EPSON
EXCEED YOUR VISION

CMOS SINGLE CHIP MICROCOMPUTER

S5U1C88000Q Manual

S1C63/S1C88 Family組み込みシステムシミュレータパッケージ Ver.7

セイコーエプソン株式会社

Rev.1.0

本資料のご使用につきましては、次の点にご留意願います。

本資料の内容については、予告無く変更することがあります。

1. 本資料の一部、または全部を弊社に無断で転載、または、複製など他の目的に使用することは堅くお断りいたします。
2. 本資料に掲載される応用回路、プログラム、使用方法等はあくまでも参考情報であり、これら起因する第三者の知的財産権およびその他の権利侵害あるいは損害の発生に対し、弊社はいかなる保証を行うものではありません。また、本資料によって第三者または弊社の知的財産権およびその他の権利の実施権の許諾を行うものではありません。
3. 特性値の数値の大小は、数直線上の大小関係で表しています。
4. 本資料に掲載されている製品のうち「外国為替及び外国貿易法」に定める戦略物資に該当するものについては、輸出する場合、同法に基づく輸出許可が必要です。
5. 本資料に掲載されている製品は、生命維持装置その他、きわめて高い信頼性が要求される用途を前提としていません。よって、弊社は本(当該)製品をこれらの用途に用いた場合のいかなる責任についても負いかねます。
6. 本資料に掲載されている会社名、商品名は、各社の商標または登録商標です。

製品型番体系

●デバイス

S1 C 88104 F 0A01 00

■ 梱包仕様
- 00 : テープ&リール以外
- 0A : TCP BL 2方向
- 0B : テープ&リール BACK
- 0C : TCP BR 2方向
- 0D : TCP BT 2方向
- 0E : TCP BD 2方向
- 0F : テープ&リール FRONT
- 0G : TCP BT 4方向
- 0H : TCP BD 4方向
- 0J : TCP SL 2方向
- 0K : TCP SR 2方向
- 0L : テープ&リール LEFT
- 0M : TCP ST 2方向
- 0N : TCP SD 2方向
- 0P : TCP ST 4方向
- 0Q : TCP SD 4方向
- 0R : テープ&リール RIGHT
- 99 : 梱包仕様未定

■ 仕様

■ 形状
[D: ベアチップ、F: QFP、B: BGA]

■ 機種番号

■ 機種名称
[C: マイコン、デジタル製品]

■ 製品分類
[S1: 半導体]

●開発ツール

S5U1 C 88348 D1 1 00

■ 梱包仕様
[00: 標準梱包]

■ バージョン
[1: Version 1]

■ ツール種類
- Hx : ICE
- Ex : EVAボード
- Px : ペリフェラルボード
- Wx : FLASHマイコン用ROMライタ
- Xx : ROMライタ周辺ボード
- Cx : Cコンパイラパッケージ
- Ax : アセンブラパッケージ
- Dx : 機種別ユーティリティツール
- Qx : ソフトシミュレータ

■ 対応機種番号
[88348: S1C88348用]

■ ツール分類
[C: マイコン用]

■ 製品分類
[S5U1: 半導体用開発ツール]

ー目　次ー

1　S1C63/S1C88 Family組み込みシステムシミュレータパッケージの概要

1.1　はじめに

S1C63/S1C88 Family組み込みシステムシミュレータパッケージはS1C63 FamilyおよびS1C88 Familyの開発ツールパッケージの1つです。このパッケージに含まれるシミュレータにより、S1C63/S1C88 Familyマイクロコンピュータ用に作成したプログラムをICE等の専用のハードウェアツールを使用せずにパーソナルコンピュータのみでデバッグできます。このシミュレータは一般のデバッグ機能に加え、入出力ポートを使用したプッシュボタンやキーマトリクス、さらにはLCDの表示もシミュレートできます。このためのビットマップやLCDパネルデータを作成するユーティリティも本パッケージに含まれています。

1.2　ツールの概要

本パッケージに含まれるソフトウェアツールの概要を以下に示します。

シミュレータ（Sim63.exe, Sim88.exe）

パーソナルコンピュータのみで動作のシミュレーションとデバッグを行うソフトウェアです。キーボードやファイルからのコマンド入力で各種機能を実行します。なお、ブレークやステップ実行などの頻繁に使用するコマンドはツールバーに登録されており、キーボード操作の量を抑えています。また、プログラムコードやレジスタ内容、コマンド実行結果がマルチウィンドウ上に表示できるため、デバッグ作業が効率良く行えます。さらに入出力ポートを使用したプッシュボタンやキーマトリクス、LCDの表示もシミュレートできます。シミュレータはS1C63 Family用（Sim63）とS1C88 Family用（Sim88）にそれぞれ用意されています。

LCDパネルカスタマイズユーティリティ（LcdUtil.exe）　S1C63/S1C88 Family共通

シミュレータ（Sim63/Sim88）でモノクロLCDパネルの表示をシミュレートするための、パネルレイアウトおよびCOM/SEGポートの割り付けデータを作成します。アイコンなどはビットマップファイル（.bmp）から読み込むため汎用ペイントソフトで作成可能で、実際の製品と同様の画面をシミュレートすることができます。

ビットマップユーティリティ（BmpUtil.exe）　S1C63/S1C88 Family共通

ドットマトリックスLCDの表示に使用するキャラクタジェネレータやスプライトなどのビットマップイメージデータを作成します。出力データは指定のラベルが付いたアセンブリソース形式で作成されますので、そのままアセンブルしてプログラムにリンクすることができます。

ポート設定ユーティリティ（PrtUtil.exe）　S1C63/S1C88 Family共通

シミュレータ（Sim63/Sim88）で、PCのキーを使用してターゲットのキー入力をシミュレートするためのポート設定ファイルを作成します。

ロジックアナライザ（LogAna.exe）　S1C63/S1C88 Family共通

シミュレータ（Sim63/Sim88）の[I/O Terminal]ウィンドウで作成したログファイルを入力し、そこに記録されている入出力ポートの状態を波形表示します。

ROMデータ設定ユーティリティ（Rom88.exe）　S1C88 Family用

バイナリ形式のROMイメージデータをシミュレータ（Sim88）にロードします。また、Sim88の[Dump]ウィンドウでパッチをあてたデータをROMイメージで保存することもできます。

2　インストール

この章では本パッケージに収められたツールの動作環境とインストール方法について説明します。

2.1　動作環境

S1C63/S1C88 Family組み込みシステムシミュレータパッケージを使用するには、次の動作環境が必要です。

● コンピュータ本体

IBM PC/AT互換機が必要です。Pentium 400MHz以上のCPUと64MB以上のRAMを搭載した機種を推奨します。

＊ 400MHzのPentium CPUおよび64MB以上のRAMを搭載したコンピュータで、Sim63ではOSC3クロックが0.5MHz程度、Sim88ではOSC3クロックが1MHz程度のアプリケーションをリアルタイムに実行できます(コンピュータの能力や動作条件などによりリアルタイム実行できない場合もあります)。

● ディスプレイ

800×600ドット以上のディスプレイが必要です。

● ハードディスク

本パッケージをインストールするには、50MB以上の空き容量が必要です。

● システムソフトウェアについて

Windowsが必要です(詳細はS5U1C88000QのReadme_J.txtを参照してください)。

● その他

S1C63 Familyのアプリケーション開発には、本パッケージ以外にS1C63 Familyアセンブラパッケージが、S1C88 Familyのアプリケーション開発にはS1C88 Family統合ツールパッケージが必要です。

2.2　インストール方法

ツールのインストールはインストーラ(Setup.exe)によって行います。

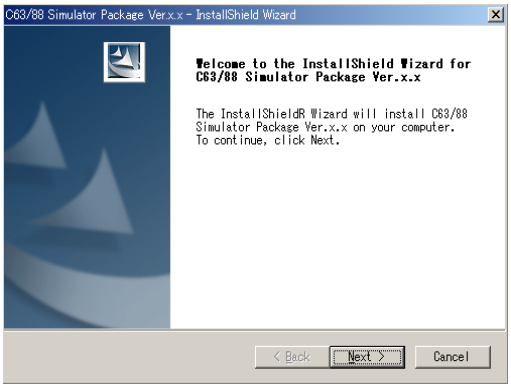
ツールをインストールするには


Setup.exe

(1) Microsoft Windowsを起動させてください。
すでに起動している場合は、開いているプログラムをすべて終了させてください。

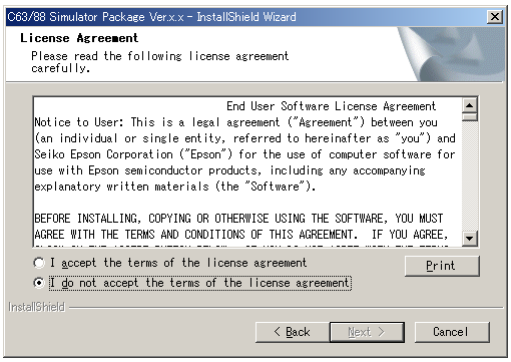
(2) S5U1C88000Qの圧縮ファイルをSeiko Epsonのユーザサイトよりダウンロードして、任意のフォルダに解凍してください。

(3) Setup.exeをダブルクリックして起動させてください。

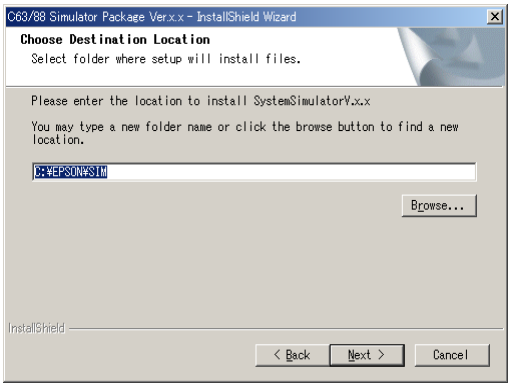


Welcom to ...
インストールウィザードのスタート画面が表示されます。

(4) [Next >]ボタンをクリックして次に進めてください。
エンドユーザソフトウェアライセンス契約画面が表示されます。表示されたライセンス契約内容は必ずお読みください。



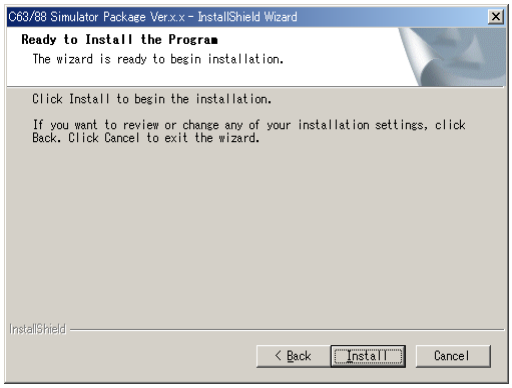
(5) ライセンス契約に同意される場合は"I accept the terms of the license agreement"を選択して、[Next >]ボタンをクリックしてください。
同意できない場合は、[Cancel]ボタンをクリックしてインストーラを終了させてください。



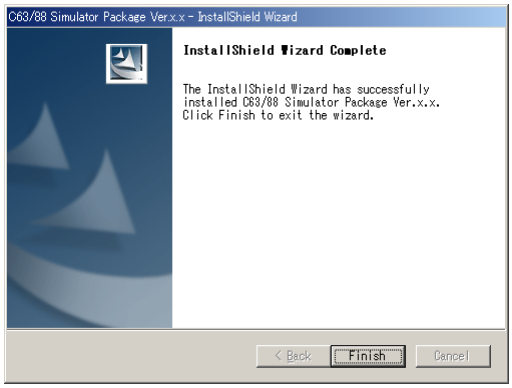
Choose Destination Location
S1C63/S1C88 Family組み込みシステムシミュレータパッケージをインストールするフォルダを指定する画面が表示されます。

(6) 表示されたインストール先を確認します。
インストール先を変更する場合は、[Browse...]ボタンでフォルダ選択ダイアログボックスを表示させます。インストールするフォルダをダイアログボックス上のリストから選択するか、[Path]テキストボックスにフォルダへのパスを入力して[OK]ボタンをクリックしてください。

(7) [Next >]ボタンをクリックします。



(8) インストールの確認画面が表示。
インストールを開始する場合は、[Install]ボタンをクリックしてください。



InstallShield Wizard Complete

(9) [Finish]をクリックしてインストーラを終了させてください。

インストールを途中で中止するには

インストール中に表示されるダイアログボックスはすべて[Cancel]ボタンを持っています。中止するにはダイアログボックスが表示されたところで[Cancel]をクリックしてください。

アンインストールするには

ツールをアンインストールするには、コントロールパネルの[アプリケーションの追加と削除]を使用してください。

2.3　インストール後のディレクトリとファイル構成

インストーラは、指定ディレクトリ(デフォルトは"C:¥EPSON¥SIM¥")に以下のファイルをコピーします。

```
[EPSON¥SIM]
    README_J.TXT          ...Readme_Jテキストファイル(日本語)
    README_E.TXT          ...Readme_Eテキストファイル(英語)

    [S1C63]               ...S1C63 Familyシミュレータフォルダ
        SIM63.EXE         ...シミュレータ
        xxxxxxxx.INI      ...初期化ファイル
        xxxxxxxx.PAR      ...パラメータファイル
        xxxxxxxx.BMC      ...コンポーネントファイル
            :
        .....             ...その他関連ファイル

        [SAMPLE]          ...サンプルフォルダ
            README_J.TXT  ...サンプルプログラム説明(日本語)
            README_E.TXT  ...サンプルプログラム説明(英語)
            SAMPLEx.xxx   ...サンプルファイル
                :
```

```
[S1C88]                 ...S1C88 Familyシミュレータフォルダ
    SIM88.EXE           ...シミュレータ
    xxxxxxxx.INI        ...初期化ファイル
    xxxxxxxx.PAR        ...パラメータファイル
    xxxxxxxx.BMC        ...コンポーネントファイル
        :
    .....               ...その他関連ファイル

    [SAMPLE]            ...サンプルフォルダ
        README_J.TXT    ...サンプルプログラム説明(日本語)
        README_E.TXT    ...サンプルプログラム説明(英語)
        SAMPLEx.xxx     ...サンプルファイル
            :

[UTILITY]               ...ユーティリティフォルダ
    BMPUTIL.EXE         ...ビットマップユーティリティ
    LCDUTIL.EXE         ...LCDパネルカスタマイズユーティリティ
    LOGANA.EXE          ...ロジックアナライザ
    PRTUTIL.EXE         ...ポート設定ユーティリティ
    ROM88.EXE           ...ROMデータ設定ユーティリティ
        :

[DOC]
    [JAPANESE]          ...ドキュメント(日本語)
        REL_xxx_J.TXT   ...リリースノート
        TBD_J.PDF       ...マニュアル(PDF)
            :

    [ENGLISH]           ...ドキュメント(英語)
        REL_xxx_E.TXT   ...リリースノート
        TBD_E.PDF       ...マニュアル(PDF)
            :
```

● PDF形式のオンラインマニュアル

添付のオンラインマニュアルはPDF形式で作成されており、参照するにはAdobe Acrobat ReaderのVer. 4.0以降が必要です。

● 今後リリースされる機種のファイル

今後リリースされる機種用のファイルは、セイコーエプソンのマイクロコンピュータユーザーズサイトで提供いたします。インストールにつきましては、それぞれのReadmeファイルを参照してください。

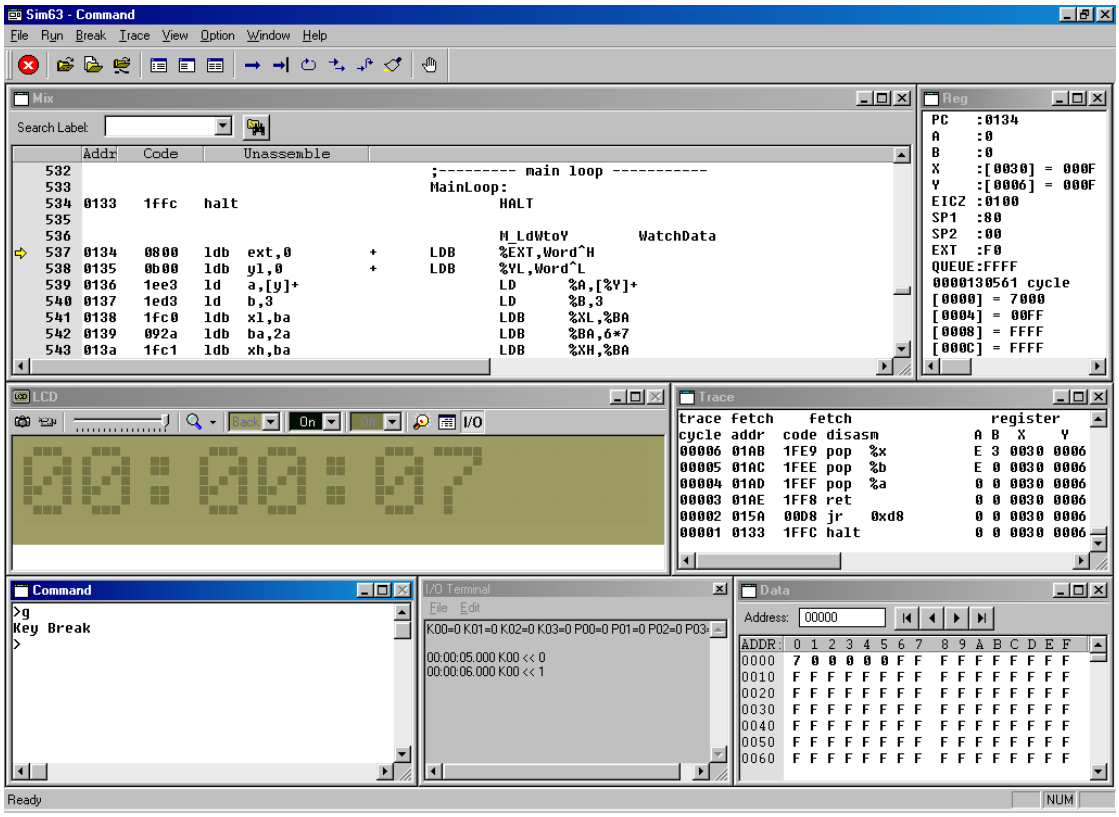
3　S1C63 Familyシミュレータ

3.1　概要

シミュレータsim63はS1C63 Familyの開発ツールです。このシミュレータにより、S1C63アセンブラで作成したプログラムをICE等の専用のハードウェアツールを使用せずにパーソナルコンピュータのみでデバッグできます。このシミュレータは一般のデバッグ機能に加え、入出力ポートを使用したプッシュボタンやキーマトリクス、シリアルおよび汎用ポートの入出力、A/D変換、さらにはLCDの表示もシミュレートできます。このためのビットマップやLCDパネルデータを作成するユーティリティも本パッケージに含まれています。

シミュレータの特長を以下に示します。

- 外部のデバッグ用ハードウェアを使用せずに、PC上でLCD表示を含めたシミュレーションが可能
- マルチウィンドウにより各種のデータを一度に参照可能
- 使用頻度の高いコマンドはツールバーおよびメニューからマウス操作で実行可能
- アセンブリソースコードに対応したソース表示およびシンボリックデバッグ機能
- プログラムの連続実行と2種類のステップ実行が可能
- 5種類のブレーク機能
- トレース機能
- コマンドファイルによるコマンドの自動実行機能



3.2　ソフトウェア開発フロー

図3.2.1にS1C63 Familyのソフトウェア開発フローを示します。太字の部分が本パッケージに含まれるツールと関連ファイルです。

注: 図に示すとおり、S1C63 Familyのソフトウェア開発には本パッケージ以外に、S1C63 Familyアセンブラパッケージが必要です。

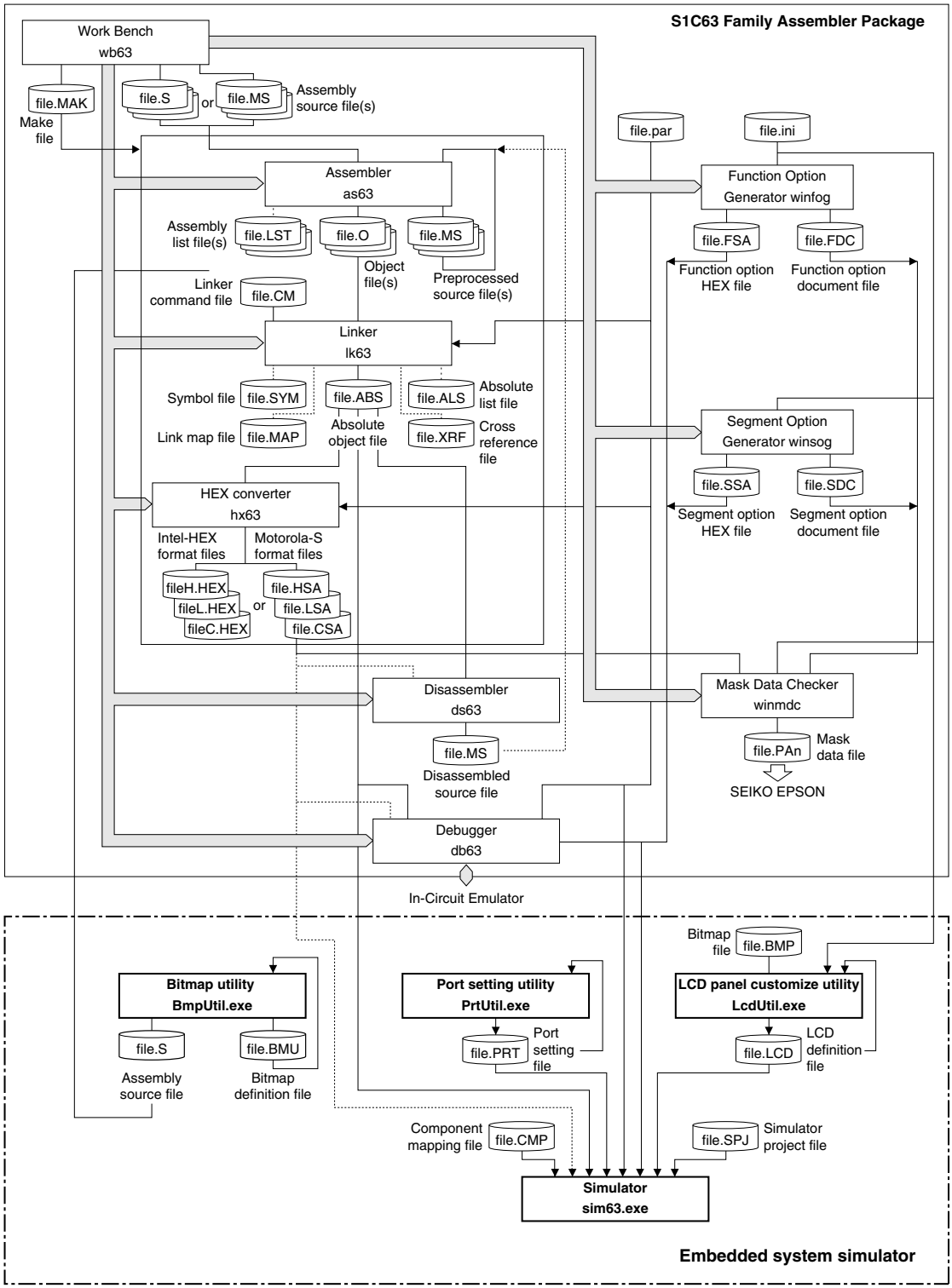


図3.2.1 ソフトウェア開発フロー

3.3　シミュレーション機能/操作の概要

ここでは、シミュレータsim63を使用したPC上でのターゲットアプリケーションのシミュレーションについて説明します。

● 対応機種

本シミュレータの対応機種につきましてはReadme_J.txtの、"サポート機種および周辺機器"を参照してください。

● ターゲットとしているアプリケーション

本シミュレータは、時計、電卓、電子手帳、携帯ゲーム等、キー入力およびLCD表示機能を持つアプリケーションのシミュレーションに最適です。

OSC1クロック動作はリアルタイムに実行可能です。OSC3クロック動作については、最大500kHz～1MHz程度までをリアルタイムに実行できます(Pentium 400MHz、64MB RAM相当のPCで500kHz程度のリアルタイム実行が可能)。

ただし、インストラクションレベルのシミュレーション精度のため、制御系などの高精度なタイミングのシミュレーションは行えません。

外部デバイスは、モノクロLCDパネル、バックライト、キー、キーマトリクスのみをサポートします。

注: PC上でのシミュレーションのため、上記以外の制限事項もあります。"3.15　制限事項"およびReadme_J.txtを参照してください。

● 回路図情報のシミュレータへの入力

外部デバイスの動作をシミュレートするために必要な情報は、ファイルを作成してシミュレータに設定することができます。

・外部デバイスのマッピングと内部発振クロック周波数

バックライトを割り付けるアドレスとバックライトの制御ビットを記述したコンポーネントマップファイル(file.cmp)を作成してシミュレータにロードすることにより、これらの外部デバイスの制御をシミュレートすることができます。また、コンポーネントマップファイルにはOSC1/OSC3発振クロック周波数などのシミュレート条件も設定します。

