリセット

シミュレーションのリセット時，リセット動作が実行されるときに実行すべき複数の動作を選択できます。

## B.22.8　レジスタの初期化

すべての HSM77016 変数は，最初に使用する前に定義値を設定する必要があります。この設定をしない場合，シミュレーションは未定義値が処理されるたびに通知メッセージで中断されてしまいます。

## B.22.9　レジスタのロック

シミュレーションで現在使用されているレジスタは（たとえばユーザ自身またはタイミング・ファイルの実行でも）修正できません。

保守／廃止

### B.22.10　オンライン・ヘルプの機能

HSM77016 のオンライン・ヘルプから，さらに充実した内容のコンテキスト依存ヘルプが得られます。コンテキスト・ヘルプは各ダイアログ項目ごとに使用できます。On-line Help には Contents，Index，および Find タブがついており，ヘルプ・システムをナビゲートできるようになっています。

## B.23　SM77016 から HSM77016 への変更点

### B.23.1　シミュレーション・パフォーマンス

シミュレーション・パフォーマンスが大きく向上しました。HSM77016 はシミュレーション速度を上げるコンパイル技法を使用しています。実際のコンパイルは非常に速く，ユーザには見えないバックグラウンドで実行されます。

### B.23.2　永久設定

シミュレーション・セッション間で保持される HSM77016 の設定は，異なるユーザが同じマシンで異なる設定ができるようにシステム・レジストリに格納されています。SM77016 では永久設定を格納するために Windows のディレクトリにあるファイル SM77016.INI を使用していましたが，HSM77016 では永久設定は自動的に格納されるので，このファイルは不要です。

**注意　この最終ベータ版で** HSM77016 **を起動する際に問題が生じた場合，次の手順を実行してください。**

1. Windows **の** Start **ボタンで，**Run...**を選択し，**REGEDIT **とタイプ入力し，**OK **を押します。**
2. **レジストリ・エディタでキー“** HKEY_CURRENT_USER＼Software＼ATAIR＼μPD77016 High-Speed Simulator **”を探して選択します。**
3. Registry **メニューから** Export Registry File...**を選択し，選択したキーをファイルにエクスポートします。このとき，**Export Registry File **ダイアログが** Export **範囲として“** Selected branch **”を指定しており，また** branch **が“** HKEY_CURRENT_USER＼Software＼ATAIR＼μPD77016 High-Speed Simulator **”であることを確認します。**
4. **キー“** HKEY_CURRENT_USER＼Software＼ATAIR＼μPD77016 High-Speed Simulator **”を削除します。**
5. **手順** 3 **でエクスポートしたファイルを** ATAIR Software **にレポートします。**
6. **ほかのエントリを修正するとシステムが動作不能になることがありますので十分注意してください。**

### B.23.3　セッション・イメージ・ファイル（SIM ファイル）

セッション・イメージ・ファイルは，SM77016 バージョン 3 と HSM77016 間で交換可能です。ただし次の制約があります。シミュレーション・コアが著しく異なっているため，この HSM77016 ベータ版と SM77016 バージョン 3 はそれぞれが SIM ファイル書き込んだ次の情報を，互いに復元することができません。

- μPD77016 CPU タイミングの状態
- シリアル・インタフェース・シフト動作の状態

ほかのすべての情報，たとえばメモリ内容やレジスタ内容は，SM77016 と HSM77016 バージョン間で SIM ファイルを介してやり取りできます。

〔メ　モ〕

保守／廃止

# 付録C 索　　引

## C. 1　五十音で始まる語句の索引

保守／廃止

保守／廃止

# C. 2 アルファベットで始まる語句の索引

保守／廃止

# C. 3 コマンド別索引

保守／廃止

## お問い合わせ先

### 【技術的なお問い合わせ先】

ＮＥＣ半導体テクニカルホットライン
（電話：午前 9:00～12:00，午後 1:00～5:00）

電　話　：044−435−9494
FAX　：044−435−9608
E-mail　：s-info@saed.tmg.nec.co.jp

### 【営業関係お問い合わせ先】

第一販売事業部
東　京　(03)3798−6106, 6107, 6108
名古屋　(052)222−2375
大　阪　(06)6945−3178, 3200, 3208, 3212
仙　台　(022)267−8740
郡　山　(024)923−5591
千　葉　(043)238−8116

第二販売事業部
東　京　(03)3798−6110, 6111, 6112
立　川　(042)526−5981, 6167
松　本　(0263)35−1662
静　岡　(054)254−4794
金　沢　(076)232−7303
松　山　(089)945−4149

第三販売事業部
東　京　(03)3798−6151, 6155, 6586, 1622, 1623, 6156
水　戸　(029)226−1702
広　島　(082)242−5504
高　崎　(027)326−1303
鳥　取　(0857)27−5313
太　田　(0276)46−4014
名古屋　(052)222−2170, 2190
福　岡　(092)261−2806

### 【資料の請求先】

上記営業関係お問い合わせ先またはNEC特約店へお申しつけください。

### 【インターネット電子デバイス・ニュース】

NECエレクトロンデバイスの情報がインターネットでご覧になれます。　URL（アドレス）　http://www.ic.nec.co.jp/

保守／廃止

# アンケート記入のお願い

お手数ですが，このドキュメントに対するご意見をお寄せください。今後のドキュメント作成の参考にさせていただきます。

［ドキュメント名］ HSM77016 ユーザーズ・マニュアル

（(U11602JJ3V0UM00（第 3 版)）

［お名前など］（さしつかえのない範囲で）

御社名（学校名，その他） （ ）
ご住所 （ ）
お電話番号 （ ）
お仕事の内容 （ ）
お名前 （ ）

1．ご評価（各欄に○をご記入ください）

| 項　　目 | 大変良い | 良　い | 普　通 | 悪　い | 大変悪い |
|---|---|---|---|---|---|
| 全体の構成 | | | | | |
| 説明内容 | | | | | |
| 用語解説 | | | | | |
| 調べやすさ | | | | | |
| デザイン，字の大きさなど | | | | | |
| その他（ ） | | | | | |
| （ ） | | | | | |

2．わかりやすい所（第　章，第　章，第　章，第　章，その他 ）

理由［ ］

3．わかりにくい所（第　章，第　章，第　章，第　章，その他 ）

理由［ ］

4．ご意見，ご要望

| |
|---|
| |

5．このドキュメントをお届けしたのは

NEC 販売員，特約店販売員， その他（ ）

ご協力ありがとうございました。

下記あてに FAX で送信いただくか，最寄りの販売員にコピーをお渡しください。

日本電気(株) NEC エレクトロンデバイス
半導体テクニカルホットライン
FAX：(044) 435-9608

2000.6

キリトリ