ファイルの作成方法については、"3.12 コンポーネントマップファイル(.cmp)"を参照してください。

・キー、キーマトリクス仕様の入力

キー入力ポート、キーマトリクスを構成する入力/出力ポート、ターゲットとPCキーボードの対応を記述したポート設定ファイル(.prt)を作成してシミュレータにロードすることにより、PCキーボードを使用してキー入力をシミュレートすることができます。

ファイルの作成方法については、"7 ポート設定ユーティリティ"を参照してください。

・LCDデザインの入力

LCDパネルのレイアウトおよびSEG/COMポートの割り付けを記録したLCDパネル定義ファイル(file.lcd)を作成してシミュレータにロードすることにより、LCDパネル上の表示をシミュレートすることができます。

LCDパネル定義ファイルの作成方法については、"5 LCDパネルカスタマイズユーティリティ"を参照してください。

これらのファイルは通常、シミュレータの起動時に読み込まれます。

● ターゲットプログラムロード、実行方法

ターゲットプログラムはシミュレータのlfコマンドでロードし、g(またはgr、s、n)コマンドで実行します。

プログラム実行中のLCD表示や入出力のシミュレーションはシミュレータの[LCD]ウィンドウ、[I/O Terminal]ウィンドウを使用して行います。

・LCD表示: [LCD]ウィンドウ内に実際のLCDと同様に表示されます。
・キー入力: [LCD]ウィンドウをアクティブにしてPCキーボードの該当キーを入力します。
専用のウィンドウでキー入力状態の設定/確認も行えます。
・SVD評価: [LCD]ウィンドウ上のSVDスライドバーを操作してSVD機能をシミュレートできます。
・シリアルインタフェースと汎用ポートの入出力、A/D変換:
[I/O Terminal]ウィンドウをアクティブにして入力シーケンスを記述したIOTファイルを読み込み、シミュレーションを行います。

3.4　入出力ファイル

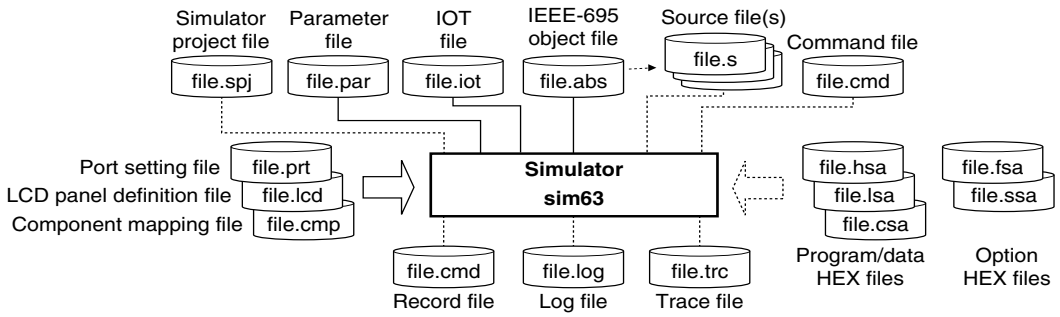


図3.4.1　入出力ファイル

● パラメータファイル(file_name.par)
各機種のメモリ情報などが記録された、シミュレータを起動するために必須のファイルです。S1C63 Familyアセンブラパッケージに添付されています。

● アブソリュートオブジェクトファイル(file_name.abs)
リンカで生成したIEEE-695形式のオブジェクトファイルです。lfコマンドにより読み込みます。デバッグ情報を含んだIEEE-695形式のファイルを読み込むことで、ソース表示およびシンボリックデバッグが行えます。

● ソースファイル(file_name.s)
上記オブジェクトファイルのソースファイルで、ソース表示を行うときに読み込まれます。

● プログラムHEXファイル(file_name.hsa, file_name.lsa)
コードROMのロードイメージファイル(モトローラS2形式のHEXファイル)で、loコマンドにより読み込みます。".hsa"はプログラムコードの上位5ビット、".lsa"は下位8ビットに対応しています。これらのファイルは、IEEE-695形式のオブジェクトファイルからマスクデータ作成用にHEXコンバータで生成したものです。IEEE-695形式のファイルのようにソース表示およびシンボリックデバッグは行えませんが、最終プログラムデータの動作確認が行えます。

● データROM HEXファイル(file_name.csa)
データROMのロードイメージファイル(モトローラS2形式のHEXファイル)で、loコマンドにより読み込みます。このファイルは、IEEE-695形式のオブジェクトファイルからマスクデータ作成用にHEXコンバータで生成したものです。IEEE-695形式のアブソリュートオブジェクトファイルをロードした場合、このファイルをロードする必要はありません。

● ファンクションオプションHEXファイル(file_name.fsa)
ファンクションオプションジェネレータで生成した、モトローラS2フォーマットのマスクオプション設定用ファイルです。loコマンドにより読み込みます。

● セグメントオプションHEXファイル(file_name.ssa)
セグメントオプションジェネレータで生成した、モトローラS2フォーマットのマスクオプション設定用ファイルです。loコマンドにより読み込みます。機種によっては、セグメントオプションがありません。それらの機種においても、シミュレータ設定用のHEXファイルが用意されます。

● シミュレータプロジェクトファイル(file_name.spj)
sim63の起動時に、パラメータファイル、LCDパネル定義ファイル、コンポーネントマップファイル、ポート設定ファイルを一括して指定するためのファイルです。エディタにより各ファイル名を入力して作成します。なお、sim63はこのファイルを生成しない場合でも、各ファイルをダイアログで選択することによって起動可能です。

● IOTファイル(file_name.iot)
I/Oテキストターミナルでシリアル/汎用ポート入出力、A/D変換をシミュレートするための入力データを記述したテキストファイルです。

● LCDパネル定義ファイル(file_name.lcd)
LCDパネルのレイアウトおよびCOM/SEGポート割り付け情報が記録されたファイルです。LCDパネルカスタマイズユーティリティ(LcdUtil)で生成します。

● コンポーネントマップファイル(file_name.cmp)
バックライトなどのマップアドレスなどを設定するテキストファイルです。

● ポート設定ファイル(file_name.prt)
プッシュキーやキーマトリクスとポートの対応を定義しておくテキストファイルです。ポート設定ユーティリティ(PrtUtil)で生成します。

● コマンドファイル(file_name.cmd)
連続して実行させるデバッグコマンドを記述したテキストファイルです。頻繁に使用する一連のコマンドを書き込んでおくことで、キーボードからのコマンド入力の手間を省くことができます。このファイルはcomまたはcmwコマンドにより読み込み、実行させます。

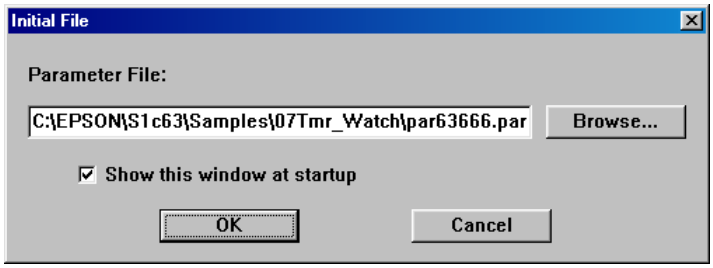
● ログファイル(file_name.log)
実行したコマンドと実行結果が出力されます。このファイル出力はlogコマンドによって制御できます。

● レコードファイル(file_name.cmd)
実行したコマンドがテキスト形式で出力されます。このファイル出力はrecコマンドによって制御できます。出力されたファイルはそのままコマンドファイルとして使用できます。

● トレースファイル(file_name.trc)
トレースデータの指定範囲が出力されます。このファイル出力はtfコマンドによって制御できます。

3.5　起動と終了

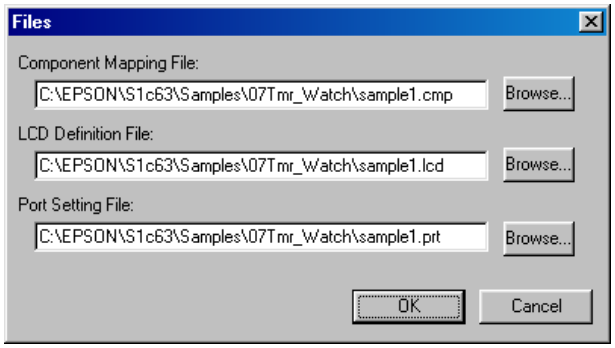
このアイコンをダブルクリックすると、シミュレータが起動します。
シミュレータを初めて起動すると次のダイアログボックスが表示されますので、パラメータファイルまたはシミュレータプロジェクトファイル(3.14項参照)の名称をテキストボックスに入力するか、[Browse...]ボタンで選択してください。



[Show this window at startup]チェックボックスをオフにすると、このダイアログボックスは次回の起動時から表示されなくなり、同じファイルが選択されます。その後、再選択する場合はparコマンド([File | Load Parameter File...])で再表示させることができます。

チェックボックスをオンにした場合、次回の起動時はテキストボックスに同じファイル名が表示され、[OK]のクリックのみで済みます。

ここでパラメータファイルを選択すると次のダイアログボックスが表示されますので、コンポーネントマップファイル、LCDパネル定義ファイル、ポート設定ファイルをそれぞれのテキストボックスに入力するか、[Browse...]ボタンで選択してください。



パラメータファイルを選択した場合、上記の[Show this window at startup]チェックボックスをオフにしても、このダイアログボックスが次回の起動時にも表示されます。ただし、テキストボックスに同じファイル名が表示され、[OK]のクリックのみで済みます。
シミュレータプロジェクトファイルを選択した場合、このダイアログボックスは表示されません。

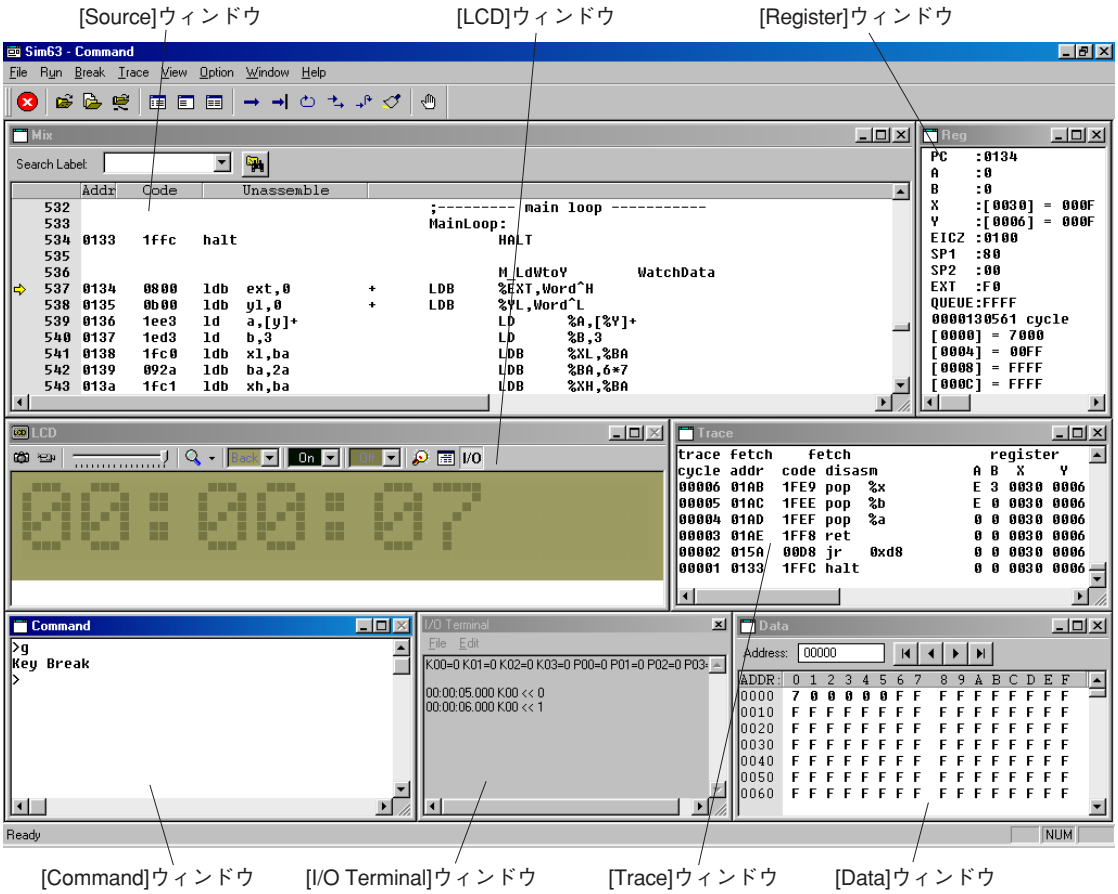
シミュレータを終了するには、[File]メニューから[Exit]を選択します。

3.6　ウィンドウ

ここでは、シミュレータで使用するウィンドウの種類を説明します。

3.6.1 ウィンドウの基本構成

シミュレータのウィンドウ構成は次のとおりです。



● 全ウィンドウの共通な操作

(1) ウィンドウのオープン/クローズとアクティブ化

[Command]と[LCD]以外のウィンドウはすべて閉じることと開くことができます。
ウィンドウを開くには、[View]メニューからそのウィンドウ名を選択してください。結果を特定のウィンドウに表示するデバッグコマンドを実行した場合も、対応するウィンドウが開きます。
ウィンドウを閉じるには、ウィンドウの[閉じる]ボタンをクリックしてください。
開いているウィンドウは[Window]メニューにリストされます。そこからウィンドウ名を選択することで、そのウィンドウがアクティブになります。ウィンドウ上をクリックすることでも同様です。また、[Ctrl]+[Tab]のキー操作によってもアクティブウィンドウの切り替えが行えます。

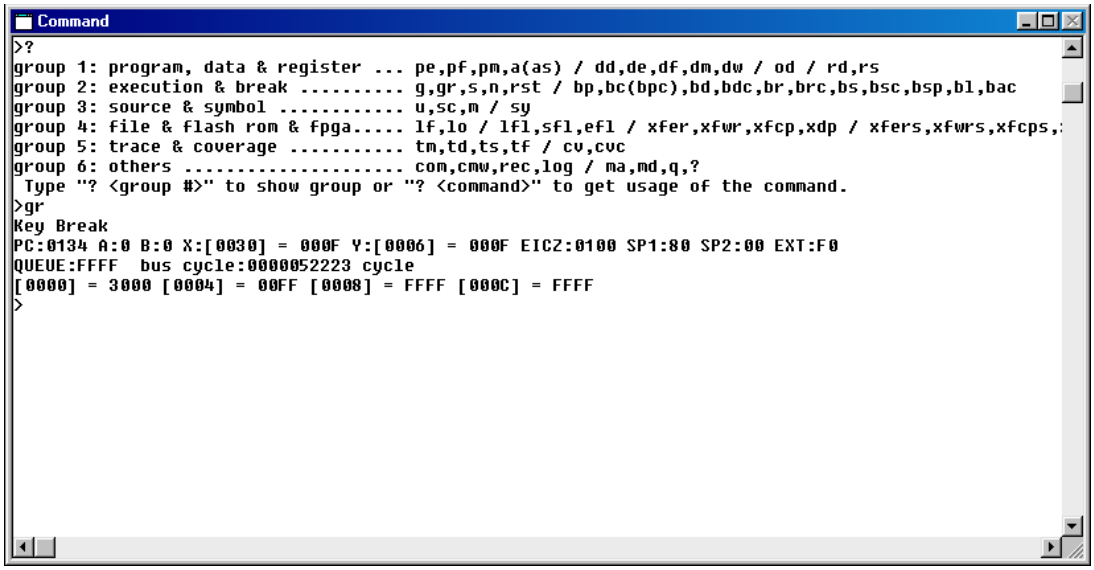
(2) サイズの変更と移動

それぞれのウィンドウサイズは、ウィンドウの境界をドラッグすることによって任意の大きさに変更できます。[最小化]ボタン、[最大化]ボタン等も一般のWindowsアプリケーションと同様です。各ウィンドウはタイトルバーをドラッグすることによって、表示位置を変更できます。ただし、サイズ変更、移動ともに、アプリケーションウィンドウの範囲内に限られます。

(3) その他

開いているウィンドウは、[Window]メニューの[Cascade]または[Tile]を選択することで整列させることができます。

3.6.2 [Command]ウィンドウ



[Command]ウィンドウは以下の目的に使用します。

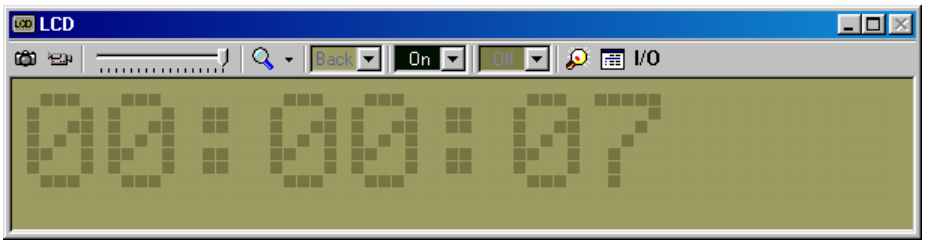
(1)デバッグコマンドの入力
[Command]ウィンドウにプロンプト">"が表示され、キーボードからのコマンド入力を受け付けます。

(2)メニュー/ツールバーから選択したコマンドの表示
メニューやツールバーからデバッグコマンドを選択して実行させた場合は、そのコマンドラインが[Command]ウィンドウに表示されます。

(3)コマンド実行結果の表示
コマンドの実行結果を表示します。ただし、コマンド実行結果の中には、[Source]ウィンドウ、[Data]ウィンドウ、[Register]ウィンドウ、[Trace]ウィンドウに表示される内容もあります。これらの内容は対応するウィンドウが開いていれば、その中に表示されます。ウィンドウが閉じている場合は、[Command]ウィンドウに表示されます。
ログファイルへの書き込みを設定中は、その書き込み内容が表示されます。(logコマンド参照)

注: [Command]ウィンドウを閉じることはできません。

3.6.3 [LCD]ウィンドウ

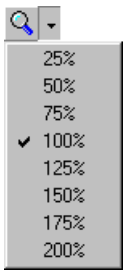


注: [LCD]ウィンドウを閉じることはできません。

[LCD]ウィンドウには、以下の機能があります。

(1)LCD表示のシミュレーション

LCDパネル定義ファイルに設定されたLCDパネルを表示します。アイコンやドットマトリクスの表示は、プログラムの実行に従って変化します。
また、以下のコントロールでパネルのサイズや色を設定できます。



[拡大縮小]ボタン・ドロップダウンリスト
ボタンをクリックするごとにパネルのサイズが25%ずつ拡大します。200%まで拡大し、その次は25%のサイズに縮小します。プルダウンリストでは、直接拡大・縮小の数値を選択します。

[Back]、[On]、[Off]ドロップダウンリスト
LCDパネル自体の色を設定します。
[Back] 背景色
[On]　ドットがOnのときの色
[Off]　ドットがOffのときの色

[Back Light]ボタン
次のダイアログボックスが表示され、バックライトの色を最大4色まで登録しておくことができます。色はRGBの各スライダで調整してください。左下のチェックボックスを選択すると、そのバックライトが設定されます。

(2) パネルイメージのキャプチャ

カメラボタン

このボタンをクリックすると、その時点のLCDパネルイメージをキャプチャし、ビットマップイメージとしてクリップボードに転送します。これをペイントソフト等にペーストし、印刷したりファイルに保存することができます。

このボタンはプログラム実行中は無効となるため、キャプチャしたいイメージのところで実行を中断してから使用してください。

ビデオボタン

このボタンをクリックすると、パネルイメージの録画機能(AVIファイルとして保存)が有効になります。その状態でプログラムの実行を開始すると、ファイル名を入力するダイアログボックス、圧縮形式を指定するダイアログボックスが順次表示されますので、それぞれ入力と選択を行ってください。圧縮形式は利用可能なものを選択してください。録画はプログラムの実行を中断するまで継続します。

作成されたファイルはWindows標準のメディアプレーヤーで再生することができます。

このボタンをクリックすると、解除することはできません。録画を中止するには、ファイル名を入力するダイアログボックスで[Cancel]ボタンをクリックしてください。プログラムは実行されますが、録画はキャンセルされます。

注: 録画中はプログラムの実行速度が低下します。

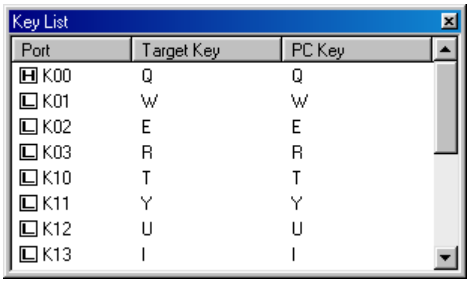
(3) キー入力のシミュレーション

実行中のプログラムがキー入力待ちの場合、[LCD]ウィンドウをアクティブにすることで、PCのキーボードを使用してキー入力をシミュレートすることもできます。

ターゲットのキー名称と使用するポート、PCのキーとの対応はポート設定ファイルで定義します。

この定義内容は、[Key List]ボタンで一覧表示できます。


[Key List]ボタン



[Key List]ウィンドウの表示内容は、左からポート名、ターゲット上のキー名称、PC上のキー名称です。

ここで、ターゲットとPCのキー対応を確認することができます。

左端の記号はポートの状態で、HはポートがHigh、LはLowを示します。キーを押したときの入力レベルは機種ごとのポートの仕様で異なります。

ブレーク時に、ここでキーの状態を設定することもできます。設定したキー入力状態は、その後のプログラム実行時も、ユーザがそのキーを操作するまで保持されます。これにより、ステップ実行時もキー入力状態を保持しておくことができます。

キーを押した状態に設定した場合、キーが入力ポートに直接接続されているものであれば、その入力ポートの状態はアクティブレベルに保持されますが、キーマトリクスの場合はそのキーが交差する出力ポートがアクティブレベルになった場合にのみ入力ポートもアクティブレベルになります。

(4) SVDのシミュレーション

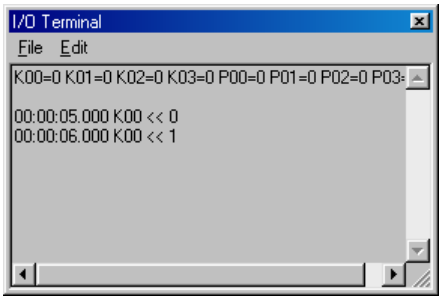
このスライダによりSVDの検出レベルを17段階に変化させることができます。

(5) シリアル/汎用ポートの入出力、A/D変換のシミュレーション

[I/O]ボタンで[I/O Terminal]ウィンドウを表示させ、IOTファイルを読み込むことにより、シリアルインタフェースの入出力、汎用ポートの入出力およびA/D変換をシミュレートすることができます。詳細は次項を参照してください。

[I/O]ボタン

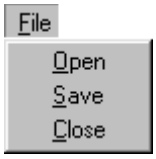
3.6.4 [I/O Terminal]ウィンドウ



[I/O Terminal]ウィンドウは、汎用入出力ポートおよびシリアルインタフェースの入出力、A/D変換をシミュレートして入出力状態を表示するI/Oテキストターミナル機能を提供します。

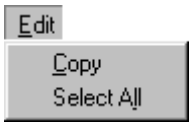
(1)メニュー

Fileメニュー



Open　IOTファイルを読み込みます。
Save　ウィンドウに表示されたログをテキストファイルに保存します。
Close　IOTファイルを閉じます(I/Oシミュレーションを中止します)。

Editメニュー



Copy　ウィンドウの選択範囲をクリップボードへコピーします。
Select All　ウィンドウの全てのログを選択します。

※ウィンドウ内の右クリックでも、Copy/Select Allメニューを表示できます。

(2)IOTファイル

IOTファイルは、以下の情報を記述したテキストファイルで、汎用のエディタにより作成します。

a. 入出力状態を監視する汎用ポート(Rxx、Pxx)の指定
b. 汎用ポート(Kxx、Pxx)への入力タイミングと入力レベルの指定
c. A/D変換器の電源電圧および基準電圧の指定
d. A/D入力電圧の指定
e. シリアルインタフェースの入力データの指定

例:

```
[General I/O]
watch P00, P01, P02, P10, P11, P20  ←a
5.0 K00=1                           ←b   先頭の数値はリセットからの秒数
5.5 K00=0                                1 / H = High, 0 / L = Low
8.0 K00=H
8.5 K00=L
[A/D Converter]
AVdd=3.0                            ←c   数値は電圧値(V)
AVss=0.2
AVref=2.9
[A/D CH2]
2.40,2.39,2.38,2.37,2.36            ←d   数値は電圧値(V)
2.00, 2.10, 2.20, 2.30
0.5
0.65
0.80
[Serial CH1]
"aBcDeFg"                           ←e   文字列または16進数で入力データを指定
0x25,0x20,0x41
```

このファイルを読み込んで、各入力端子の状態をシミュレートします。ファイルの詳細については"3.13 IOTファイル(.iot)"を参照してください。

(3)入出力シミュレーションの開始と終了

入出力シミュレーションは次の操作で開始します。

1. [I/O Terminal]ウィンドウを表示します。
2. [File]メニューの[Open]によってIOTファイルを読み込みます。
3. リセットからターゲットプログラムを実行します。

ターゲットプログラム実行中でもIOTファイルの読み込みは可能ですが、入出力シミュレーションを行うにはリセットすることが必要です。
入出力シミュレーションを終了するには、[File]メニューから[Close]を選択します。

(4)シミュレーションとログの表示

指定入出力ポートの状態が変化(1→0または0→1)した場合、シリアルインタフェースがデータの入出力を行った場合、A/D変換器がアナログ信号を入力した場合、その情報をウィンドウに表示します。
例:

```
P00=1 R01=1                              ┐
AVdd=3.000 AVss=0.200 AVref=2.900        ┘ 初期データ

00:00:01.029 S2 << 31h(1)                ┐
00:00:02.033 S2 << 32h(2)                │
00:00:03.036 S1 >> 33h(3)                │
00:00:04.040 S1 >> 34h(4)                │
00:00:05.000 P30 << 0 P31 << 0           │ ログデータ
00:00:06.016 AD0 << 2.400                │
00:00:07.020 AD0 << 2.390                │
00:00:08.023 AD0 << 2.380                │
00:00:09.027 AD0 << 2.370                │
00:00:05.000 R01 >> 1                    ┘
```

リセット実行直後に、初期データが表示されます。表示される内容は、IOTファイルの[General I/O]セクションにある"Watch"文で指定した汎用入出力ポートの初期値、および[A/D Converter]セクションで指定した"AVdd"、"AVss"、"AVref"の値です。これらを指定していない場合は表示されません。
以下、シミュレーション実行中に入出力したデータが表示されます。各行の先頭に表示される数値はリセット実行からの経過時間(時分秒)です。"<<"は入力、">>"は出力を意味しています。
各ポートのデータと処理内容は以下のとおりです。

汎用入出力ポート:

Watch文で指定したポートが入力または出力が変化すると、ポート名(Kxx、Pxx、Rxx)と変化後の値(0または1)を表示します。入力の変化はIOTファイルに記述してあるタイミングで発生します。このとき、割り込みが許可されている場合は設定されている割り込み条件に従って入力割り込みが発生します。

A/D 変換器:

ターゲットプログラムがA/D変換を行うごとにIOTファイルに記述されているデータを上(左)から順に読み込んで、A/D変換のシミュレーションを行います。このとき、ファイルから読み込んだチャンネル番号(ADx)と入力電圧値(V)がウィンドウに表示されます。割り込みが許可されている場合、入力を開始してから一定時間(サンプリング時間 + A/D変換時間)経過後に割り込みが発生します。
データを指定しなかったチャンネルを変換した場合、入力はレベル0の電圧(AVssと等価)として読み出されます。また、指定したデータ数よりも多くA/D変換を行った場合は、最後に指定したデータが繰り返し読み出されます。AVssよりも小さな値を指定した場合、レベル0(AVssと等価)として読み出され、AVrefよりも大きな値を指定した場合は最大レベル(AVrefと等価)として読み出されます。

シリアルインタフェース:

ターゲットプログラムがシリアルインタフェースの入出力を行うと、1バイトごとにそのチャンネル番号(S1、S2)と入力値/出力値(16進表記・ASCIIキャラクタ表示)を表示します。入力時は、IOTファイルに記述されている入力データが上(左)から順次読み出されます。割り込みが許可されている場合はシリアルインタフェースの割り込みは、入力サンプリング終了のタイミングで割り込みが発生します。

(5) ログの保存

[I/O Terminal]ウィンドウに表示されたログは、[File]メニューの[Save]でテキストファイルとして保存することができます。

また、[I/O Terminal]ウィンドウ内をドラッグして選択した範囲を、[Edit]メニューの[Copy]でクリップボードにコピーできますので、エディタなどで他の書類にペースト可能です。このとき、ログをすべてコピーしたい場合は、[Edit]メニューの[Select All]で全体を選択してからコピーします。

注意事項

・ すでにS1C63/88シミュレータをインストール済みの環境では、シリアルインタフェースに対応したコンポーネントをコンポーネントマップファイル(.cmp)に追加する必要があります。シリアルインタフェース未サポート機種を使用する場合、コンポーネントマップファイルにNullDev.bmcを追加してください。
また、A/D変換器を使用する場合も、同様にコンポーネントマップファイルを修正してください。

例: 63666.cmp

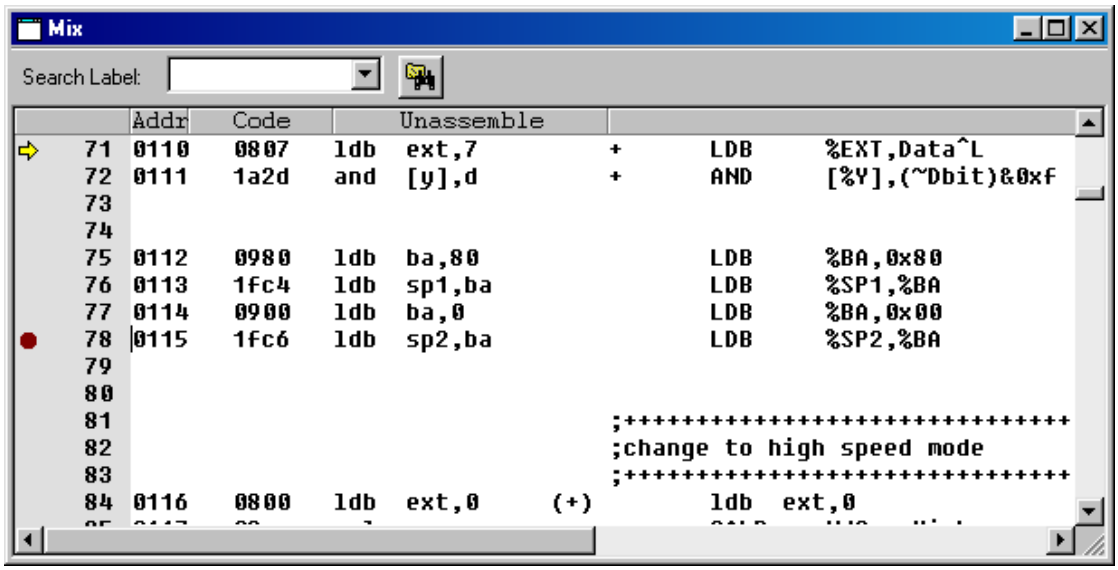
```
[Internal]
CPU=CPU.bmc
LCD=LcdDrv63.bmc
K/P/R port=KPRport.bmc
SVD=SVD63.bmc
Sound=Sound63.bmc
Serial=Serial63666.bmc ← 追加
Adc=Adc63.bmc          ← 追加
```

例: 63458.cmp

```
[Internal]
CPU=CPU.bmc
LCD=LcdDrv63.bmc
K/P/R port=KPRport.bmc
SVD=SVD63.bmc
Sound=Sound63.bmc
Serial=Serial63.bmc ← 追加
Adc=NullDev.bmc     ← 追加
```

・ 入出力シミュレーションを行う時刻は、シミュレータのインストラクションサイクルから計算されますので、実機(実時間)と若干異なります。

3.6.5 [Source]ウィンドウ



[Source]ウィンドウは、次の(1)～(3)の内容を表示します。また、ブレークポイントの設定や、文字列/ラベルの検索もウィンドウ上で行えます。

(1) プログラムの逆アセンブルおよびソースコード
3種類の表示形式が選択できます。


[Mixボタン]

1. ミックス表示モード([Mix]ボタンまたはmコマンド入力)
アドレス、コード、逆アセンブル内容、対応するソース行番号とソースを表示します。(上図)


[Sourceボタン]

2. ソース表示モード([Source]ボタンまたはscコマンド入力)
対応するソース行番号とソースを表示します。


[Unassembleボタン]

3. 逆アセンブル表示モード([Unassemble]ボタンまたはuコマンド入力)
デバッガ起動時の表示形式で、アドレス、コード、逆アセンブル内容を表示します。

注: [Source]ウィンドウが開いている場合、m、sc、uコマンドは表示内容を更新します。[Source]ウィンドウが閉じていた場合、プログラムコードは[Command]ウィンドウに表示されます。

64Kのアドレス空間すべてのプログラムコードをスクロールさせて参照できます。ブレーク時は次に実行されるアドレスの行が表示されるように表示内容が更新され、先頭に"矢印"を表示します。
スクロールはスクロールバーまたは矢印キーによって行います。コマンドによって指定位置から表示させることもできます。

※ソース行番号とソースの表示
ソース行番号とソースは、ソース表示用のデバッグ情報を含むIEEE-695形式のアブソリュートオブジェクトファイルを読み込んだ場合に表示可能です。その中でも、アセンブラで-gオプションを指定したソースのみが表示されます。

※表示の更新
プログラムをロードして実行(g、gr、s、n、rstコマンド)するか、プログラムメモリの内容を変更(a(as)、pe、pf、pmコマンド)すると表示内容が更新されます。この場合、現在のPCが示すアドレスが[Source]ウィンドウ内に表示されるように表示が更新されます。また、表示形式を変更しても更新されます。

(2) カレントPC

現在のPC(プログラムカウンタ)が示すアドレスの行は先頭に"矢印"を表示します。(図のアドレス 0x0110)

(3) PCブレークポイント

ブレークポイントに設定されたアドレスの行は、先頭に赤の"●"マークを表示します。(図のアドレス 0x0115)

(4) カーソル位置でブレーク設定

ブレークポイントを設定したいアドレスの行(ソースのみの行は不可)にカーソルを置きます。そこで、

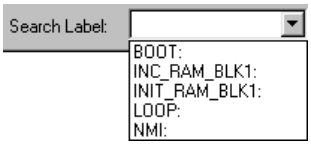
[Break]ボタン

[Break]ボタンをクリックすると、そのアドレスがPCブレークポイントに設定されます。PCブレークポイントに設定されたアドレスの行で同じ操作をすると、そのブレークポイントは解除されます。

[Go to Cursor]ボタン

[Go to Cursor]ボタンをクリックすると、プログラムが現在のPCから実行を開始し、カーソルのある行の実行後にブレークします。

(5) ラベルと文字列の検索



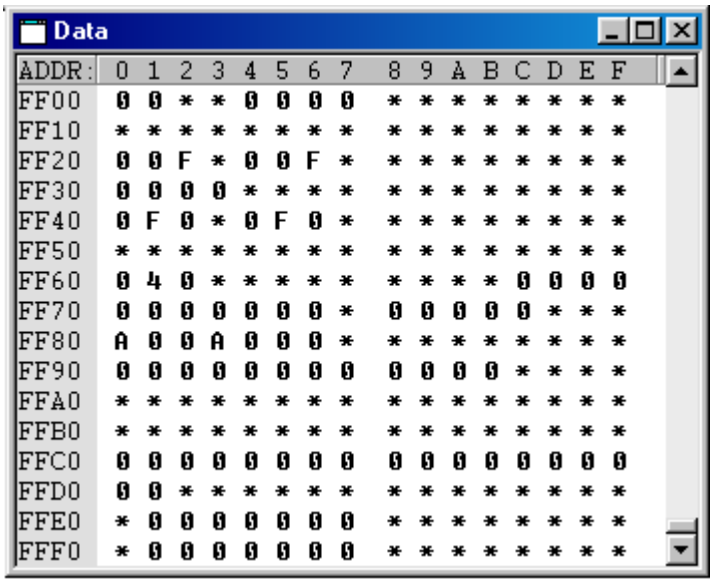
[Search Label]プルダウンボックス

[Source]ウィンドウ上の[Search Label]プルダウンボックスで、ソース内に定義されているラベル位置に表示を移動することができます。

[Find]ボタン

また、[Find]ボタンで任意の文字列をソース内で検索することができます。

3.6.6 [Data]ウィンドウ



(1) メモリ内容の表示

データメモリのダンプ結果を16進数で表示します。
表示領域は64Kワードのデータメモリ全体(RAM、データROM、I/O)で、0x0000から0xffffまですべてのアドレスの内容をスクロールにより表示可能です。各機種のマップされないアドレスの内容は"*"となります。

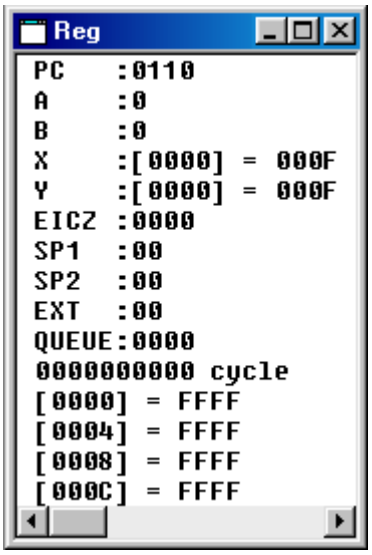
※表示の更新

メモリ内容をコマンド(de、df、dmコマンド)で変更すると、[Data]ウィンドウの表示内容が更新されます。また、プログラムを実行(g、gr、s、n、rstコマンド)した場合も更新されます。
これ以外で、最新の内容を表示させるには、ddコマンドを実行するか、垂直スクロールバーをクリックしてください。

(2) データメモリ内容の直接変更

[Data]ウィンドウ上で、データメモリを直接変更することができます。変更するデータの直前にカーソルを置くか、データをダブルクリック後、16進の数値(0～9、a～f)を入力してください。そのアドレスのデータが変更されます。カーソルは次のアドレスのデータに移動し、連続的なデータ変更を可能にしています。

3.6.7 [Register]ウィンドウ



(1) レジスタ内容

PC、Aレジスタ、Bレジスタ、Xレジスタとそれが示すメモリ、Yレジスタとそれが示すメモリ、フラグ(E、I、C、Z)、スタックポインタ(SP1、SP2)、EXTレジスタ、QUEUEレジスタの内容を表示します。

(2) 実行サイクルカウンタ

CPUがリセットされてからの実行サイクル数または実行時間を積算し、表示します。

(3) 監視データ

本デバッガでは、RAM上の4ヵ所のアドレスを指定して、その内容を監視することができます。
この4つの監視データアドレス(指定アドレスから4ワードずつ)の内容を表示します。起動時は0、4、8、C番地が監視データアドレスとして初期設定されます。左からアドレス順に並んでいます。

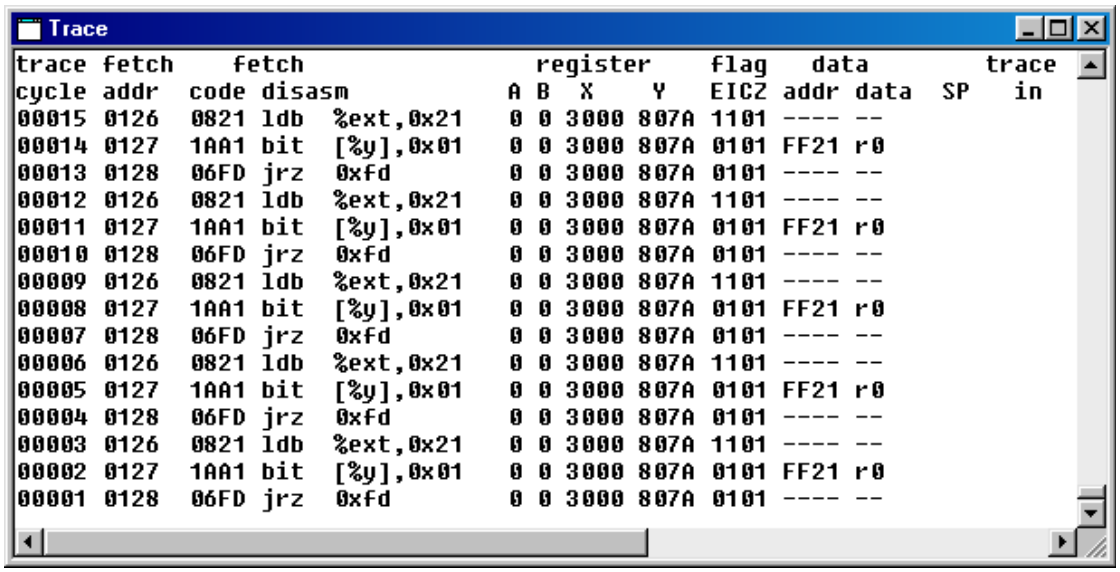
※表示の更新

レジスタダンプ時(rdコマンド)、監視データアドレス設定時(dwコマンド)、レジスタデータ変更時(rsコマンド)、CPUリセット時(rstコマンド)、プログラムの実行(g、gr、s、nコマンド)終了後に更新されます。

(4) レジスタ内容の直接変更

[Register]ウィンドウ上で、レジスタの内容を直接変更することができます。変更するデータを選択(ハイライト)後、16進の数値(0～9、a～f)を入力し[Enter]キーを押してください。そのレジスタの内容が変更されます。

3.6.8 [Trace]ウィンドウ



tmコマンドでトレース機能をONに設定すると、それ以降のプログラム実行時にトレース情報を採取します。トレース用のバッファは8192命令分の容量があり（容量を越えた分は先頭から上書き）、この中に記録した情報を[Trace]ウィンドウに表示することができます。
表示されるトレース内容は、次のとおりです。

- トレースサイクル数
- フェッチアドレス
- フェッチコードと逆アセンブル内容
- レジスタの内容（A、B、X、Yおよびフラグ）
- メモリのアクセス内容（アドレス、R/W、データおよびSP1/SP2）

[Trace]ウィンドウは、tsコマンドによるトレースデータの検索結果の表示にも使用します。

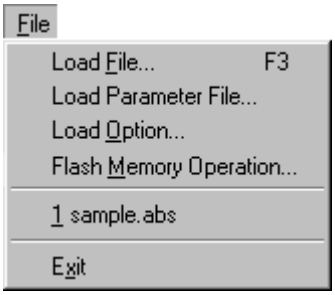
※表示の更新

ターゲットプログラムの実行により、[Trace]ウィンドウの内容はクリアされます。プログラム実行中は、[Trace]ウィンドウのスクロールやサイズ変更操作は受け付けられません。
プログラムの実行を中断すると、[Trace]ウィンドウは実行中にトレースした最新のデータを表示します。指定サイクルから表示させるには、tdコマンドを使用します。

3.7 メニュー

ここでは、メニューバーの概要を説明します。
シミュレータのメニューバーには8つのメニュー項目があり、頻繁に使用するコマンドが設定されています。

[File]メニュー



注: [Flash Memory Operation...]はSim63では無効です。

[Load File...]
IEEE-695形式のアブソリュートオブジェクトファイルを読み込みます。この選択はlfコマンドを実行するのと同じ働きがあります。

[Load Parameter File...]
パラメータファイルを読み込みます。この選択はparコマンドを実行するのと同じ働きがあります。

[Load Option...]
モトローラS2形式のプログラムファイル、データROM用のデータファイル、オプションHEXファイルを読み込みます。この選択はloコマンドを実行するのと同じ働きがあります。

[Exit]
シミュレータを終了します。この選択はqコマンドを実行するのと同じ働きがあります。

[Run]メニュー

[Go]
現在のPC(プログラムカウンタ)からターゲットプログラムを実行します。この選択はgコマンドを実行するのと同じ働きがあります。

[Go to Cursor]
現在のPCから、[Source]ウィンドウのカーソル位置(その行のアドレス)までターゲットプログラムを実行します。この選択はg <address>コマンドを実行するのと同じ働きがあります。このメニュー項目を選択するには、[Source]ウィンドウを開き、ブレークするアドレスの行をクリックしておく必要があります。なお、クリックによるブレークアドレスの選択は、実コードのある行のみ有効で、ソースのみの行に対しては無効です。

[Go from Reset]
CPUをリセット後、プログラム開始アドレス(0x110)からターゲットプログラムを実行します。この選択はgrコマンドを実行するのと同じ働きがあります。

[Step]
現在のPCからターゲットプログラムを1ステップ実行します。この選択はsコマンドを実行するのと同じ働きがあります。

[Next]
現在のPCからターゲットプログラムを1ステップ実行します。実行命令がcalr、calz、int命令の場合は、次のアドレスにリターンするまでを1ステップとみなし、それらのサブルーチンのステップをすべて実行します。この選択はnコマンドを実行するのと同じ働きがあります。

[Command File...]
コマンドファイルを読み込み、記述されているコマンドを実行します。この選択はcom、cmwコマンドを実行するのと同じ働きがあります。

[Reset CPU]
CPUをリセットします。この選択はrstコマンドを実行するのと同じ働きがあります。

[Break]メニュー

[Breakpoint Set...]
PCブレークポイントをダイアログボックスを使用して設定/解除します。この選択はbpコマンドを実行するのと同じ働きがあります。

[Data Break...]
データブレーク条件をダイアログボックスを使用して設定/解除します。この選択はbdコマンドを実行するのと同じ働きがあります。

[Register Break...]
レジスタブレーク条件をダイアログボックスを使用して設定/解除します。この選択はbrコマンドを実行するのと同じ働きがあります。

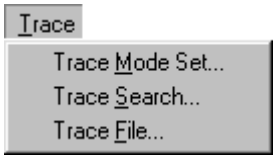
[Sequential Break...]
シーケンシャルブレーク条件をダイアログボックスを使用して設定/解除します。この選択はbsコマンドを実行するのと同じ働きがあります。

[Stack Break...]
スタックブレーク条件をダイアログボックスを使用して設定します。この選択はbspコマンドを実行するのと同じ働きがあります。

[Break List]
設定されているすべてのブレーク条件を表示します。この選択はblコマンドを実行するのと同じ働きがあります。

[Break All Clear]
すべてのブレーク条件を解除します。この選択はbacコマンドを実行するのと同じ働きがあります。

[Trace]メニュー

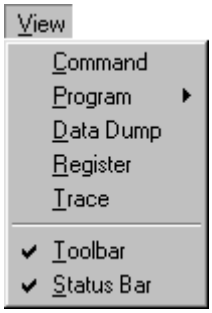


[Trace Mode Set...]
トレースモードをON/OFFします。この選択はtmコマンドを実行するのと同じ働きがあります。

[Trace Search...]
トレースメモリ内のトレース情報を検索します。検索条件はダイアログボックスで指定します。この選択はtsコマンドを実行するのと同じ働きがあります。

[Trace File...]
[Trace]ウィンドウに表示したトレース情報の指定範囲をファイルに保存します。この選択はtfコマンドを実行するのと同じ働きがあります。

[View]メニュー



[Command]
[Command]ウィンドウをアクティブにします。

[Program]

[Source]ウィンドウを開いてアクティブにします。[Source]ウィンドウはプログラムを現在のPCアドレスから、サブメニューで選択した表示モードで表示します。この選択はu、sc、mコマンドを実行するのと同じ働きがあります。

[Data Dump]
[Data]ウィンドウを開いてアクティブにします。[Data]ウィンドウはデータメモリの内容をメモリの先頭から表示します。

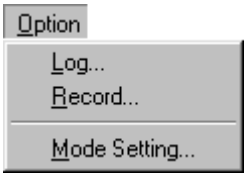
[Register]
[Register]ウィンドウを開いてアクティブにします。[Register]ウィンドウは各レジスタの内容を表示します。

[Trace]
[Trace]ウィンドウを開いてアクティブにします。[Trace]ウィンドウはトレースメモリの内容を表示します。

[Toolbar]
ツールバーの表示/非表示を切り換えます。

[Status Bar]
ステータスバーの表示/非表示を切り換えます。

[Option]メニュー

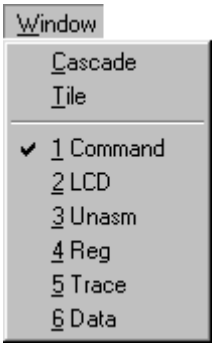


[Log...]
ログ出力のON/OFFを切り換えます。この選択はlogコマンドを実行するのと同じ働きがあります。

[Record...]
実行コマンドのファイルへの記録を制御します。この選択はrecコマンドを実行するのと同じ働きがあります。

[Mode Setting...]
シミュレータの動作モードを設定します。この選択はmdコマンドを実行するのと同じ働きがあります。

[Window]メニュー

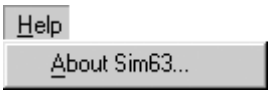


[Cascade]
開いているウィンドウを斜めに整列させます。

[Tile]
開いているウィンドウを縦横に整列させます。

このメニューには、現在開いているウィンドウ名が表示されます。いずれかを選択すると、そのウィンドウがアクティブになります。

[Help]メニュー



[About Sim63...]
シミュレータのアバウトダイアログボックスを表示します。

3.8　ツールバー

シミュレータのツールバーには15個のボタンがあり、頻繁に使用するコマンドが設定されています。

[Key Break]ボタン
ターゲットプログラムの実行を強制的にブレークします。この機能はプログラムが永久ループ状態になった場合などにブレークさせることができます。

[Load File]ボタン
IEEE-695形式のオブジェクトファイルを読み込みます。この選択はlfコマンドを実行するのと同じ働きがあります。

[Load Parameter]ボタン
パラメータファイルを読み込みます。この選択はparコマンドを実行するのと同じ働きがあります。

[Load Option]ボタン
モトローラS2形式のプログラムファイル、データROM用のデータファイル、オプションHEXファイルを読み込みます。この選択はloコマンドを実行するのと同じ働きがあります。

[Source]ボタン
[Source]ウィンドウをソース表示モードに切り換えます。[Source]ウィンドウが閉じている場合は開きます。scコマンドを実行したのと同じ機能があります。

[Unassemble]ボタン
[Source]ウィンドウを逆アセンブル表示モードに切り換えます。[Source]ウィンドウが閉じている場合は開きます。uコマンドを実行したのと同じ機能があります。

[Mix]ボタン
[Source]ウィンドウの表示をミックス(逆アセンブル&ソース)表示モードに切り換えます。[Source]ウィンドウが閉じている場合は開きます。mコマンドを実行したのと同じ機能があります。

[Go]ボタン
現在のPC(プログラムカウンタ)からターゲットプログラムを実行します。この選択はgコマンドを実行するのと同じ働きがあります。

[Go to Cursor]ボタン
現在のPCから、[Source]ウィンドウのカーソル位置(その行のアドレス)までターゲットプログラムを実行します。この選択はg <address>コマンドを実行するのと同じ働きがありますこのボタンを選択するには、[Source]ウィンドウを開き、ブレークするアドレスの行をクリックしておく必要があります。なお、クリックによるブレークアドレスの選択は、実コードのある行のみ有効で、ソースのみの行に対しては無効です。

[Go from Reset]ボタン
CPUをリセット後、プログラム開始アドレス(0x110)からターゲットプログラムを実行します。この選択はgrコマンドを実行するのと同じ働きがあります。

[Step]ボタン
現在のPCからターゲットプログラムを1ステップ実行します。この選択はsコマンドを実行するのと同じ働きがあります。

[Next]ボタン
現在のPCからターゲットプログラムを1ステップ実行します。実行命令がcalr、calz、int命令の場合は、次のアドレスにリターンするまでを1ステップとみなし、それらのサブルーチンのステップをすべて実行します。この選択はnコマンドを実行するのと同じ働きがあります。

[Reset]ボタン
CPUをリセットします。この選択はrstコマンドを実行するのと同じ働きがあります。

[Break]ボタン
[Source]ウィンドウ上のカーソルが置かれたアドレスに対し、ブレークポイントの設定と解除を行うのに使用します。[Source]ウィンドウが開いているときのみ有効です。なお、クリックによるブレークアドレスの選択は、実コードのある行のみ有効で、ソースのみの行に対しては無効です。

[Help]ボタン
シミュレータのアバウトダイアログボックスを表示します。

3.9 コマンド実行方法

デバッグ機能はすべてデバッグコマンドによって実行できます。ここでは、コマンドを実行させる方法を説明します。

3.9.1 コマンドのキーボード入力

[Command]ウィンドウを選択してください([Command]ウィンドウ上をクリック)。その中の最終行にプロンプト">"が表示され、その後にカーソルが点滅していればコマンドが入力できる状態にあります。
そこに、デバッグコマンドを入力してください。コマンドは大文字でも、小文字でも受け付けます。

● コマンド入力の一般形

＞コマンド [パラメータ [パラメータ... パラメータ]] ↲

・コマンドとパラメータ間にはスペースが必要です。
・パラメータ間にはスペースが必要です。

入力ミスの修正には矢印キー、[Back Space]キー、[Delete]キーが使用できます。
最後に[Enter]キーを入力すると、そのコマンドを実行します(ガイダンス付きのコマンドは、表示に従って必要なデータを入力した時点で実行されます)。
入力例:

```
>g↲                    (コマンドのみの入力)
>com test.cmd↲         (コマンドとパラメータの入力)
```

● ガイダンス付きのコマンド入力

パラメータが指定されないと実行できないコマンドや、既存のデータを変更するコマンドはコマンドのみの入力でガイダンスモードとなります。ガイダンスが表示されますので、そこにパラメータを入力してください。
入力例:

```
>lf↲
File name   ? test.abs↲        ...ガイダンスに従ってデータ(アンダーライン部)を入力
>
```

- パラメータ入力が必須のコマンド
  上記例のlfコマンドはアブソリュートオブジェクトファイルを読み込むコマンドです。このような、パラメータの入力が必須のコマンドはパラメータを入力後、[Enter]キーを押すと実行されます。複数のパラメータを持つコマンドでは、次のガイダンスが表示されますので、最後のパラメータまで順次入力してください。いずれかのガイダンスで[Enter]キーのみを入力すると、そのコマンドはキャンセルされ実行されません。

- 既存のデータを確認して置き換えるコマンド
  メモリやレジスタを1つずつ書き換えるコマンドではガイダンスをスキップ(その内容を変更しない)、1つ前のガイダンスに戻す、および途中で入力を終了することができます。
  [Enter]キー ... 入力をスキップ
  [^]キー 1つ前のガイダンスに戻す
  [q]キー ..... 入力を終了する

  入力例:

```
>de↲                          ...データメモリを変更するコマンド
Data enter address ? :0↲      ...開始アドレスを入力
0000   A:1↲                   ...0x0000番地を1に変更
0001   A:^↲                   ...1つ前のアドレスに戻す
0000   1:0↲                   ...0x0000番地の入力をし直す
0001   A:↲                    ...[Enter]キーのみの入力で0x0001番地をスキップ
0002   A:↲
0001   A:q↲                   ...入力を終了
>
```

● パラメータの数値データ形式

パラメータとして入力する数値は、ほとんどのコマンドが16進数のみを受け付けます。
ただし、一部のコマンドのパラメータは、10進数または2進数で指定します。

数値として有効な文字は次のとおりです。
16進数: 0～9、a～f、A～F、*
10進数: 0～9
2進数:　0、1、*
(*はデータパターン指定でマスクするビットに使用します。)

● シンボルによる指定

アドレスを指定する場合、ソース内で定義されているシンボル(ラベル)を使用することができます。その場合、ロードしたアブソリュートオブジェクトファイルがデバッグ情報を含んでいる必要があります。シンボルは、次のように使用します。

グローバルシンボル: @<シンボル名>　　例: `@RAM_BLK1`
ローカルシンボル:　@<シンボル名>@<ソースファイル名>　例: `@LOOP@main.s`

パラメータの入力例については、各コマンドの説明を参照してください。

● [Enter]キーによる連続実行

以下に示すコマンドは、一度実行した後に[Enter]キーのみを入力することにより、前回と同じ実行を繰り返す、あるいは前回の表示内容に引き続いて表示を行うことができます。

実行系:　g (go)、s (step)、n (next)、com (execute command file)
表示系:　sc (source)、m (mix)、u (unassemble)、dd (data memory dump)、td (trace data display)、
　　sy (symbol list)、ma (map information)

他のコマンドを実行すると、連続実行機能は終了します。

3.9.2 メニュー、ツールバーからの実行

3.7項、3.8項に示したように、メニューとツールバーには頻繁に使用するコマンドが登録されています。メニューコマンドを選択するか、ツールバーボタンをクリックするだけで、指定のコマンドを実行できます。表3.9.2.1に登録されているコマンドの一覧を示します。

表3.9.2.1　メニュー、ツールバーで指定可能なコマンド

コマンド	機能	メニュー	ボタン
lf	IEEE-695オブジェクトファイルのロード	[File I Load File...]	
par	パラメータファイルのロード	[File I Load Parameter File...]	
lo	モトローラS2ファイルのロード	[File I Load Option...]	
g	プログラムの連続実行	[Run I Go]	
g <address>	プログラムを<address>まで連続実行	[Run I Go to Cursor]	
gr	CPUリセット後、プログラムの連続実行	[Run I Go after Reset]	
s	ステップ実行	[Run I Step]	
n	ステップ&サブルーチンスキップ	[Run I Next]	
com	コマンドファイルのロード、実行	[Run I Command File...]	–
cmw	コマンドファイルウェイト付き実行	[Run I Command File...]	–
rst	CPUリセット	[Run I Reset CPU]	
bp, bc (bpc)	PCブレークポイント設定/解除	[Break I Breakpoint Set...]	
bd, bdc	データブレーク条件の設定/解除	[Break I Data Break...]	–
br, brc	レジスタブレーク条件の設定/解除	[Break I Register Break...]	–
bs, bsc	シーケンシャルブレーク条件の設定/解除	[Break I Sequential Break...]	–
bsp	スタック領域指定	[Break I Stack Break...]	–
bl	ブレーク条件の表示	[Break I Break List]	–
bac	全ブレーク条件の解除	[Break I Break All Clear]	–
tm	トレースモードの設定	[Trace I Trace Mode Set...]	–
ts	トレース情報の検索	[Trace I Trace Search...]	–
tf	トレース情報のファイルへの保存	[Trace I Trace File...]	–
u	逆アセンブル表示	[View I Program I Unassemble]	
sc	ソース表示	[View I Program I Source Display]	
m	ミックス表示	[View I Program I Mix Mode]	
dd	データメモリダンプ	[View I Data Dump]	–
rd	レジスタ値の表示	[View I Register]	–
td	トレース情報の表示	[View I Trace]	–
log	ログ出力ON/OFF	[Option I Log...]	–
rec	実行コマンドの記録	[Option I Record...]	–
md	モード設定	[Option I Mode Setting...]	–

3.9.3 コマンドファイルによる実行

一連のデバッグコマンドを記述したコマンドファイルを読み込んで、それらのコマンドを実行させることができます。

● コマンドファイルの作成

コマンドファイルはエディタでテキストファイルとして作成してください。
ファイル名の拡張子は".cmd"とします。
コマンドファイルは、recコマンドによっても作成できます。recコマンドを使用すると、シミュレータで実際に実行したコマンドをコマンドファイルに記録することができます。

● コマンドファイル例

次の例は、プログラムファイルを読み込み、ブレークポイントを設定して実行させるためのコマンド群です。
例: ファイル名=start.cmd

```
lo test.fsa
lo test.ssa
lf test.abs
bp 0004d7
g
```

コマンドファイルにはガイダンスモードに対応した記述も可能です。その場合は、ガイダンス入力の各項目ごとに改行して記述してください。

● コマンドファイルの読み込み/実行

コマンドファイルの実行用に、comコマンドとcmwコマンドが用意されています。
comコマンドは指定されたファイルを読み込み、その中のコマンドを記述順に指定された間隔(0～256秒)で実行します。cmwコマンドも同様ですが、個々のコマンドはmdコマンドで指定された間隔(1～256秒)で実行されます。
例: `com start.cmd`

```
cmw test.cmd
```

コマンドファイルに記述されたコマンドは[Command]ウィンドウに表示されます。

● 制限事項

コマンドファイル内から別のコマンドファイルを読み込むことも可能です。ただし、最大5階層までに制限されます。6階層目のcom/cmwコマンドが現れるとエラーとなり、それ以後の実行を中止します。

3.9.4 ログファイル

実行したコマンドと実行結果を、テキスト形式のログファイルとして保存することができます。これによって、後からデバッグの手順と内容を確認することができます。
保存の対象となるのは[Command]ウィンドウに表示された内容です。

● コマンド例

```
>log tst.log
```

logコマンドでログモードに設定後(出力開始後)は、logコマンドがトグル動作(ログモード/出力ON↔通常モード/出力OFF)となりますので、必要な部分のみの出力が簡単に行えます。

● ログモードでの[Command]ウィンドウの表示

ログモードでは、[Command]ウィンドウに表示される内容が通常の場合と異なります。

(1)各ウィンドウが開いている場合のコマンド実行時
(ウィンドウに結果が表示されるコマンドをそのウィンドウが開いている状態で実行した場合)
通常モード: 表示先のウィンドウの内容が更新されます。[Command]ウィンドウに実行結果は表示されません。
ログモード: ウィンドウに表示される情報と同等の内容が[Command]ウィンドウにも表示されます。ただし、ウィンドウのスクロール操作、ウィンドウを開いたことによる表示内容は、[Command]ウィンドウには表示されません。

(2)各ウィンドウが閉じている場合のコマンド実行時
表示先のウィンドウが閉じている場合、ログモード/通常モードにかかわらず実行結果が[Command]ウィンドウに表示されます。

3.10 デバッグ機能

ここでは、デバッグ機能の概要を機能別に説明します。

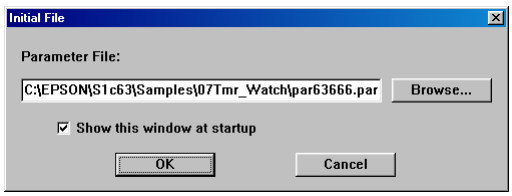
3.10.1 ファイルの読み込み

表3.10.1.1にシミュレータが読み込むファイルの一覧と読み込みコマンドを示します。

表3.10.1.1　ファイルと読み込みコマンド一覧

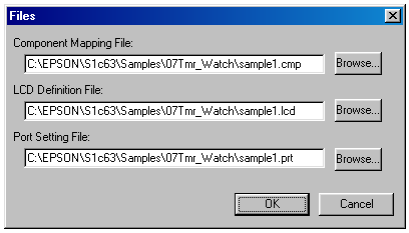
ファイル	拡張子	生成ツール	コマンド	メニュー	ボタン
1. パラメータファイル	.par	–	par	[File I Load Parameter File...]	
2. LCDパネル定義ファイル	.lcd	LcdUtil	–	–	–
3. コンポーネントマップファイル	.cmp	–	–	–	–
4. ポート設定ファイル	.prt	–	–	–	–
5. シミュレータプロジェクトファイル	.spj	–	par	[File I Load Parameter File...]	
6. IEEE-695アブソリュート オブジェクトファイル	.abs	lk63	lf	[File I Load File...]	
7. プログラムHEXファイル	.hsa .lsa	hx63	lo	[File I Load Option...]	
8. データROM HEXファイル	.csa				
9. ファンクションオプションファイル	.fsa	fog63xxx or winfog			
10. セグメントオプションファイル	.ssa	sog63xxx or winsog			
11. コマンドファイル	.cmd	–	com/cmw	[Run I Command File...]	–

1～4はシミュレータが起動するために必要です。
シミュレータの最初の起動時は、次のダイアログボックスで1のパラメータファイルまたは5のシミュレータプロジェクトファイルを選択する必要があります。



パラメータファイルまたはシミュレータプロジェクトファイルの名称をテキストボックスに入力するか、[Browse...]ボタンで選択してください。
[Show this window at startup]チェックボックスをオフにすると、このダイアログボックスは次回の起動時から表示されなくなり、同じファイルが選択されます。

パラメータファイルをロードすると、その時点でシミュレータがリセットされます。パラメータファイルで設定されたメモリマップ情報はmaコマンドで表示させることができます。
パラメータファイルを選択すると2～4のファイルを選択する次のダイアログボックスが表示されます。



シミュレータプロジェクトファイルを選択した場合は、その中で2～4のファイルが指定されているためダイアログボックスは表示されません。
コンポーネントマップファイル、LCDパネル定義ファイル、ポート設定ファイルをそれぞれのテキストボックスに入力するか、[Browse...]ボタンで選択してください。これらのファイルを一度選択すると、次回からはテキストボックスに同じファイル名が表示され、[OK]のクリックのみで済みます。

lfコマンドはIEEE-695形式のアブソリュートオブジェクトファイル(.abs)をロードします。ソース表示とシンボルを使用してデバッグを行うためには、この形式でデバッグ情報を含んだファイルを読み込む必要があります。また、ソースファイルも作成時のディレクトリに置いておく必要があります。

loコマンドはモトローラS2形式のプログラム/データファイル(.hsa、.lsa、.cas)とオプションファイル(.fsa、.ssa)をロードします。この形式のプログラムファイルを読み込んだ場合は、逆アセンブル表示のみとなります。

コマンドファイルは3.9.3項で説明したとおりです。

3.10.2 ソース表示およびシンボリックデバッグ機能

本シミュレータは、アセンブリソースを表示させたデバッグが可能です。また、シンボル名も使用したアドレスの指定も行えます。

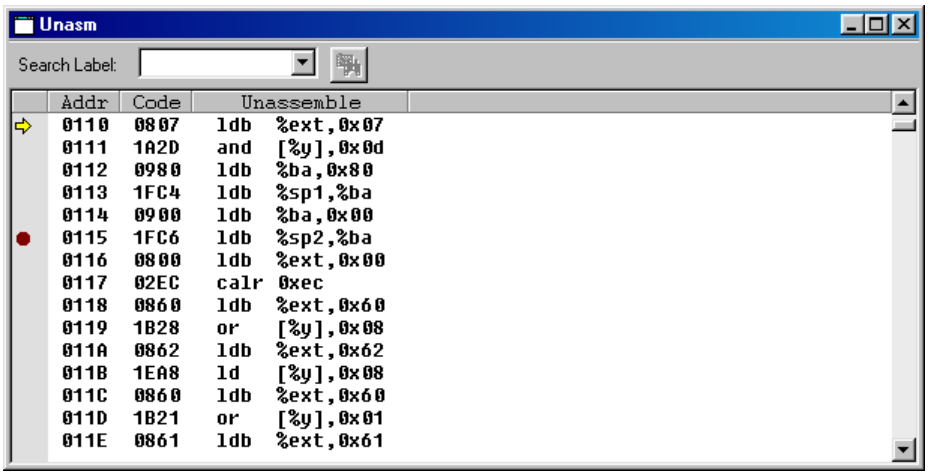
● プログラムコードの表示

[Source]ウィンドウは指定された表示モードでプログラムを表示します。表示モードは、逆アセンブル表示モード、ソース表示モード、ミックス表示モードの3種類から選択できます。

表3.10.2.1　表示モード切り換えコマンド

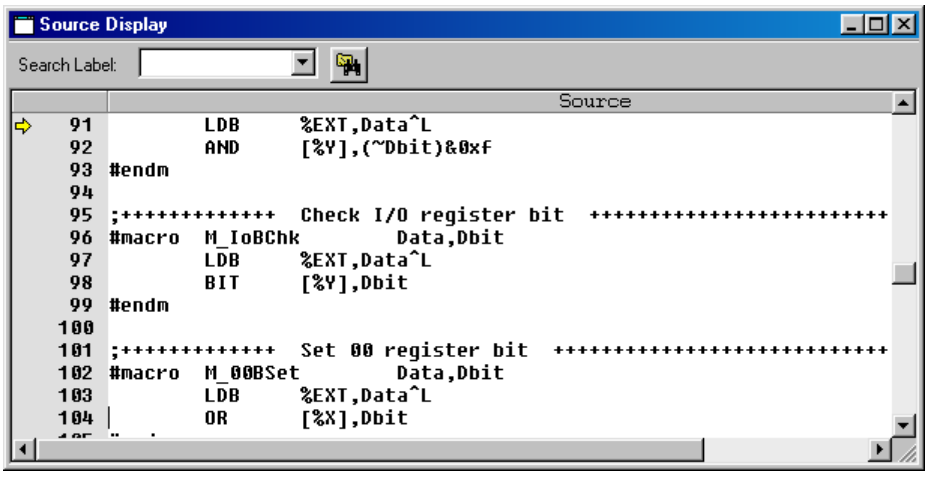
機能	コマンド	メニュー	ボタン
逆アセンブル表示モード	u	[View I Program I Unassemble]	
ソース表示モード	sc	[View I Program I Source Display]	
ミックス表示モード	m	[View I Program I Mix Mode]	

(1) 逆アセンブル表示モード



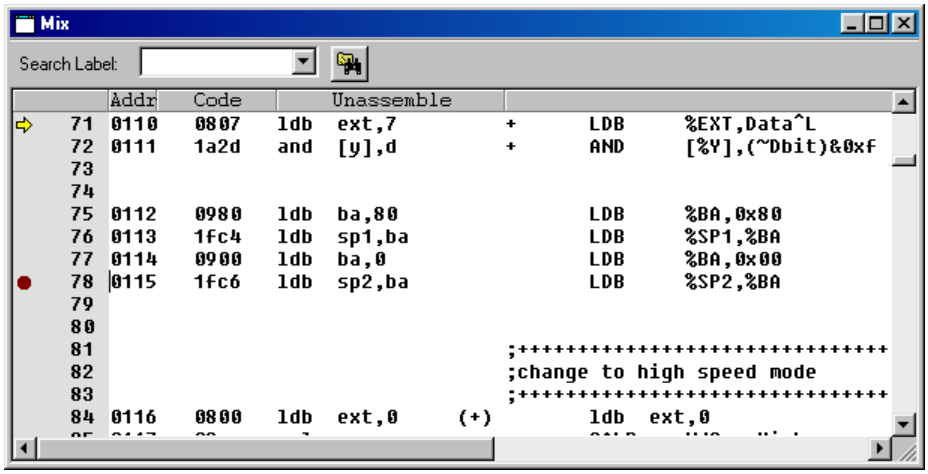
このモードでは、プログラムコードをニーモニックに逆アセンブルして表示します。

(2) ソース表示モード



このモードでは、現在のPCアドレス上のコードを含むソースがエディタと同様の形式で表示されます。このモードは、ソースデバッグ情報を含むアブソリュートオブジェクトファイルをロードしている場合にのみ指定可能です。

(3) ミックス表示モード



このモードでは、プログラムの逆アセンブル内容とソースの両方がアブソリュートリストファイルと同様の形式で表示されます。このモードは、ソースデバッグ情報を含むアブソリュートオブジェクトファイルをロードしている場合にのみ指定可能です。

表示内容については、"3.6.5 [Source]ウィンドウ"を参照してください。

● シンボル参照

IEEE-695形式のオブジェクトファイルを読み込んでデバッグを行う場合、ソースファイルで定義されたシンボルを使用してアドレスを指定することができます。パラメータに<address>を持つコマンドを[Command]ウィンドウ上で入力する際、あるいはアドレスをダイアログで指定する際に使用することができます。

(1) グローバルシンボルの参照

.global擬似命令および.comm擬似命令でグローバル宣言されたシンボル(ラベル)は、次のように指定します。

@<シンボル>

指定例:

```
>m @BOOT
>de @RAM_BLK1
```

(2) ローカルシンボルの参照

定義されたソースファイル内でのみ使用しているローカルシンボル(ラベル)は、次のように指定します。

@<シンボル>@<ファイル名>

ファイル名は、シンボルが定義されているソースファイル名(.s)です。

指定例:

```
>bp @SUB1@test.s
```

(3) シンボルリストの表示

デバッグ中のプログラムで使用しているすべてのシンボルと定義されたアドレスを、[Command]ウィンドウに表示させることができます。

表3.10.2.2　シンボルリスト表示コマンド

機能	コマンド	メニュー	ボタン
シンボルリストの表示	sy	–	–

3.10.3 プログラム、データ、レジスタの表示と変更

シミュレータはプログラムメモリ、データメモリ、レジスタに対する操作機能を持っています。各メモリ領域はパラメータファイルで与えられるマップ情報に従ってシミュレータに設定されます。

● プログラムメモリ領域の操作

プログラムメモリ領域に対しては以下の操作が行えます。

表3.10.3.1　プログラムメモリ操作コマンド

機能	コマンド	メニュー	ボタン
コードの入力/変更	pe	–	–
インラインアセンブル	a (as)	–	–
指定領域の書き換え	pf	–	–
指定領域のコピー	pm	–	–

(1) コードの入力/変更

16進データを入力して、指定アドレスのプログラムコードを書き換えます。

(2) インラインアセンブル

ニーモニックを入力して、指定アドレスのプログラムコードを書き換えます。

(3) 指定領域の書き換え

指定した領域を指定したコードですべて書き換えます。

(4) 指定領域のコピー

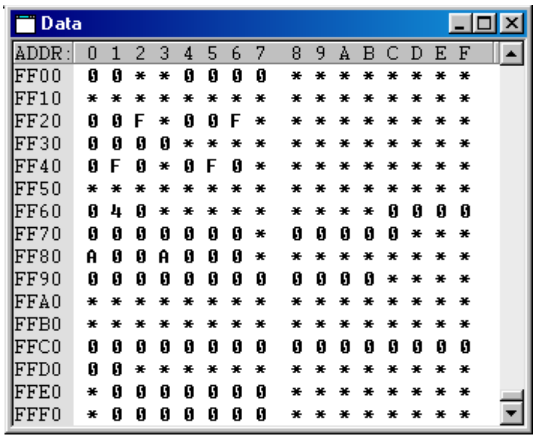
指定した領域の内容を、別の領域にコピーします。

● データメモリ領域の操作

データメモリ領域(RAM、データROM、表示メモリ、I/Oメモリ)に対しては以下の操作が行えます。

表3.10.3.2　データメモリ領域操作コマンド

機能	コマンド	メニュー	ボタン
データメモリダンプ	dd	[View I Data Dump]	–
データの入力/変更	de	–	–
指定領域の書き換え	df	–	–
指定領域のコピー	dm	–	–



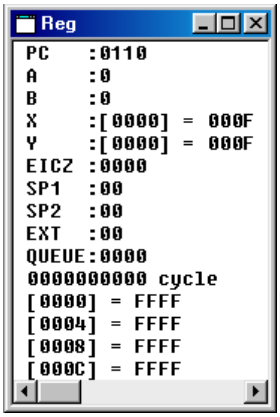
(1) データメモリダンプ
データメモリの内容を16進ダンプ形式で表示します。[Data]ウィンドウが開いていれば[Data]ウィンドウの内容を更新し、開いていなければ[Command]ウィンドウに表示します。

(2) データの入力/変更
16進データを入力して、指定アドレスのデータを書き換えます。[Data]ウィンドウ上で直接変更することもできます。

(3) 指定領域の書き換え
指定した領域を指定したデータですべて書き換えます。

(4) 指定領域のコピー
指定した領域の内容を、別の領域にコピーします。

(5) メモリの監視
連続した4ワード分のメモリ領域を4ヵ所、監視データアドレスとして登録することができます。登録した監視データは[Register]ウィンドウ上で確認できます。初期設定では、0番地、4番地、8番地、C番地が監視データアドレスとなっています。



表示されているメモリの内容は左端が指定アドレスのデータ、右端が4ワード分上位のアドレスのデータです。

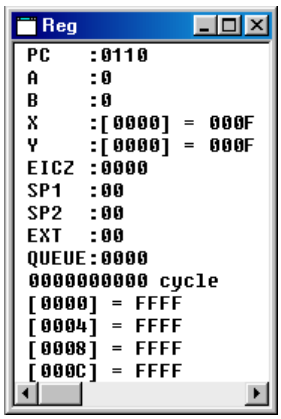
監視データ

● レジスタの操作

レジスタに対しては以下の操作が行えます。

表3.10.3.3　レジスタ操作コマンド

機能	コマンド	メニュー	ボタン
レジスタの表示	rd	[View I Register]	–
レジスタ値の変更	rs	–	–



(1) レジスタの表示
レジスタの内容は[Register]ウィンドウまたは[Command]ウィンドウに表示させることができます。
レジスタ: PC、A、B、Xおよび[X]、Yおよび[Y]、F、SP1、SP2、EXT、QUEUE

(2) レジスタ値の変更
上記レジスタの内容を任意の値に設定できます。
レジスタ値は[Register]ウィンドウ上で直接変更することもできます。

3.10.4 プログラムの実行

シミュレータにはターゲットプログラムを連続実行およびシングルステップ実行させる機能があります。

● 連続実行

(1) 連続実行の種類

2種類の連続実行機能があります。

- 現在のPCアドレスから連続実行
- CPUをリセット後にプログラム開始アドレス(0x0110)から連続実行

表3.10.4.1　連続実行コマンド

機能	コマンド	メニュー	ボタン
現在のPCアドレスから連続実行	g	[Run I Go]	
		[Run I Go to Cursor]	
CPUをリセット後に連続実行	gr	[Run I Go from Reset]	

(2) 連続実行の停止

連続実行コマンド(g <アドレス>)によって、その実行中にのみ有効なテンポラリブレークアドレスを2ヵ所まで指定することができます。

テンポラリブレークアドレスは[Source]ウィンドウで指定することもできます(1ヵ所のみ)。[Source]ウィンドウ上のブレークさせるアドレスの行にカーソルを置いて[Go to Cursor]ボタンをクリックすると、プログラムは現在のPCから実行を開始し、カーソル位置の命令を実行前にブレークします。

このテンポラリブレーク以外では、実行中のプログラムは次のいずれかの要因によってブレークするまで停止しません。

- ブレーク設定コマンドで設定したブレーク条件が成立
- [Key Break]ボタンのクリック
- マップブレーク等の発生

 [Key Break]ボタン　※プログラムが停止しない場合は、このボタンで強制ブレークさせることができます。

(3) LCDパネル、外部入出力のシミュレーション

シミュレータ起動時にターゲットに合わせて作成したコンポーネントマップファイル、LCDパネル定義ファイル、ポート設定ファイルを読み込んでおくことにより、[LCD]ウィンドウはプログラムの制御に従った表示を行います。コンピュータのキーボードを使用してキー入力もシミュレートできます。これらの機能については、"3.6.3 [LCD]ウィンドウ"を参照してください。また、[I/O Terminal]ウィンドウからIOTファイルを読み込んで、シリアル/汎用ポートの入出力、A/D変換をシミュレートすることもできます。詳細については、"3.6.4 [I/O Terminal]ウィンドウ"を参照してください。

● シングルステップ実行

(1) シングルステップ実行の種類

2種類のシングルステップ実行機能があります。

・ 全命令をシングルステップ実行(STEP)
命令の種類によらずPCに従ってシングルステップで実行します。

・サブルーチン以外をシングルステップ実行(NEXT)
calr、calz、int命令を、リターン命令で次のステップに戻るまでを1ステップとして実行します。他の命令は、通常のシングルステップ実行と同様です。

どちらの場合も現在のPCから実行します。

表3.10.4.2　シングルステップコマンド

機能	コマンド	メニュー	ボタン
全命令をシングルステップ実行	s	[Run	Step]	
サブルーチン以外をシングルステップ実行	n	[Run	Next]	

コマンド入力による実行では、実行するステップ数を最大65,535ステップまで指定することができます。メニューコマンド、ツールバーでは1ステップずつ実行します。
以下の場合には、シングルステップ実行が指定のステップ数の実行前に終了します。
• [Key Break]ボタンのクリック
• マップブレーク等の発生

PCブレーク、データブレーク等のユーザ設定ブレークでは停止しません。

 [Key Break]ボタン　※プログラムが停止しない場合は、このボタンで強制ブレークさせることができます。

(2) ステップ動作中の表示

シミュレータの初期設定では、次のように表示を更新します。
[Register]ウィンドウの表示内容は各ステップごとに更新されます。[Register]ウィンドウが閉じている場合は、その内容が[Command]ウィンドウに表示されます。この表示モードを、指定ステップ数の最終ステップでのみ更新するように、mdコマンドで切り換えることもできます。
[Source]ウィンドウおよび[Data]ウィンドウの表示は指定ステップ数の実行終了時に更新されます。

(3) HALT、SLEEP状態と割り込み

halt命令またはslp命令を実行するとCPUはスタンバイモードとなり、その解除には割り込みが必要です。
シミュレータでは、シングルステップ動作用に外部割り込みの許可/禁止モードが設定されています。

表3.10.4.3　外部割り込みモード

	許可モード	禁止モード
外部割り込み	割り込みを処理する	割り込みを処理しない
halt、slp命令	halt命令として実行 外部割り込み、または[Key Break]ボタンで処理を継続	halt、slp命令をnop命令に置き換えて実行

シミュレータの初期設定で、割り込み禁止モードに設定されます。mdコマンドによって割り込み許可モードに設定することもできます。

(4) ステップ動作中のキー入力シミュレーション

[LCD]ウィンドウの[Key List]ボタンで表示される[Key List]ウィンドウ上でキー入力状態が設定でき、その設定状態はステップ動作中も保持されます。

● 実行サイクルの測定

(1) 実行サイクルカウンタ

シミュレータは31ビットの実行サイクルカウンタを内蔵しており、プログラムを実行したバスサイクル数を測定することができます。
実行サイクルカウンタで測定可能な最大値は2,147,483,647cycleです。

(2) 測定結果の表示

測定結果は[Register]ウィンドウに表示されます。プログラム実行中はクリアされ、実行終了後に更新されます。
[Register]ウィンドウが閉じている場合は、rdコマンドによって[Command]ウィンドウに表示することができます。シングルステップの実行結果としても表示されます。
カウンタの最大値を越えた場合は"over flow"を表示します。

(3) ホールドモードとリセットモード

シミュレータの初期設定では、実行サイクルカウンタはホールドモードに設定されます。このモードでは、カウンタがリセットされるまで測定値を積算します。
リセットモードはmdコマンドで設定することができ、プログラムの実行ごとにカウンタがリセットされます。
連続実行ではgコマンドの入力によるプログラムの実行開始時にリセットされ、その実行終了(ブレーク)までを測定します。(grコマンド実行時も同様ですが、CPUのリセットによりカウンタもリセットされるため、ホールドモードでもリセットモードと同じ結果となります。)
シングルステップ実行では、 s またはnコマンドの入力による実行開始時にリセットされ、指定ステップ数の実行終了までを測定します。1ステップのみの指定またはツールバーボタン/メニューコマンドによる実行では、1ステップごとにリセットされます。

(4) 実行サイクルカウンタのリセット

実行サイクルカウンタは次の場合にリセットされます。

- rstコマンド、[Run]メニュー/[Reset CPU]、[Reset]ボタンによってCPUをリセットした場合
- grコマンド、[Run]メニュー/[Go from Reset]を実行した場合
- 実行サイクルカウンタのモードをmdコマンドにより切り換えた場合
  (ホールドモード↔リセットモード)
- リセットモードでプログラムの実行を開始した場合

● CPUのリセット

表3.10.4.4　CPUリセットコマンド

機能	コマンド	メニュー	ボタン
CPUリセット	rst	[Run	Reset CPU]	
CPUをリセット後に連続実行	gr	[Run	Go from Reset]	

CPUはgrコマンドの実行時、およびrstコマンドの実行によりリセットされます。
CPUのリセットによる初期設定内容は以下のとおりです。

(1) CPUの内部レジスタ

PC　...0x0110
A、B　...0xa
X、Y、QUEUE　...0xaaaa
F　...0b0000
SP1、SP2、EXT　...0xaa

(2) 実行サイクルカウンタを0に設定

(3) [Source]ウィンドウ、[Register]ウィンドウを再表示

PCが0x0110に設定されるため、そのアドレスから再表示します。
[Register]ウィンドウを上記の設定で再表示します。

データメモリの内容は変更されません。

3.10.5 ブレーク機能

ターゲットプログラムは次の要因により実行を中断します。
- ブレークコマンドの条件成立
- [Key Break]ボタン
- マップブレーク等の発生

● コマンドによるブレーク機能

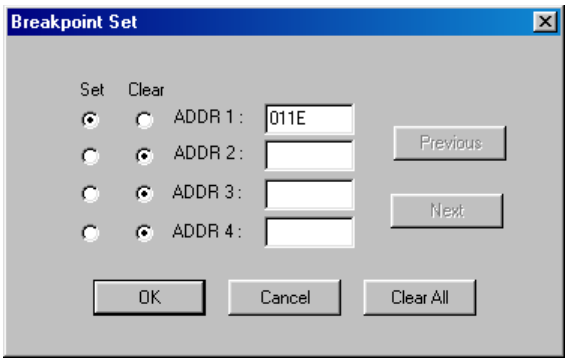
シミュレータはコマンドによってブレーク条件が設定できる5種類のブレーク機能を持っています。それぞれ、条件が成立すると実行中のプログラムはブレークします。

(1) PCによるブレーク機能

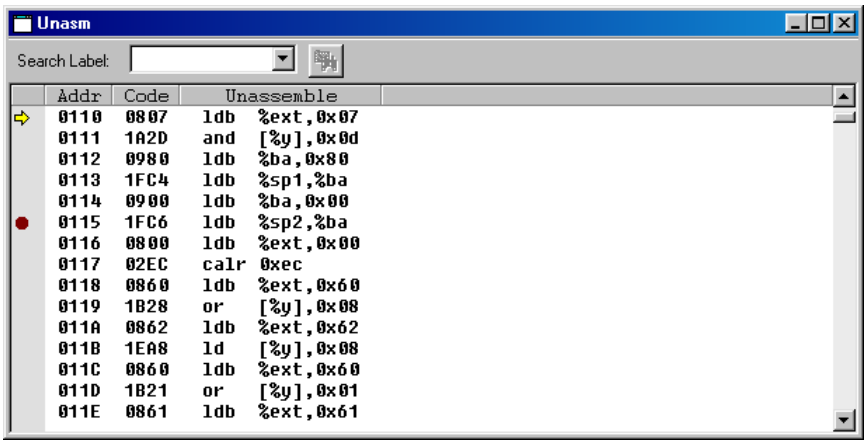
PCが設定したアドレスに一致するとブレークする機能です。プログラムは、そのアドレスの命令を実行前にブレークします。PCブレークポイントは複数のアドレスに設定できます。

表3.10.5.1　ブレークポイント設定コマンド

機能	コマンド	メニュー	ボタン
ブレークポイントの設定	bp	[Break I Breakpoint Set...]	
ブレークポイントの解除	bc (bpc)	[Break I Breakpoint Set...]	



PCブレークポイントに設定されたアドレスは、[Source]ウィンドウ上の行の先頭に●が表示されます。



[Break]ボタンを使用して簡単にブレークポイントの設定と解除を行うこともできます。
[Source]ウィンドウ上で、ブレークさせるアドレスの行をクリック(カーソルを移動)し、[Break]ボタンをクリックします。その行の先頭に●マークが表示され、ブレークポイントに設定されます。そのアドレスはブレークポイントリストに登録されます。●で始まる行をクリックして[Break]ボタンをクリックすると、ブレーク設定が解除され、アドレスがブレークポイントリストから削除されます。

※ 連続実行コマンド(g)で指定可能なテンポラリブレークアドレスは、ブレークポイントリストに設定されたアドレスには影響を与えません。

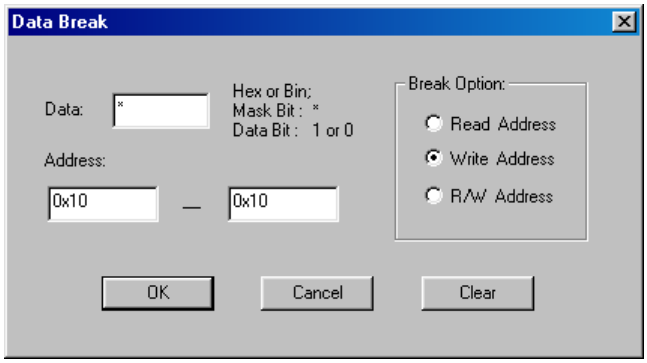
(2) データブレーク機能

データブレークは、指定したデータメモリ領域内をアクセスした場合にブレークを発生させる機能です。アクセスを監視するメモリ領域のほかに、読み出し、書き込みのどちらが行われた場合にブレークさせるか、またそのデータの内容まで指定することができます。読み出し/書き込み条件はマスク可能で、どちらが行われた場合でもブレークさせることができます。同様にデータ条件もビット単位のマスクが可能です。

ブレークは上記の条件を満たす動作が行われたサイクルの終了時に発生します。

表3.10.5.2　データブレーク設定コマンド

機能	コマンド	メニュー	ボタン
データブレーク条件の設定	bd	[Break | Data Break...]	–
データブレーク条件の解除	bdc	[Break | Data Break...]	–



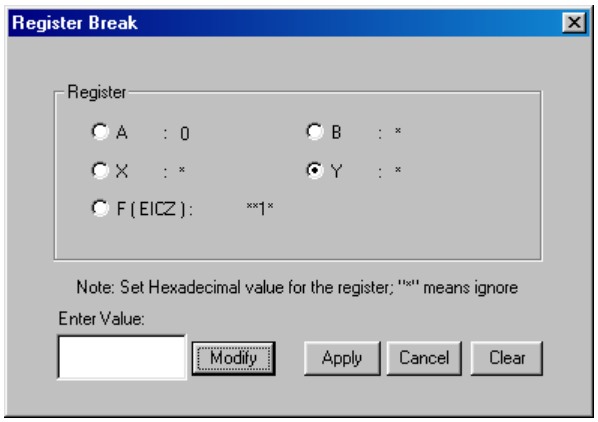
たとえば、アドレスを0x10、データパターンを*(マスク)、読み出し/書き込み条件を書き込みに設定してプログラムを実行すると、プログラムがデータメモリのアドレス0x10にデータの書き込みを行った場合にブレークが発生します。

(3) レジスタブレーク

レジスタブレークは、A、B、F、XおよびYレジスタがそれぞれ指定した値になった場合にブレークを発生させる機能です。各レジスタはマスク(ブレーク条件に含めない設定)が可能です。Fレジスタはビットごとにマスク可能です。ブレークは上記のレジスタがすべて設定条件を満たすように変更された時点で発生します。

表3.10.5.3　レジスタブレーク設定コマンド

機能	コマンド	メニュー	ボタン
レジスタブレーク条件の設定	br	[Break | Register Break...]	–
レジスタブレーク条件の解除	brc	[Break | Register Break...]	–



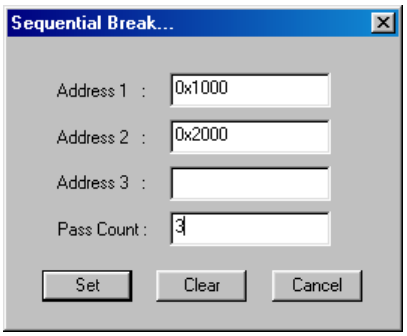
たとえば、Aレジスタのデータを0、Fレジスタ(Cフラグ=1)のデータを"**1*"、他をすべてマスクに設定してプログラムを実行させると、Aレジスタが0になり、かつCフラグが1になった時点でブレークします。

(4) シーケンシャルブレーク機能

シーケンシャルブレークでは、3ヵ所までのブレークアドレスと、その中の最後のアドレスの実行回数を設定できます。プログラムは、設定した順序ですべてのアドレスを通過するとともに、最後に指定したアドレスを指定回数実行し、その後もう一度そのアドレスの命令をフェッチするとブレークします。

表3.10.5.4　シーケンシャルブレーク設定コマンド

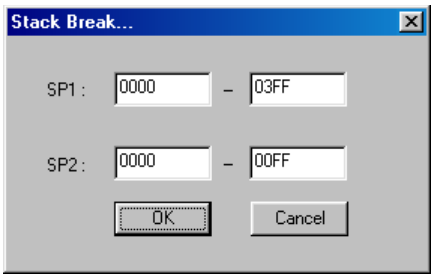
機能	コマンド	メニュー	ボタン
シーケンシャルブレーク条件の設定	bs	[Break	Sequential Break...]	–
シーケンシャルブレーク条件の解除	bsc	[Break	Sequential Break...]	–



たとえば、bsコマンドでアドレスを0x1000番地と0x2000番地の2ヵ所に、実行回数を3回に設定してプログラムを実行させた場合、0x1000番地を1回以上実行した後で0x2000番地を3回実行し、さらにPCが0x2000になると4回目の実行前にブレークします。
実行回数は4095回まで指定できます。

(5) スタック領域外へのアクセス

スタックポインタSP1、SP2によってスタック領域外へのアクセスが行われた場合にブレークします。



この機能を使用するためには、bspコマンドでSP1、SP2領域をあらかじめ設定しておく必要があります。初期値はSP1が0x0～0x3ff、SP2が0x0～0xffとなっています。SP1のアドレスは4ワード単位で指定する必要があります。

表3.10.5.5　スタックブレーク設定コマンド

機能	コマンド	メニュー	ボタン
スタックブレーク条件の設定	bsp	[Break	Stack Break...]	–

● [Key Break]ボタンによる強制ブレーク

[Key Break]ボタンは、プログラムが永久ループまたはスタンバイ(HALT、SLEEP)状態から抜け出せない場合など、実行中のプログラムを強制的に終了させます。


[Key Break]ボタン

● マップブレーク、不当命令ブレーク

プログラム実行中に以下のエラーが発生した場合もブレークします。

(1) 未定義プログラム領域へのアクセス

プログラムメモリマップの未定義領域にアクセスした場合にブレークします。

(2) 未定義データ領域へのアクセス

データメモリマップの未定義領域にアクセスした場合にブレークします。

(3) データROM領域への書き込み

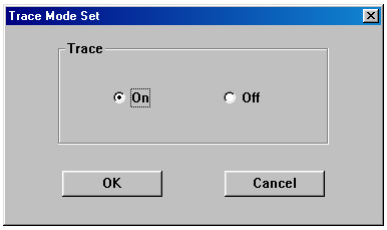
データROM領域に書き込みが行われた場合にブレークします。

3.10.6 トレース機能

シミュレータには、プログラムの実行をトレースする機能があります。
トレース機能は、tmコマンドの設定により有効になります。初期状態では無効になっています。

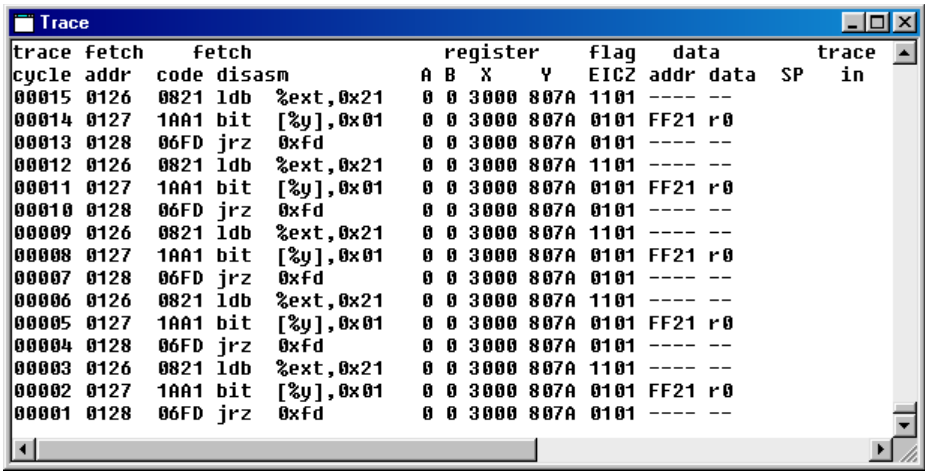
表3.10.6.1　トレースモード設定コマンド

機能	コマンド	メニュー	ボタン
トレースモードの設定/解除	tm	[Trace	Trace Mode Set...]	–



● トレースデータバッファとトレース情報

シミュレータはトレースデータバッファを持っています。プログラムを実行すると各実行命令ごとのトレース情報がこのバッファに取り込まれます。トレースデータバッファは8,192命令分の容量があり、トレース情報がこの容量を越えると、古いデータから上書きしていきます。したがって、トレースデータバッファには常に8,192命令以内のトレース情報が記録されています。プログラムの実行によりトレースデータバッファはクリアされ、新しい実行データをトレースします。



各命令の実行でトレースデータバッファに取り込まれるトレース情報は次のとおりです。[Trace]ウィンドウの表示に対応させて示します。

trace cycle:	トレースサイクル(10進数)トレースメモリ内に最後に取り込まれた情報が00001となります。
fetch addr:	フェッチアドレス(16進数)
fetch code disasm:	フェッチコード(16進表示)と逆アセンブル内容
register:	サイクル実行後のA、B、X、Yレジスタ値(16進数)
flag:	サイクル実行後のE、I、C、Zフラグの状態(2進数)
data:	アクセスしたデータメモリアドレス(16進数)、リード/ライト(データの先頭のrまたはw)、データ(4ビットアクセスの場合1桁の16進数、16ビットアクセスの場合4桁の16進数)
SP:	スタックアクセス(SP1アクセス時が1、SP2アクセス時が2)
trace in:	未使用

● トレース情報の表示と検索

採取したトレース情報はプログラムの実行を中断すると[Trace]ウィンドウに表示されます。[Trace]ウィンドウではスクロールによってトレースデータバッファの全データを見ることができます。コマンドにより指定したサイクルから表示させることもできます。表示内容は前記のとおりです。[Trace]ウィンドウが閉じている場合はコマンドで[Command]ウィンドウに表示することもできます。

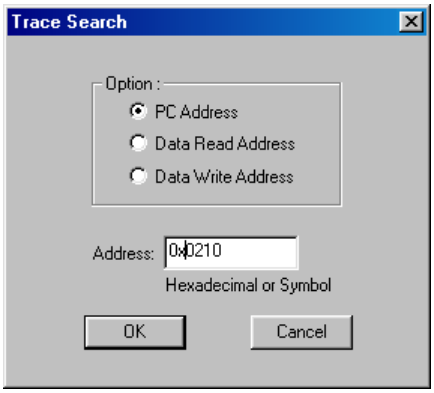
表3.10.6.2　トレース情報表示コマンド

機能	コマンド	メニュー	ボタン
トレース情報の表示	td	[View	Trace]	–

検索条件を指定し、条件に合ったトレース情報のみを表示させることができます。

表3.10.6.3　トレース情報検索コマンド

機能	コマンド	メニュー	ボタン
トレース情報の検索	ts	[Trace	Trace Search...]	–



検索条件は次の3種類から選択できます。
1. プログラムの実行アドレス
2. データを読み出したアドレス
3. データを書き込んだアドレス

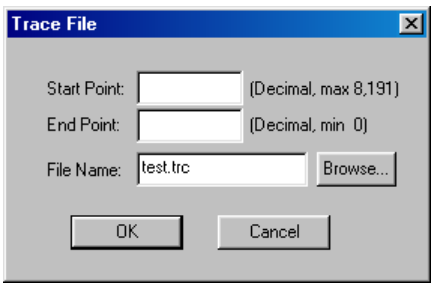
上記条件とアドレスを1ヵ所指定して検索を行います。条件に合ったトレース情報が見つかると、件数を[Command]ウィンドウに表示します。検索データは[Trace]ウィンドウ（閉じている場合は[Command]ウィンドウ）に表示します。

● トレース情報の保存

トレース情報の指定サイクル範囲をファイルに保存することができます。

表3.10.6.4　トレース情報保存コマンド

機能	コマンド	メニュー	ボタン
トレース情報の保存	tf	[Trace	Trace File...]	–



3.11 コマンド一覧

表3.11.1　コマンド一覧

分類	コマンド		機能
プログラム メモリ操作	**a (as)**	[<addr> <mnemonic> [<file name>]]	インラインアセンブル
	pe	[<addr> <code1> [..<code8>]]	プログラムコードの入力
	pf	[<addr1> <addr2> <code>]	プログラム領域のフィル
	pm	[<addr1> <addr2> <addr3>]	プログラム領域のコピー
データ メモリ操作	**dd**	[<addr1> [<addr2>] [{-Bl-Wl-Ll-Fl-D}]]	データメモリダンプ
	de	[<addr. <data1> [..<data16>]]	データの入力
	df	[<addr1> <addr2> <data>]	データ領域のフィル
	dm	[<addr1> <addr2> <addr3>]	データ領域のコピー
	dw	[<addr1> [..<addr4>]]	監視データアドレス設定
レジスタ操作	**rd**		レジスタの表示
	rs	[<reg> <value> [..<reg> <value>]]	レジスタ値の変更 reg={pc|a|b|x|y|f|sp1|sp2|ext|q}
プログラム実行	**g**	[<addr1> [<addr2>]]	カレントPCから連続実行
	gr	[<addr1> [<addr2>]]	CPUリセット後に連続実行
	s	[<step>]	カレントPCからシングルステップ実行
	n	[<step>]	サブルーチン以外をシングルステップ実行
CPUリセット	**rst**		CPUをリセット
ブレーク	**bp**	[<addr1> [..<addr16>]]	ブレークポイントの設定
	bc	[<addr1> [..<addr16>]]	ブレークポイントの解除
	bpc	[<addr1> [..<addr16>]]	
	bd	[<data> {r|w|∗} <addr1> <addr2>]	データブレーク条件の設定
	bdc		データブレーク条件の解除
	br	[<reg> <value> [..<reg> <value>]]	レジスタブレーク条件の設定 reg={pc|a|b|x|y|f|sp1|sp2|ext|q}
	brc		レジスタブレーク条件の解除
	bs	[<pass> <addr1> [<addr2> [<addr3>]]]	シーケンシャルブレーク条件の設定
	bsc		シーケンシャルブレーク条件の解除
	bsp	[<addr1> <addr2> <addr3> <addr4>]	スタック領域指定
	bl		全ブレーク条件の表示
	bac		全ブレーク条件の解除
プログラム表示	**u**	[<addr>]	逆アセンブル表示
	sc	[<addr>]	ソース表示
	m	[<addr>]	ミックス表示
シンボル情報	**sy**	[{$<keyword>|#<keyword>}][/a]	シンボル一覧の表示
ファイルロード	**lf**	[<file name>]	IEEE-695形式オブジェクトファイルのロード
	lo	[<file name>]	モトローラS2形式ファイルのロード
	par	[<file name>]	パラメータファイルのロード
トレース	**tm**	[{on|off}]	トレースモードの設定
	td	[<cycle>]	トレース情報の表示
	ts	[{pc|dr|dw} <addr>]	トレース情報の検索
	tf	[[<cycle1> [<cycle2>]] <file name>]	トレース情報の保存
その他	**com**	[<file name> [<interval>]]	コマンドファイル読み込み/実行
	cmw	[<file name>]	ウェイト付きコマンドファイル読み込み/実行
	rec	[<file name>]	実行コマンドの記録
	log	[<file name>]	ログ出力
	ma		マップ情報の表示
	md	[<option> <num> [..<option> <num>]]	モード設定 option={-fl-ul-il-sl-cl-ill-cm}
	q		デバッガ終了
	?		コマンドusageの表示

3.12 コンポーネントマップファイル(.cmp)

コンポーネントマップファイル(ペリフェラル設定ファイル)は、シミュレータでCPU、キー入力、LCD表示などをシミュレートするために必要な情報を記録したテキストファイルです。各機種の内部周辺回路のデータを記述したファイルが用意されていますので、テキストエディタで必要な情報を追加して使用してください。なお、シミュレータ上で変更したLCDパネルの色などはこのファイルに保存されますので、書込み禁止(ReadOnly)属性を設定しないでください。

例:

```
[Internal]
CPU=CPU.bmc
LCD=LcdDrv63.bmc
K/P/R port=KPRport.bmc
SVD=SVD63.bmc
Sound=Sound63.bmc
Serial=Serial63666.bmc (注1)
Adc=Adc63.bmc (注2)

[Settings]
CpuType=63666
OSC1=32.768KHz
OSC3=1.0MHz

[External]
EXT0=CMulDiv63.bmc,FF80h,FF86h
```

[Internal] 内蔵周辺回路の設定

この部分はユーザが変更することはできません。ただし、すでににS1C63/S1C88シミュレータをインストール済みの環境では、シリアルインタフェースに対応したコンポーネント(注1)をここに追加する必要があります。A/D変換器を使用する場合も同様です(注2)。シリアルインタフェースまたはA/D変換器を搭載していない機種を使用する場合、注1はSerial=NullDev.bmc、注2はAdc=NullDev.bmcとなります。

[Settings] CPUの設定

これらはすべて必須項目です。下記を参考に変更してください。

```
[Settings]          ←Settingsセクション開始
CpuType=63666       ←ファミリ名
OSC1=32.768KHz      ←OSC1クロック周波数
OSC3=1.0MHz         ←OSC3クロック周波数
```

[External] 外部デバイスの設定

添付のファイルに記述された内容は変更しないでください。

```
[External]          ←Externalセクション開始
EXT0=CMulDiv63.bmc,FF80h,FF86h
```

定義形式: Ext<N>=<device>.bmc, <Start_addr>, <End_addr>, <Option>

Ext<N>: Nは0から始まる連番で個々のデバイス番号です。
<device>.bmc: 周辺デバイス定義ファイル名
<Start_addr>: I/Oマップ開始アドレス
<End_addr>: I/Oマップ終了アドレス
<Option>: ペリフェラル固有のオプション

BkLightの設定

バックライトを接続する場合は以下のように記述してください。

```
EXT1=BkLight.bmc, FF31h, FF31h, 1L
```

バックライトを接続したポートのI/Oアドレスを指定します。開始・終了アドレスは同じになります。バックライトを点灯させるビット、およびH/Lのどちらでonにするかをオプションとして指定します。例では1ビット目がLowのときにバックライトを点灯するように定義しています。

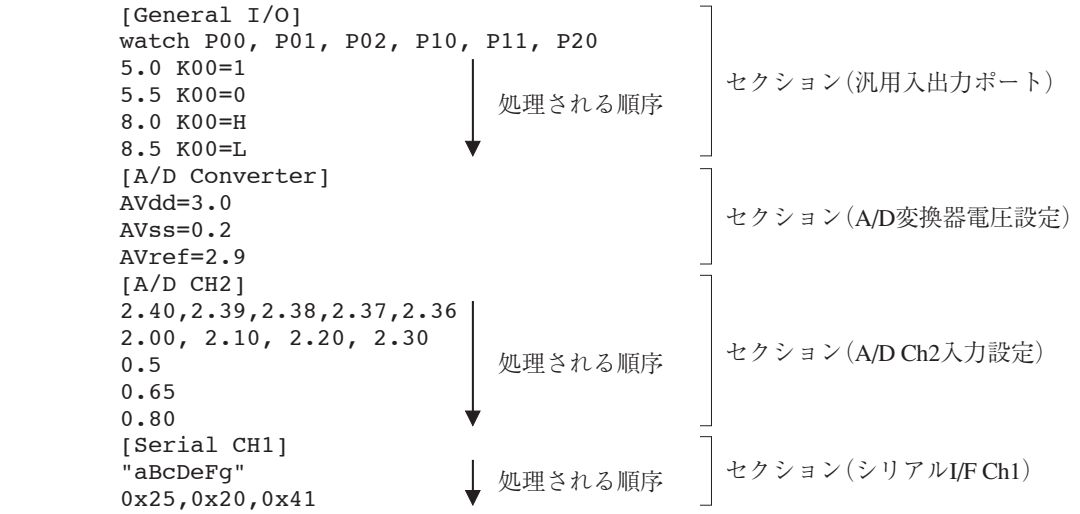
3.13 IOTファイル(.iot)

IOTファイルは、汎用ポートおよびシリアルインタフェースの入出力、A/D変換をシミュレートするための設定および入力データを記述しておくテキストファイルです。これを[I/O Terminal]ウィンドウのメニューで読み込み、入出力シミュレーションを行います。[I/O Terminal]ウィンドウおよび入出力シミュレーションの操作方法については、"3.6.4 [I/O Terminal]ウィンドウ"を参照してください。
入出力シミュレーションを行う場合は、以下の説明に従い、テキストエディタでIOTファイルを作成します。作成したデータは、拡張子を".iot"として保存してください。

ファイルの内容

入力データは[General I/O]、[A/D Converter] のようにセクション単位で記述します。セクションには順序はありませんが、セクション内部の各データは、記述した順に処理されます。使用しない機能のセクションは記述する必要はありません。

例: test.iot



汎用入出力ポート

汎用入出力ポートの設定は、[General I/O]セクションに記述します。

監視するポートの指定

入出力を監視する汎用入出力ポートをwatch文で指定します。

例: `watch P00, P01, P02, P10, P11, P20`

watch文には、入出力状態をチェックするポートの名称(Kxx、Rxx、Pxx)を記述します。ここに記述したポートの入出力がHighからLowまたはLowからHighに変化すると、[I/O Terminal]ウィンドウにログが表示されます。ポートの監視を行わない場合は、記述する必要はありません。

入力の指定

汎用ポートへ入力を行う場合は、入力を行う時刻(秒単位)とポート名、およびポートへの入力レベルを1行単位で記述します。入力シミュレーションは記述した順に上から行われます。

例:
```
5.0 K00=1
8.5 K00=0
+0.7 P02=H, P10=L
```

各行の先頭は、入力を行う時刻(秒単位)を表します。小数点を用いて1msの桁まで設定することができます。基本的に、値はリセットからの経過時間です。なお、先頭に"+"を付けることにより、前回の入力からの経過時間を指定することができます。
入力内容は、"ポート名=入力レベル"の形式で記述します。Highレベル入力はHまたは1、Lowレベル入力はLまたは0で指定します。
上記例の場合、リセット実行から5.0秒後にK00ポートにHigh、8.5秒後にK00ポートにLow、9.2秒後にP02ポートにHigh、同時にP10ポートにLowが入力されます。
入力ポート(Kポート)割り込みが許可されている場合、入力のタイミングで割り込みが発生します。

A/D変換器

A/D変換器の設定は、[A/D Converter]セクションと[A/D CHn]セクションに記述します。

A/D変換器の電圧設定

[A/D Converter]セクションには、変換器の電圧設定を記述します。

例:
```
[A/D Converter]
AVdd=3.0
AVss=0.2
AVref=2.9
```

AVdd、AVss、AVrefの各行は、それぞれA/D変換器の+電源電圧、-電源電圧、基準電圧を表します。電圧値はボルト単位で記述します。値は小数点を用いて1mVの桁まで設定することができます。A/D変換器を使用する場合、この3行を含む[A/D Converter]セクションは必ず記述しなければなりません。

アナログ入力電圧の指定

[A/D CHn]セクションには、A/D変換器への入力電圧を記述します。指定はチャンネル単位に行います。未使用のチャンネルは記述する必要はありません。

例:
```
[A/D CH4]
2.40,2.39,2.38,2.37,2.36
2.00, 2.10, 2.20, 2.30
[A/D CH5]
0.5
0.65
0.80
```

入力電圧はボルト単位で記述します。値は小数点を用いて1mVの桁まで設定することができます。
記述したデータはターゲットプログラムがA/D変換を実行するごとに、上から順に読み込まれます。1行に記述する場合はカンマ(,)で区切ります。この場合は左から順に読み込まれます。
割り込みが許可されている場合、入力を開始してから一定時間(サンプリング時間 + A/D 変換時間)経過後に割り込みが発生します。

データを指定しなかったチャンネルのA/D変換を行った場合は、レベル0の電圧(AVssと等価)が入力されているものとして処理されます。指定したデータ数を超えるA/D変換を行った場合は、最後に指定したデータが繰り返し使用されます。
また、AVssよりも小さな値を指定した場合、レベル0(AVssと等価)として読み出されます。AVrefよりも大きな値を指定した場合、最大レベル(AVrefと等価)として読み出されます。

シリアルインタフェース

シリアルインタフェースの設定は[Serial CHn]セクションに記述します。設定はチャンネル単位で行います。1チャンネルのみを内蔵している機種は"CH1"(チャンネル1)を指定します。未使用のチャンネルは記述する必要はありません。

例:
```
[Serial CH1]
"aBcDeFg"
0x25,0x20,0x41,FER
```

入力データは16進数表記または文字列表記(" "で囲む)のどちらかで指定します。1行に混在させることはできません。記述したデータはシリアルインタフェースによる受信を行うごとに、上から順に読み込まれます。複数の16進データを1行に記述する場合はカンマ(,)で区切ります。この場合は左から順に読み込まれます。
割り込みが許可されている場合、入力サンプリング終了のタイミングで割り込みが発生します。
データを指定しなかったチャンネルで受信を行った場合、割り込みは発生しません。

サンプリングに必要な時間
調歩同期式: I/Oレジスタで選択したクロック源の周期×ビット数(+パリティビット)
クロック同期式(マスタ): I/Oレジスタで選択したクロック源の周期×ビット数
クロック同期式(スレーブ): ユーザ定義の転送レート(bps)

クロック同期式(スレーブ)の場合は、転送レート(bps)を以下のように指定します。データよりも前に指定してください。この設定はクロック同期式(スレーブ)の場合のみ有効です。

例:
```
[Serial CH1]
bps=2400
"aBcDeFg"
0x25,0x20,0x41,FER
```

16進データの代わりに、以下のデータを指定することができます。
FER: フレーミングエラー
PER: パリティエラー
OER: オーバーランエラー

これらのデータは、シリアルインタフェースがエラー割り込みをサポートしている場合のみ有効です。サポートしていない機種の場合は指定しないでください。
これらのデータを読み込んだ際、割り込みが許可されているとエラー割り込みが発生します。

シリアルインタフェースの出力に関しては、特に何も指定する必要はありません。各チャンネルへの出力が行われると、ログが[I/O Terminal]ウィンドウに表示されます。

3.14 シミュレータプロジェクトファイル(.spj)

シミュレータプロジェクトファイルは、使用するパラメータファイル、LCDパネル定義ファイル、コンポーネントマップファイルおよびポート設定ファイルを記述したテキストファイルです。テキストエディタで作成してください。このファイルにより、上記のファイルの選択を一度に行うことができます。
シミュレータの起動時、またはparコマンド([File | Load Parameter File])実行時に表示されるダイアログボックスでパラメータファイルかシミュレータプロジェクトファイルのどちらか一方を選択します。パラメータファイルを選択する場合は、シミュレータプロジェクトファイルを用意しておく必要はありません。
ファイルの内容は次のとおりです。
例: 63666.spj

```
[Settings]
PAR=63666.par       ← パラメータファイル
LCD=63666.lcd       ← LCDパネル定義ファイル
CMP=63666.cmp       ← コンポーネントマップファイル
PRT=63666.prt       ← ポート設定ファイル
```

3.15 制限事項

- 本シミュレータの対応機種につきましてはReadme_J.txtの、"サポート機種および周辺機器"を参照してください。
- 外部デバイスは、モノクロLCDパネル、バックライト、キー、キーマトリクスのみをサポートしています。
- 本シミュレータはインストラクションレベルでシミュレーションを行っています。よってインストラクションサイクルよりも短いサイクルのシミュレーションを行うことはできません。
- インストラクションサイクルを元にタイマをシミュレーションしていますので、実際のハードウェアとはタイミングが同じではありません。
- 以下の機能はサポートしていません。
  1. ブザー出力
  2. TOUT/FOUT出力
  3. D/Aコンバータ、アナログコンパレータ
  4. プログラマブルタイマのイベントカウンタモード
- サウンドジェネレータをシミュレートするには、PCM音源の再生が可能なサウンドカードが必要です。
- PC上で複数のシミュレータを同時に起動してシミュレーションを行うことはできません。
- Kポート同時入力によるリセットを行う際、割り当てた4つのキーを同時に押し下げることによるリセットは実行できません。1+3または2+2の組み合わせでわずかにタイミングをずらして押し下げを行ってください。またICEとは異なり、検定時間を待たずにリセットが行われます。
- 本シミュレータのデバッグ機能はS1C63 Familyのデバッガdb63と互換性があり、メニューやダイアログボックスも基本的にdb63と同じにしてあります。ただし、ICEを使用しないため以下の機能は無効です。
  1. オンザフライ機能
  2. 実行時間の計測（サイクル数は計測可能）
  3. フラッシュメモリ操作機能（lfl、sfl、efl）
  4. カバレッジ機能（cv、cvc）
  5. オプションデータダンプ（od）
  6. シングルディレイトリガトレースおよびPC範囲指定トレース
  7. 外部からのブレーク信号およびトレーストリガ信号の入力

最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_sim63_J.txtを参照してください。
また、最新バージョンのサポート機種については、S5U1C88000QのReadme_J.txtを参照してください。

3.16 シミュレータメッセージ

● ステータスメッセージ一覧

ステータスメッセージ	メッセージ内容
Break by PC break	PCブレークポイントによりブレーク
Break by data break	データブレーク条件によりブレーク
Break by register break	レジスタブレーク条件によりブレーク
Break by sequential break	シーケンシャルブレーク条件によりブレーク
Key Break	[Key Break]ボタンによりブレーク
Break by accessing no map program area	未定義プログラムメモリ領域のアクセスによりブレーク
Break by accessing no map data area	未定義データメモリ領域のアクセスによりブレーク
Break by accessing ROM area	データROM領域への書き込みによりブレーク
Out of SP1 area	SP1スタック領域外に対するスタック操作によりブレーク
Out of SP2 area	SP2スタック領域外に対するスタック操作によりブレーク

● エラーメッセージ一覧

エラーメッセージ	メッセージ内容(エラーが発生するコマンド)
Address out of range, use 0–0xXXXX	指定アドレスがプログラムメモリの有効範囲外 (a/as, pe, pf, pm, sc, m, u, g, gr, bp, bc, bs, tm, ts, cv)
Address out of range, use 0–0xFFFF	指定アドレスがデータメモリの有効範囲外 (dd, de, df, dm, dw, bd, ts)
Cannot load program/ROM data, check ABS file	プログラム/ROMデータのロードに失敗 IEEE-695実行形式以外のファイルが指定された(lf)
Cannot open file	ファイルがオープンできない(lf, lo, com, cmw, log, rec)
Data out of range, use 0–0xF	指定の数値はデータの有効範囲外(de, df)
Different chip type, cannot load this file	指定ファイルは異なるICEパラメータで作成されている(lf)
end address < start address	開始アドレスより小さい終了アドレスが指定された (pf, pm, df, dm, bd, cv)
error file type (extension should be CMD)	".cmd"以外の拡張子を持つコマンドファイルが指定された(com, cmw)
illegal code	入力コードが不正(pe, pf)
illegal mnemonic	S1C63000用の入力ニーモニックが不正(a/as)
Incorrect number of parameters	パラメータ数が不正(全コマンド)
Incorrect option, use -f/-u/-i/-s/-c/-il/-cm	無効なモード設定オプションが指定された(md)
Incorrect r/w option, use r/w/*	不正なR/Wオプションが指定された(bd)
Incorrect register name, use A/B/X/Y/F	無効なレジスタ名が指定された(br)
Incorrect register name, use PC/A/B/X/Y/F/SP1/SP2/EXT/Q	指定されたレジスタ名が無効(rs)
Input address does not exist	未設定ブレークアドレスを解除しようとした(bp)
invalid command	無効なコマンドが入力された(全コマンド)
invalid data pattern	入力データパターンが不正(bd, br)
invalid file name	オプションファイルの拡張子が無効(lo)
invalid value	入力データ、アドレス、シンボルが不正(全コマンド)
Maximum nesting level(5) is exceeded, cannot open file	com/cmwコマンドのネストレベルが制限を越えた(com, cmw)
no such symbol	該当するシンボルがない(シンボル対応の全コマンド)
no symbol information	".abs"ファイルがロードされていないため、シンボルは使用不可(sy)
Number of passes out of range, use 0–4095	シーケンシャルブレークの実行回数指定が有効範囲外(bs)
Number of steps out of range, use 0–65535	指定ステップ数が有効範囲外(s, n)
SP1 address out of range, use 0–0x3FF	指定SP1アドレスが有効範囲外(bsp)
SP2 address out of range, use 0–0xFF	指定SP2アドレスが有効範囲外(bsp)
symbol type error	指定シンボルタイプ(プログラム/データ)が不正 (シンボル対応の全コマンド)

● ワーニングメッセージ一覧

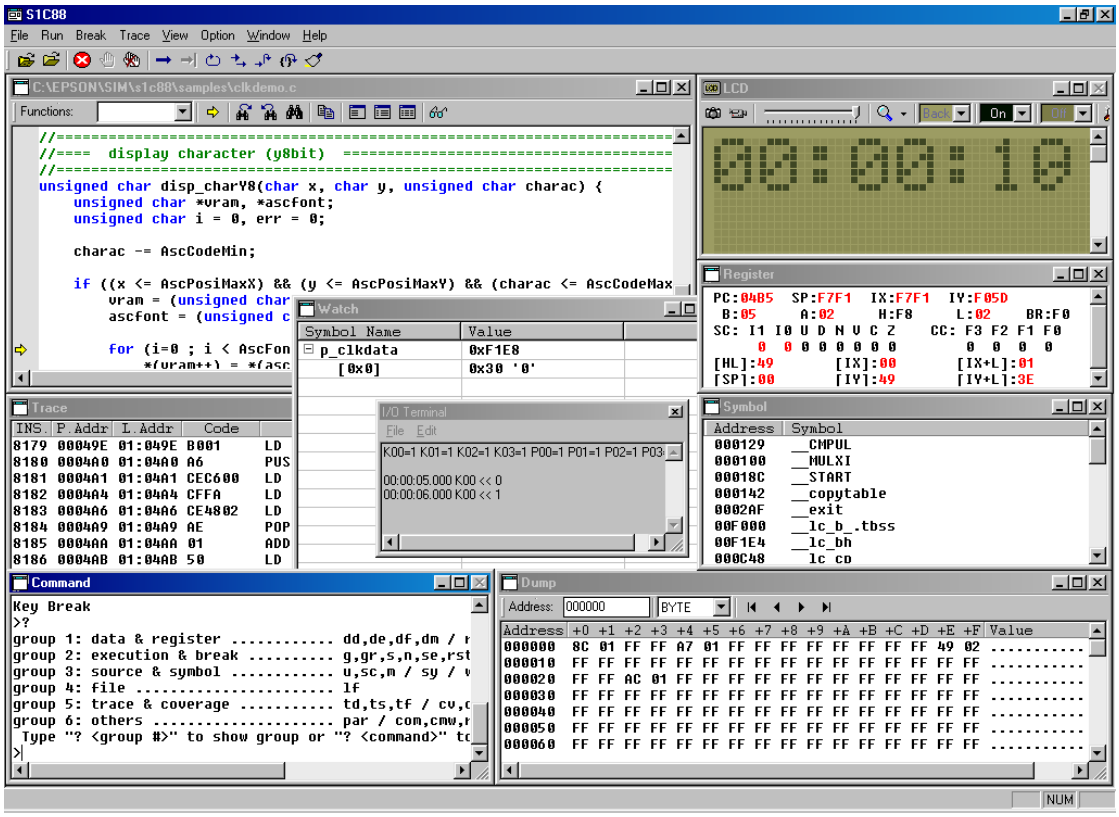
ワーニングメッセージ	メッセージ内容(エラーが発生するコマンド)
Break address already exists	設定済みのブレークアドレスを再設定(bp)
Identical break address input	ブレークアドレスの二重設定
round down to multiple of 4	監視データアドレスが不正(dw)

4　S1C88 Familyシミュレータ

4.1　概要

シミュレータsim88はS1C88 Familyの開発ツールです。このシミュレータにより、S1C88 Family統合ツール(Cコンパイラ、アセンブラ等)で作成したプログラムをICE等の専用のハードウェアツールを使用せずにパーソナルコンピュータのみでデバッグできます。このシミュレータは一般のデバッグ機能に加え、入出力ポートを使用したプッシュボタンやキーマトリクス、シリアルおよび汎用ポートの入出力、A/D変換、さらにはLCDの表示もシミュレートできます。このためのビットマップやLCDパネルデータも本パッケージに含まれるユーティリティで作成できます。
シミュレータの特長を以下に示します。

- 外部のデバッグ用ハードウェアを使用せずに、PC上でLCD表示を含めたシミュレーションが可能
- マルチウィンドウにより各種のデータを一度に参照可能
- 使用頻度の高いコマンドはツールバーおよびメニューからマウス操作で実行可能
- Cソース表示、逆アセンブルによるプログラムコード表示およびシンボル表示機能
- プログラムの連続実行と3種類のステップ実行が可能
- 3種類のブレーク機能
- トレースおよびカバレッジ機能
- コマンドファイルによるコマンドの自動実行機能



注: 動作OS環境についてはS5U1C88000QのReadme_J.txtを参照してください。

4.2　ソフトウェア開発フロー

図4.2.1にS1C88 Familyのソフトウェア開発フローを示します。太字の部分が本パッケージに含まれるツールと関連ファイルです。

注: 図に示すとおり、S1C88 Familyのソフトウェア開発には本パッケージ以外に、S1C88 Family統合ツールパッケージが必要です。

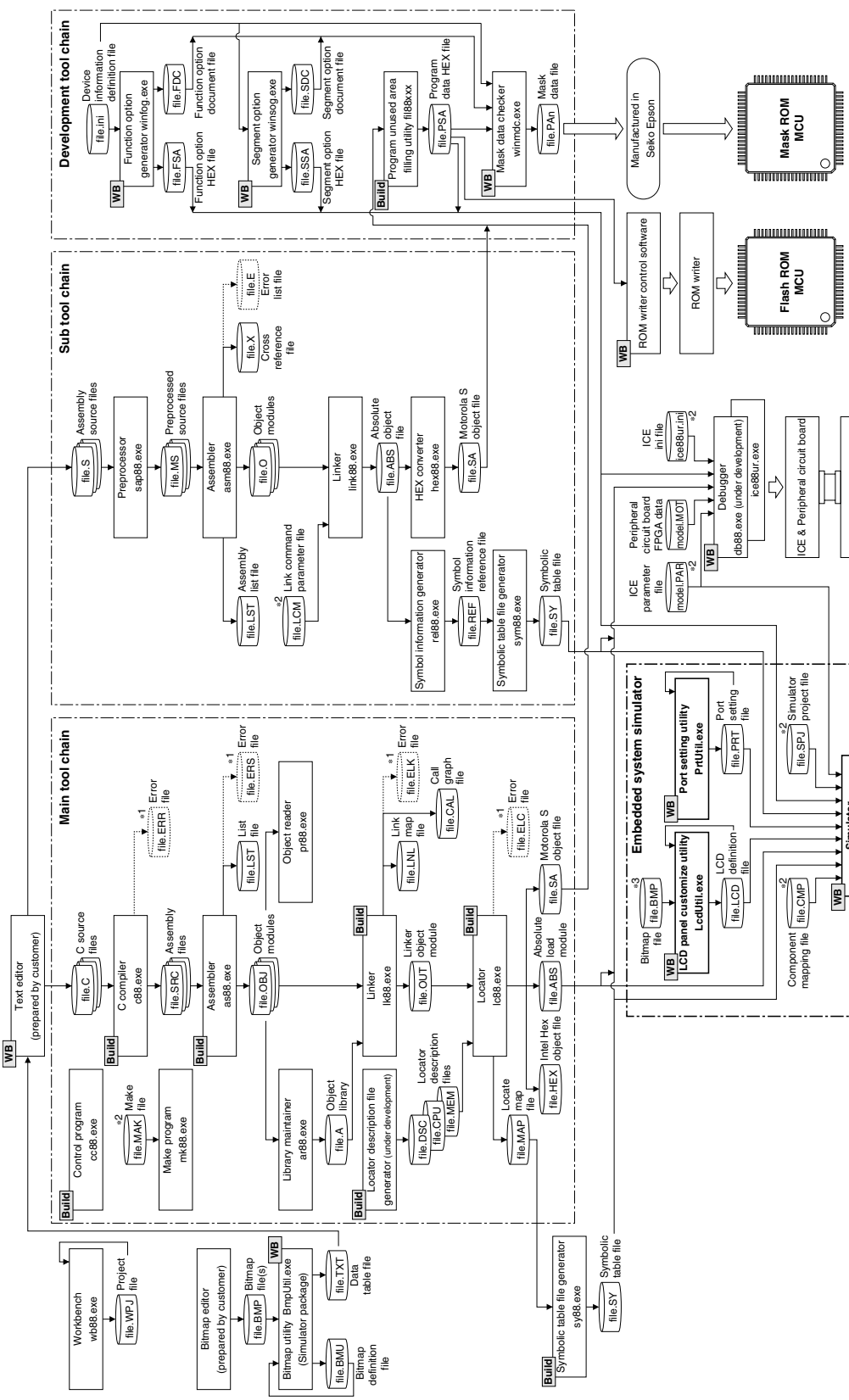


図4.2.1 ソフトウェア開発フロー

WB ワークベンチwb88から起動可能。 Build ビルド時にワークベンチwb88が自動実行。 *1: エラーファイルが生成された場合、wb88はその内容をメッセージビュー内にタグジャンプ機能付きで表示します。 *2: テキストエディタで作成。 *3: ビットマップエディタで作成。

4.3 シミュレーション機能の概要

ここでは、シミュレータsim88を使用したPC上でのターゲットアプリケーションのシミュレーションについて、概要を説明します。

● 対応機種

本シミュレータの対応機種につきましてはReadme_J.txtの、"サポート機種および周辺機器"を参照してください。

● ターゲットとしているアプリケーション

本シミュレータは、時計、電卓、電子手帳、携帯ゲーム等、キー入力およびLCD表示機能を持つアプリケーションのシミュレーションに最適です。
OSC1クロック動作はリアルタイムに実行可能です。OSC3クロック動作については、最大1MHz～2MHz程度までをリアルタイムに実行できます(Pentium 400MHz、64MB RAM相当のPCで1MHz程度のリアルタイム実行が可能)。
ただし、インストラクションレベルのシミュレーション精度のため、制御系などの高精度なタイミングのシミュレーションは行えません。
外部デバイスは、ROM、RAM、以下のLCDドライバ、モノクロLCDパネル、バックライト、キー、キーマトリクスのみをサポートします。
また、テキストI/Oターミナルプラグインにより、シリアル入出力、A/D変換器、汎用ポートの入出力もシミュレート可能です。

対応LCDドライバ: S1D15210, S1D15600, S1D15601, S1D15602, S1D15605, S1D15606, S1D15607, S1D15608

注: PC上でのシミュレーションのため、上記以外の制限事項もあります。"4.15 制限事項"およびReadme_J.txtを参照してください。

● 回路図情報のシミュレータへの入力

外部デバイスの動作をシミュレートするために必要な情報は、ファイルを作成してシミュレータに設定することができます。

・ROM、RAMのマッピング
パラメータファイル(file.par)で各デバイスを割り付けるアドレスを設定します。
ファイルの作成方法については、S5U1C88000C Manualを参照してください。

・外部デバイスのマッピングと内部発振クロック周波数
LCDドライバとバックライトを割り付けるアドレスとバックライトの制御ビットを記述したコンポーネントマップファイル(file.cmp)を作成してシミュレータにロードすることにより、これらの外部デバイスの制御をシミュレートすることができます。また、コンポーネントマップファイルにはOSC1/OSC3発振クロック周波数などのシミュレート条件も設定します。
ファイルの作成方法については、"4.12 コンポーネントマップファイル(.cmp)"を参照してください。

・キー、キーマトリクス仕様の入力
キー入力ポート、キーマトリクスを構成する入力/出力ポート、ターゲットとPCキーボードの対応を記述したポート設定ファイル(.prt)を作成してシミュレータにロードすることにより、PCキーボードを使用してキー入力をシミュレートすることができます。
ファイルの作成方法については、"7 ポート設定ユーティリティ"を参照してください。

・LCDデザインの入力
LCDパネルのレイアウトおよびSEG/COMポートの割り付けを記録したLCDパネル定義ファイル(file.lcd)を作成してシミュレータにロードすることにより、LCDパネル上の表示をシミュレートすることができます。
LCDパネル定義ファイルの作成方法については、"5 LCDパネルカスタマイズユーティリティ"を参照してください。

これらのファイルは通常、シミュレータの起動時に読み込まれます。

● ターゲットプログラムロード、実行方法

ターゲットプログラムはシミュレータのlfコマンドでロードし、g(またはgr、s、n、se)コマンドで実行します。

プログラム実行中のLCD表示や入出力のシミュレーションはシミュレータの[LCD]ウィンドウ、[I/O Terminal]ウィンドウを使用して行います。

・LCD表示: [LCD]ウィンドウ内に実際のLCDと同様に表示されます。
・キー入力: [LCD]ウィンドウをアクティブにしてPCキーボードの該当キーを入力します。
専用のウィンドウでキー入力状態の設定/確認も行えます。
・SVD評価: [LCD]ウィンドウ上のSVDスライドバーを操作してSVD機能をシミュレートできます。
・シリアルインタフェースと汎用ポートの入出力、A/D変換:
[I/O Terminal]ウィンドウをアクティブにして入力シーケンスを記述したIOTファイルを読み込み、シミュレーションを行います。

4.4　入出力ファイル

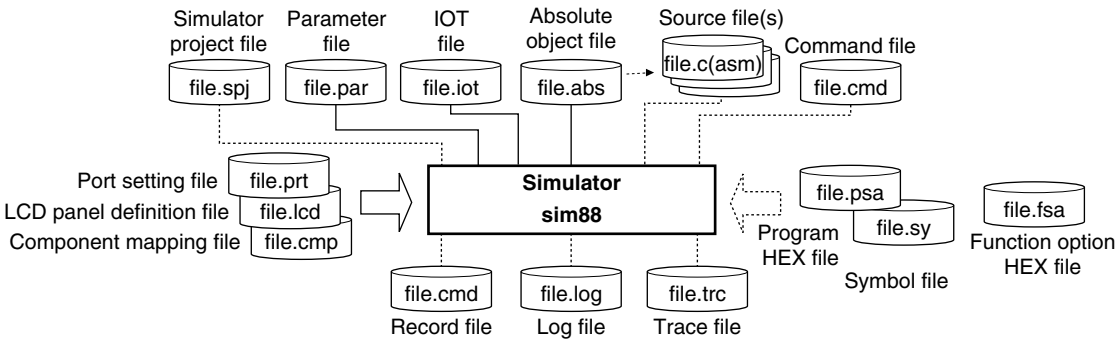


図4.4.1　入出力ファイル

● パラメータファイル(file_name.par)
各機種のメモリ情報などが記録されたテキストファイルで、シミュレータ内のメモリマップ設定に使用されます。内容については、"S5U1C88000C Manual"を参照してください。

● アブソリュートオブジェクトファイル(file_name.abs)
ロケータで生成したIEEE-695形式のオブジェクトファイルです。デバッグ情報を含んだIEEE-695形式のファイルを読み込むことで、Cソース表示およびシンボリックデバッグが行えます。

● ソースファイル(file_name.c, file_name.asm)
上記オブジェクトファイルのソースファイルで、ソース表示を行うときに読み込まれます。

● 内蔵ROMデータHEXファイル(file_name.psa)
内蔵ROM未使用領域FF詰めユーティリティ(fil88xxx)で生成した、モトローラS2フォーマットの内蔵ROMデータファイルです。内蔵ROMの未使用領域にはFFHのコードが埋め込まれています。また、S1C88xxxのシステム予約領域にはシステムコードが埋め込まれています。

● ファンクションオプションHEXファイル(file_name.fsa)
ファンクションオプションジェネレータで生成した、モトローラS2フォーマットのマスクオプション設定用ファイルです。

● シンボル情報ファイル(file_name.sy)
シンボルテーブルファイルジェネレータで生成したシンボル情報ファイルです。上記のROMデータHEXファイルと同じ名称で同じディレクトリに用意しておくことにより、ROMデータHEXファイルのロード時に自動的に読み込まれます。これによりソースで定義したシンボルの表示等が行えるようになります。

● シミュレータプロジェクトファイル(file_name.spj)
sim88の起動時に、パラメータファイル、LCDパネル定義ファイル、コンポーネントマップファイル、ポート設定ファイルを一括して指定するためのファイルです。エディタにより各ファイル名を入力して作成します。なお、sim88はこのファイルを生成しない場合でも、各ファイルをダイアログで選択することによって起動可能です。

● IOTファイル(file_name.iot)
I/Oテキストターミナルでシリアル/汎用ポート入出力、A/D変換をシミュレートするための入力データを記述したテキストファイルです。

● LCDパネル定義ファイル(file_name.lcd)
LCDパネルのレイアウトおよびCOM/SEGポート割り付け情報が記録されたファイルです。LCDパネルカスタマイズユーティリティ(LcdUtil)で生成します。

● コンポーネントマップファイル(file_name.cmp)
外付けLCDドライバやバックライトのマップアドレスなどを設定するテキストファイルです。

● ポート設定ファイル(file_name.prt)
プッシュキーやキーマトリクスとポートの対応を定義しておくテキストファイルです。ポート設定ユーティリティ(PrtUtil)で生成します。

● コマンドファイル(file_name.cmd)
連続して実行させるデバッグコマンドを記述したテキストファイルです。頻繁に使用する一連のコマンドを書き込んでおくことで、キーボードからのコマンド入力の手間を省くことができます。このファイルはcomコマンドにより読み込み、実行させます。

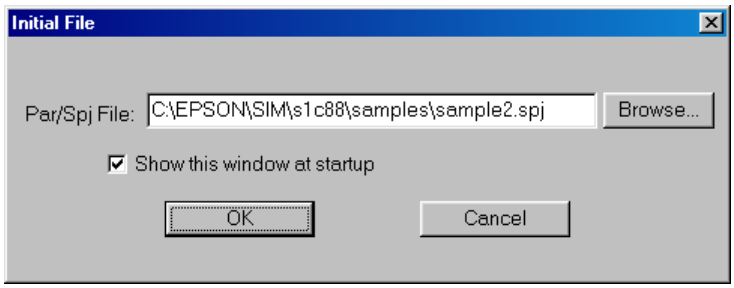
● ログファイル(file_name.log)
実行したコマンドと実行結果が出力されます。このファイル出力はlogコマンドによって制御できます。

● レコードファイル(file_name.cmd)
実行したコマンドがテキスト形式で出力されます。このファイル出力はrecコマンドによって制御できます。出力されたファイルはそのままコマンドファイルとして使用できます。

● トレースファイル(file_name.trc)
トレースデータの指定範囲が出力されます。このファイル出力はtfコマンドによって制御できます。

4.5　起動と終了

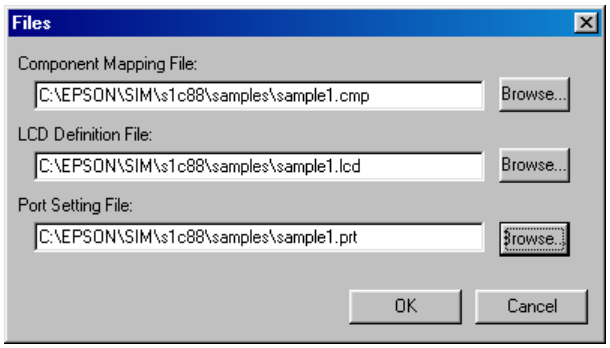
このアイコンをダブルクリックすると、シミュレータが起動します。
シミュレータを初めて起動すると次のダイアログボックスが表示されますので、パラメータファイルまたはシミュレータプロジェクトファイル(4.14項参照)の名称をテキストボックスに入力するか、[Browse...]ボタンで選択してください。



[Show this window at startup]チェックボックスをオフにすると、このダイアログボックスは次回の起動時から表示されなくなり、同じファイルが選択されます。その後、再選択する場合はparコマンド([File ¦ Load Parameter File...])で再表示させることができます。

チェックボックスをオンにした場合、次回の起動時はテキストボックスに同じファイル名が表示され、[OK]のクリックのみで済みます。

ここでパラメータファイルを選択すると次のダイアログボックスが表示されますので、コンポーネントマップファイル、LCDパネル定義ファイル、ポート設定ファイルをそれぞれのテキストボックスに入力するか、[Browse...]ボタンで選択してください。



パラメータファイルを選択した場合、上記の[Show this window at startup]チェックボックスをオフにしても、このダイアログボックスが次回の起動時にも表示されます。ただし、テキストボックスに同じファイル名が表示され、[OK]のクリックのみで済みます。
シミュレータプロジェクトファイルを選択した場合、このダイアログボックスは表示されません。

シミュレータをワークベンチwb88から起動する場合は、ワークベンチのダイアログボックスでこれらのファイルを指定します。したがって、シミュレータ起動時に、これらのダイアログボックスは表示されません。

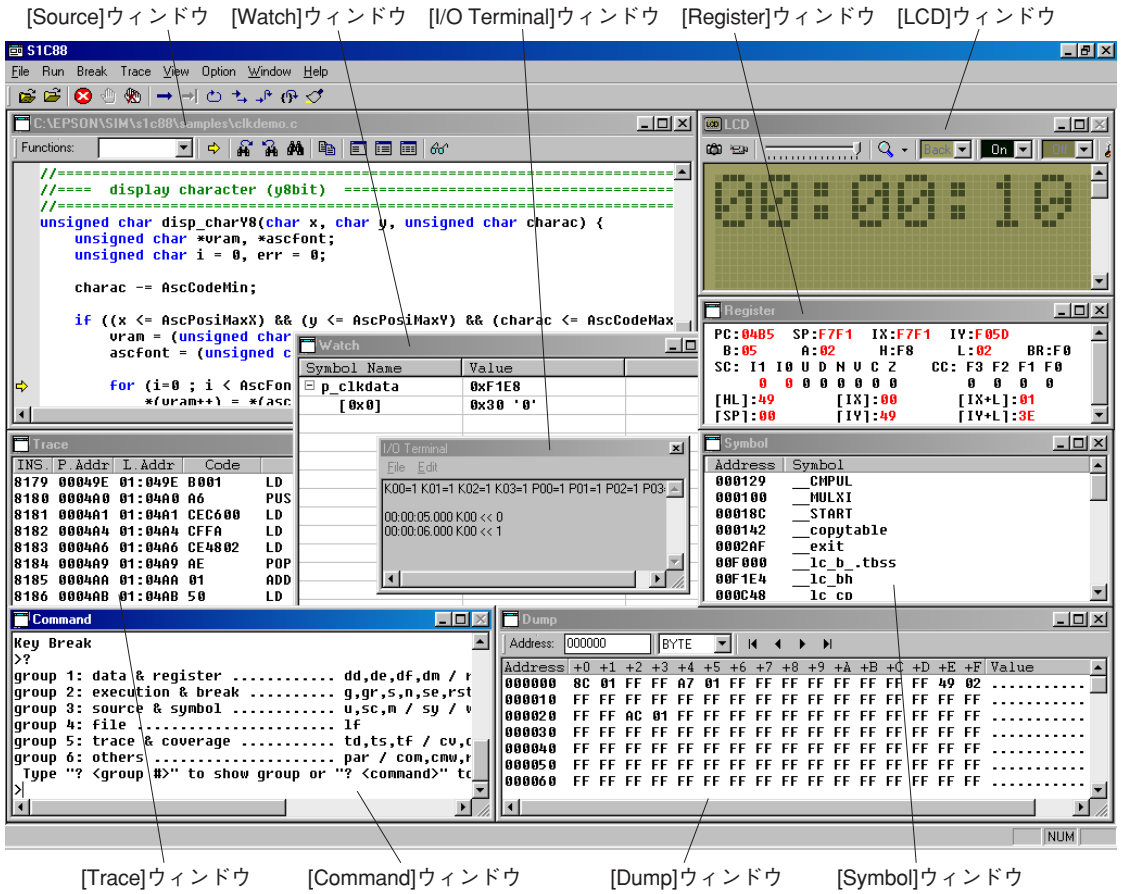
ミュレータを終了するには、[File]メニューから[Exit]を選択します。

4.6　ウィンドウ

ここでは、シミュレータで使用するウィンドウの種類を説明します。

4.6.1 ウィンドウの基本構成

シミュレータのウィンドウ構成は次のとおりです。



● 全ウィンドウの共通な操作

(1) ウィンドウのオープン/クローズとアクティブ化

[Command]と[LCD]以外のウィンドウはすべて閉じることと開くことができます。
ウィンドウを開くには、[View]メニューからそのウィンドウ名を選択してください。結果を特定のウィンドウに表示するデバッグコマンドを実行した場合も、対応するウィンドウが開きます。
ウィンドウを閉じるには、ウィンドウの[閉じる]ボタンをクリックしてください。
開いているウィンドウは[Window]メニューにリストされます。そこからウィンドウ名を選択することで、そのウィンドウがアクティブになります。ウィンドウ上をクリックすることでも同様です。また、[Ctrl]+[Tab]のキー操作によってもアクティブウィンドウの切り替えが行えます。

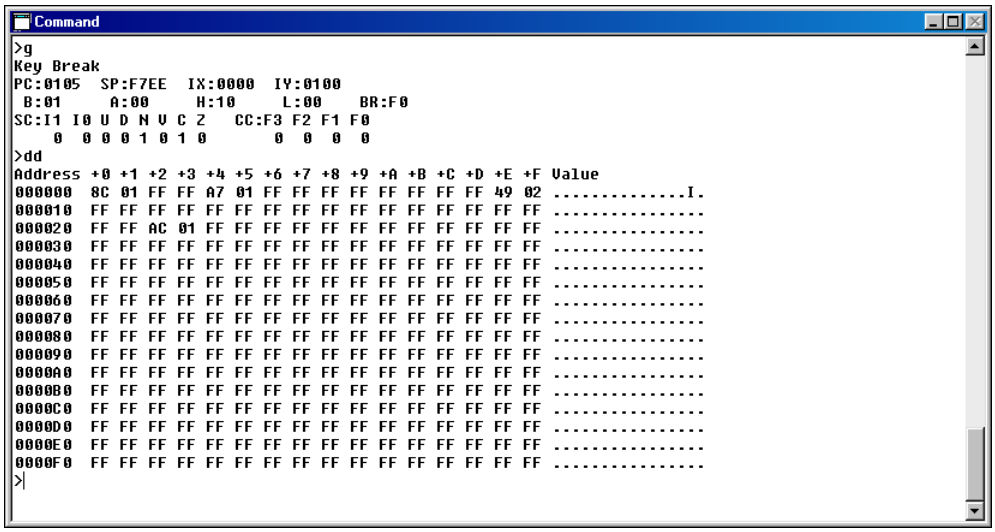
(2) サイズの変更と移動

それぞれのウィンドウサイズは、ウィンドウの境界をドラッグすることによって任意の大きさに変更できます。[最小化]ボタン、[最大化]ボタン等も一般のWindowsアプリケーションと同様です。各ウィンドウはタイトルバーをドラッグすることによって、表示位置を変更できます。ただし、サイズ変更、移動ともに、アプリケーションウィンドウの範囲内に限られます。

(3) その他

開いているウィンドウは、[Window]メニューの[Cascade]または[Tile]を選択することで整列させることができます。

4.6.2 [Command]ウィンドウ



[Command]ウィンドウは以下の目的に使用します。

(1) デバッグコマンドの入力

[Command]ウィンドウにプロンプト">"が表示され、キーボードからのコマンド入力を受け付けます。

(2) メニュー/ツールバーから選択したコマンドの表示

メニューやツールバーからデバッグコマンドを選択して実行させた場合は、そのコマンドラインが[Command]ウィンドウに表示されます。

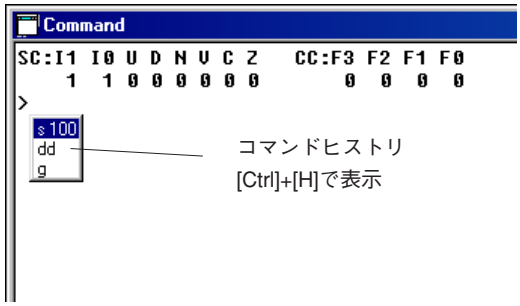
(3) コマンド実行結果の表示

コマンドの実行結果を表示します。ただし、コマンド実行結果の中には、[Source]ウィンドウ、[Dump]ウィンドウ、[Register]ウィンドウ、[Symbol]ウィンドウ、[Trace]ウィンドウに表示される内容もあります。これらの内容は対応するウィンドウが開いていれば、その中に表示されます。ウィンドウが閉じている場合は、[Command]ウィンドウに表示されます。

ログファイルへの書き込みを設定中は、その書き込み内容が表示されます。(logコマンド参照)

(4) コマンドヒストリの表示

Sim88はコマンドヒストリとして、起動後から現在までに実行しているコマンドの中で最新の32個を記憶します(まったく同じコマンドが複数回実行されていた場合はそれらを1つとして数えます)。記憶しているコマンドは、[Command]ウィンドウがアクティブになっている状態で[Ctrl]+[H]キーの入力によって呼び出すことができます。



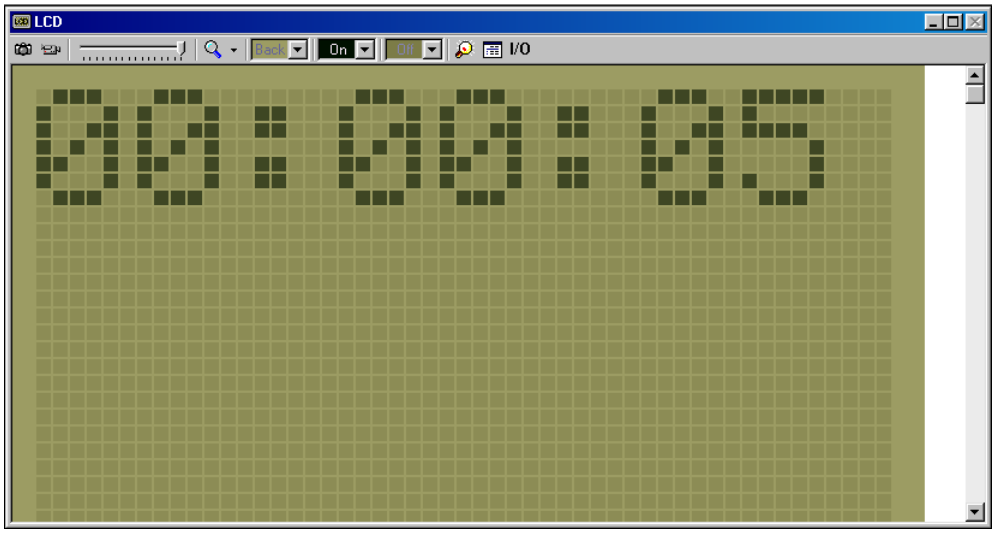
・単に[Ctrl]+[H]を入力すると、コマンドヒストリがポップアップリストの形式で表示されます。再実行するコマンドをマウスでダブルクリックするか、上下矢印キーで選択して[Enter]を押すと、そのコマンドがプロンプト位置にペーストされ、さらに[Enter]キーを押すことで実行できます。コマンドヒストリにコマンドが1個のみ登録されている場合、ポップアップリストは表示されずに、プロンプト位置に直接ペーストされます。

・文字入力に続けて[Ctrl]+[H]を入力した場合は、以下のいずれかの動作をします。
- コマンドヒストリにその文字(列)で始まるコマンドが複数登録されている場合、それらのコマンドがリストされます。その状態でさらに文字(列)を入力すると、表示されたコマンドの中で、その文字(列)を含む最近実行したコマンドが選択(ハイライト)状態となります。
- コマンドヒストリにその文字(列)で始まるコマンドが1個だけ登録されている場合、そのコマンドが直接プロンプト位置にペーストされます。
- コマンドヒストリにその文字(列)で始まるコマンドが1個もない場合は何も行いません。

たとえば、dd、sy、sの3つのコマンドがコマンドヒストリに登録されている場合、
- sを入力後、[Ctrl]+[H] を入力するとsとsyがリスト表示されます。この段階では、最近実行したsコマンドが上に表示され、ハイライト状態となります。
- 上記の操作に続けてyを入力すると、syがハイライト表示されます。
- dを入力後、[Ctrl]+[H] を入力するとプロンプト位置にddがペーストされます。

注: [Command]ウィンドウを閉じることはできません。

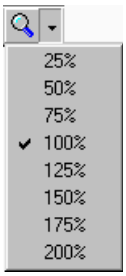
4.6.3 [LCD]ウィンドウ



[LCD]ウィンドウには、以下の機能があります。

(1) LCD表示のシミュレーション

LCDパネル定義ファイルに設定されたLCDパネルを表示します。アイコンやドットマトリクスの表示は、プログラムの実行に従って変化します。
また、以下のコントロールでパネルのサイズや色を設定できます。



[拡大縮小]ボタン・ドロップダウンリスト
ボタンをクリックするごとにパネルのサイズが25%ずつ拡大します。200%まで拡大し、その次は25%のサイズに縮小します。プルダウンリストでは、直接拡大・縮小の数値を選択します。

[Back]、[On]、[Off]ドロップダウンリスト
LCDパネル自体の色を設定します。
[Back] 背景色
[On] ドットがOnのときの色
[Off] ドットがOffのときの色

[Back Light]ボタン
次のダイアログボックスが表示され、バックライトの色を最大4色まで登録しておくことができます。色はRGBの各スライダで調整してください。左下のチェックボックスを選択すると、そのバックライトが設定されます。

(2) パネルイメージのキャプチャ

カメラボタン

このボタンをクリックすると、その時点のLCDパネルイメージをキャプチャし、ビットマップイメージとしてクリップボードに転送します。これをペイントソフト等にペーストし、印刷したりファイルに保存することができます。
このボタンはプログラム実行中は無効となるため、キャプチャしたいイメージのところで実行を中断してから使用してください。

ビデオボタン

このボタンをクリックすると、パネルイメージの録画機能(AVIファイルとして保存)が有効になります。その状態でプログラムの実行を開始すると、ファイル名を入力するダイアログボックス、圧縮形式を指定するダイアログボックスが順次表示されますので、それぞれ入力と選択を行ってください。圧縮形式は利用可能なものを選択してください。録画はプログラムの実行を中断するまで継続します。
作成されたファイルはWindows標準のメディアプレーヤーで再生することができます。

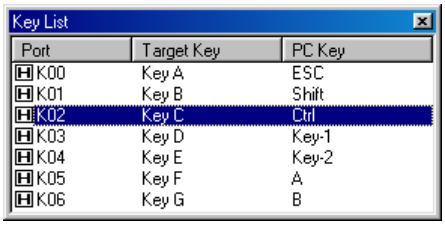
このボタンをクリックすると、解除することはできません。録画を中止するには、ファイル名を入力するダイアログボックスで[Cancel]ボタンをクリックしてください。プログラムは実行されますが、録画はキャンセルされます。

注: 録画中はプログラムの実行速度が低下します。

(3) キー入力のシミュレーション

実行中のプログラムがキー入力待ちの場合、[LCD]ウィンドウをアクティブにすることで、PCのキーボードを使用してキー入力をシミュレートすることもできます。
ターゲットのキー名称と使用するポート、PCのキーとの対応はポート設定ファイルで定義します。
この定義内容は、[Key List]ボタンで一覧表示できます。


[Key List]ボタン



[Key List]ウィンドウの表示内容は、左からポート名、ターゲット上のキー名称、PC上のキー名称です。
ここで、ターゲットとPCのキー対応を確認することができます。
左端の記号はポートの状態で、HはポートがHigh、LはLowを示します。キーを押したときの入力レベルは機種ごとのポートの仕様で異なります。

ブレーク時に、このボックスでキーの状態を設定することもできます。設定したキー入力状態は、その後のプログラム実行時も、ユーザがそのキーを操作するまで保持されます。これにより、ステップ実行時もキー入力状態を保持しておくことができます。
チェックされているキーが入力ポートに直接接続されているものであれば、その入力ポートの状態はLowに保持されますが、キーマトリクスの場合はそのキーが交差する出力ポートがLowになった場合にのみ入力ポートがLowになります。

(4) SVDのシミュレーション

このスライダによりSVDの検出レベルを16段階に変化させることができます。

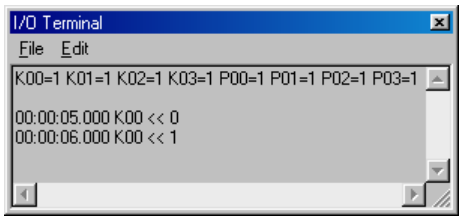
(5) シリアル/汎用ポートの入出力、A/D変換のシミュレーション

[I/O]ボタンで[I/O Terminal]ウィンドウを表示させ、IOTファイルを読み込むことにより、シリアルインタフェースの入出力、汎用ポートの入出力およびA/D変換をシミュレートすることができます。詳細は次項を参照してください。

[I/O]ボタン

注: [LCD]ウィンドウを閉じることはできません。

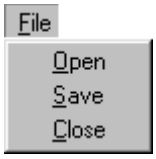
4.6.4 [I/O Terminal]ウィンドウ



[I/O Terminal]ウィンドウは、汎用入出力ポートおよびシリアルインタフェースの入出力、A/D変換をシミュレートして入出力状態を表示するI/Oテキストターミナル機能を提供します。

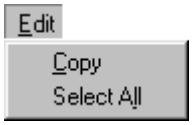
(1)メニュー

Fileメニュー



Open　IOTファイルを読み込みます。
Save　ウィンドウに表示されたログをテキストファイルに保存します。
Close　IOTファイルを閉じます(I/Oシミュレーションを中止します)。

Editメニュー



Copy　ウィンドウの選択範囲をクリップボードへコピーします。
Select All　ウィンドウの全てのログを選択します。
※ウィンドウ内の右クリックでも、Copy/Select Allメニューを表示できます。

(2)IOTファイル

IOTファイルは、以下の情報を記述したテキストファイルで、汎用のエディタにより作成します。
a. 入出力状態を監視する汎用ポート(Rxx、Pxx)の指定
b. 汎用ポート(Kxx、Pxx)への入力タイミングと入力レベルの指定
c. A/D変換器の電源電圧および基準電圧の指定
d. A/D入力電圧の指定
e. シリアルインタフェースの入力データの指定
例:

```
[General I/O]
watch P00, P01, P02, P10, P11, P20  ← a
5.0 K00=1                           ← b   先頭の数値はリセットからの秒数
5.5 K00=0                                 1 / H = High, 0 / L = Low
8.0 K00=H
8.5 K00=L
[A/D Converter]
AVdd=3.0                            ← c   数値は電圧値(V)
AVss=0.2
AVref=2.9
[A/D CH2]
2.40,2.39,2.38,2.37,2.36            ← d   数値は電圧値(V)
2.00, 2.10, 2.20, 2.30
0.5
0.65
0.80
[Serial CH1]
"aBcDeFg"                           ← e   文字列または16進数で入力データを指定
0x25,0x20,0x41
```

このファイルを読み込んで、各入力端子の状態をシミュレートします。ファイルの詳細については"4.13 IOTファイル(.iot)"を参照してください。

(3) 入出力シミュレーションの開始と終了

入出力シミュレーションは次の操作で開始します。

1. [I/O Terminal]ウィンドウを表示します。
2. [File]メニューの[Open]によってIOTファイルを読み込みます。
3. リセットからターゲットプログラムを実行します。

ターゲットプログラム実行中でもIOTファイルの読み込みは可能ですが、入出力シミュレーションを行うにはリセットすることが必要です。
入出力シミュレーションを終了するには、[File]メニューから[Close]を選択します。

(4) シミュレーションとログの表示

指定入出力ポートの状態が変化(1→0または0→1)した場合、シリアルインタフェースがデータの入出力を行った場合、A/D変換器がアナログ信号を入力した場合、その情報をウィンドウに表示します。
例:

```
P00=1 R01=1                          ┐ 初期データ
AVdd=3.000 AVss=0.200 AVref=2.900    ┘

00:00:01.029 S2 << 31h(1)            ┐
00:00:02.033 S2 << 32h(2)            │
00:00:03.036 S1 >> 33h(3)            │
00:00:04.040 S1 >> 34h(4)            │
00:00:05.000 P30 << 0 P31 << 0       │ ログデータ
00:00:06.016 AD0 << 2.400            │
00:00:07.020 AD0 << 2.390            │
00:00:08.023 AD0 << 2.380            │
00:00:09.027 AD0 << 2.370            │
00:00:05.000 R01 >> 1                ┘
```

リセット実行直後に、初期データが表示されます。表示される内容は、IOTファイルの[General I/O]セクションにある"Watch"文で指定した汎用入出力ポートの初期値、および[A/D Converter]セクションで指定した"AVdd"、"AVss"、"AVref"の値です。これらを指定していない場合は表示されません。
以下、シミュレーション実行中に入出力したデータが表示されます。各行の先頭に表示される数値はリセット実行からの経過時間(時分秒)です。"<<"は入力、">>"は出力を意味しています。
各ポートのデータと処理内容は以下のとおりです。

汎用入出力ポート:

Watch文で指定したポートが入力または出力が変化すると、ポート名(Kxx、Pxx、Rxx)と変化後の値(0または1)を表示します。入力の変化はIOTファイルに記述してあるタイミングで発生します。このとき、割り込みが許可されている場合は設定されている割り込み条件に従って入力割り込みが発生します。

A/D 変換器:

ターゲットプログラムがA/D変換を行うごとにIOTファイルに記述されているデータを上(左)から順に読み込んで、A/D変換のシミュレーションを行います。このとき、ファイルから読み込んだチャンネル番号(ADx)と入力電圧値(V)がウィンドウに表示されます。割り込みが許可されている場合、入力を開始してから一定時間(サンプリング時間 + A/D変換時間)経過後に割り込みが発生します。データを指定しなかったチャンネルを変換した場合、入力はレベル0の電圧(AVssと等価)として読み出されます。また、指定したデータ数よりも多くA/D変換を行った場合は、最後に指定したデータが繰り返し読み出されます。AVssよりも小さな値を指定した場合、レベル0(AVssと等価)として読み出され、AVrefよりも大きな値を指定した場合は最大レベル(AVrefと等価)として読み出されます。

シリアルインタフェース:

ターゲットプログラムがシリアルインタフェースの入出力を行うと、1バイトごとにそのチャンネル番号(S1、S2)と入力値/出力値(16進表記・ASCIIキャラクタ表示)を表示します。入力時は、IOTファイルに記述されている入力データが上(左)から順次読み出されます。割り込みが許可されている場合はシリアルインタフェースの割り込みは、入力サンプリング終了のタイミングで割り込みが発生します。

(5)ログの保存

[I/O Terminal]ウィンドウに表示されたログは、[File]メニューの[Save]でテキストファイルとして保存することができます。

また、[I/O Terminal]ウィンドウ内をドラッグして選択した範囲を、[Edit]メニューの[Copy]でクリップボードにコピーできますので、エディタなどで他の書類にペースト可能です。このとき、ログをすべてコピーしたい場合は、[Edit]メニューの[Select All]で全体を選択してからコピーします。

注意事項

・すでにS1C63/S1C88シミュレータをインストール済みの環境では、シリアルインタフェースに対応したコンポーネントをコンポーネントマップファイル(.cmp)に追加する必要があります。シリアルインタフェース未サポート機種を使用する場合、コンポーネントマップファイルにNullDev.bmcを追加してください。
また、A/D変換器を使用する場合も、同様にコンポーネントマップファイルを修正してください。

例)88348.cmp

```
[Internal]
CPU=CPU.bmc
LCD=LcdDrv88.bmc
K/P/R port=KPRport.bmc
SVD=SVD88.bmc
Sound=Sound88.bmc
Serial=Serial88.bmc ← 追加
Adc=NullDev.bmc     ← 追加
```

例)88349.cmp

```
[Internal]
CPU=CPU.bmc
LCD=LcdDrv88.bmc
K/P/R port=KPRport.bmc
SVD=SVD88.bmc
Sound=Sound88.bmc
Serial=Serial88.bmc ← 追加
Adc=Adc88.bmc       ← 追加
```

・入出力シミュレーションを行う時刻は、シミュレータのインストラクションサイクルから計算されますので、実機(実時間)と若干異なります。

4.6.5 [Source]ウィンドウ

[Source]ウィンドウはプログラムコードを表示します。以下に示す3種類の表示形式に対応しています。

1. 逆アセンブル表示モード
   ロードしたオブジェクトを逆アセンブルしてアドレス、コード、ニーモニックを表示します。[Source]ウィンドウをこの表示モードで開くには[View]メニューから[Source¦Disassemble]を選択します。他のモードで表示中に、逆アセンブル表示モードにするには、上記メニューを選択するか[Source]ウィンドウ上の[Disassemble]ボタンをクリック、あるいはuコマンドを実行します。[Source]ウィンドウがこの表示モードになると、タイトルバーには"Disassemble"と表示されます。この表示モードは読み込んだオブジェクトファイルの種類にかかわらず選択可能です。

    [Disassemble]ボタン

2. ソース表示モード
   現在のプログラムカウンタアドレスを含むオブジェクトに対応するソースを表示します。ただし、このモードは、ソース表示用のデバッグ情報を含むIEEE-695形式のアブソリュートオブジェクトファイル(.abs)を読み込んだ場合にのみ選択可能です。[Source]ウィンドウをこの表示モードで開くには[View]メニューから[Source¦Source]を選択します。他のモードで表示中に、ソース表示モードにするには、上記メニューを選択するか[Source]ウィンドウ上の[Source]ボタンをクリック、あるいはscコマンドを実行します。また、[Source]ウィンドウが表示されている状態で、Cソースデバッグ情報を含むアブソリュートオブジェクトファイル(.abs)を読み込むと、[Source]ウィンドウは自動的にこのモードになります。この表示モードになると、タイトルバーにはソースファイル名が表示されます。

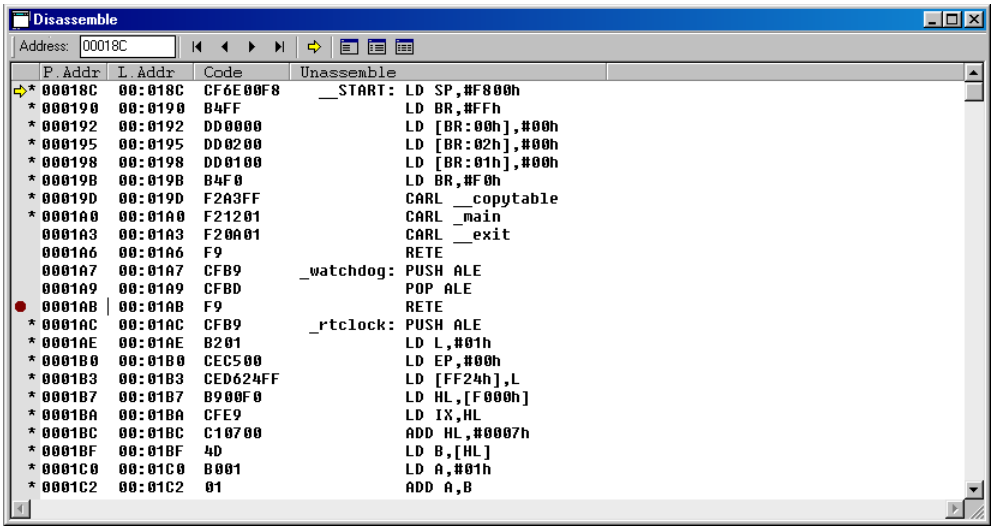
    [Source]ボタン

3. ミックス表示モード
   ソースと逆アセンブル内容(アドレス、コード、ニーモニック)を上下に対応させて表示します。ただし、このモードは、ソース表示用のデバッグ情報を含むIEEE-695形式のアブソリュートオブジェクトファイル(.abs)を読み込んだ場合にのみ選択可能です。[Source]ウィンドウをこの表示モードで開くには[View]メニューから[Source¦Mix]を選択します。他のモードで表示中に、ソース表示モードにするには、上記メニューを選択するか[Source]ウィンドウ上の[Mix]ボタンをクリック、あるいはmコマンドを実行します。[Source]ウィンドウがこの表示モードになると、タイトルバーには"Mix"と表示されます。

    [Mix]ボタン

※ソースの表示
ソースは、ソース表示用のデバッグ情報を含むIEEE-695形式のアブソリュートオブジェクトファイルを読み込んだ場合にのみ表示可能です。
また、オブジェクトファイルのデバッグ情報(ソースファイルの相対パス情報)からソースファイルを探して読み込みますので、ソースファイルを削除あるいは移動した(オブジェクトファイルからの相対位置が変わった)場合、ソースは表示されません。この場合、ソース表示モードではウィンドウが空白となり、ミックス表示モードでは逆アセンブル内容のみが表示されます。

逆アセンブル表示モード



逆アセンブル表示モード時の[Source]ウィンドウの機能を以下に示します。

(1) プログラムコードの表示

物理/論理アドレス、オブジェクトコード、逆アセンブル内容を表示します。
プログラムの表示箇所は、スクロール以外に以下の方法で変更できます。

・ [Address]テキストボックスにアドレスを入力またはuコマンドでアドレスを指定します。
そのアドレスを先頭に表示します。

・

プログラムの先頭、または最後を表示します。
現在のウィンドウサイズで1ページ前、または後を表示します。
現在のPCアドレスから表示します。

注: S1C88 Familyプロセッサのニーモニックは可変長のため、上方向にスクロールした場合、実際のコードと異なる逆アセンブルを行うことがあります。

※ 表示の更新

プログラムをロードして実行(g、gr、s、n、se、rstコマンド)するか、プログラムメモリの内容を変更(de、df、dmコマンド)すると表示内容が更新されます。この場合、現在のPCが示すアドレスがウィンドウ内に表示されるように表示が更新されます。

(2) カレントPCの表示

現在のPC(プログラムカウンタ)が示すアドレスの行は、先頭に黄色の矢印を表示します。

(3) PCブレークポイントの表示

ブレークポイントに設定されたアドレスの行は、先頭に赤の●マークを表示します。

(4) カバレッジ情報の表示

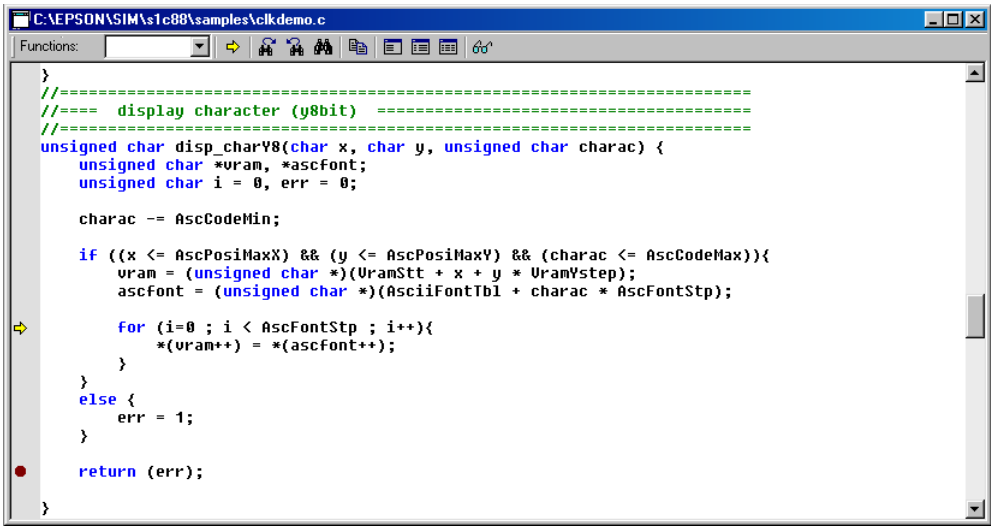
mdコマンドでカバレッジ機能をONに設定した場合、それ以降に実行したアドレスの先頭に*が表示されます。

(5) カーソル位置でブレーク設定

ブレークポイントを設定したいアドレスの行にカーソルを置きます。そこで、[Break]ボタンをクリックすると、そのアドレスがPCブレークポイントに設定されます(行内をダブルクリックすることによっても設定可能)。PCブレークポイントに設定されたアドレスの行で同じ操作をすると、そのブレークポイントは解除されます。ブレークポイントは複数のアドレスに設定可能です。

[Go to Cursor]ボタンをクリックすると、プログラムが現在のPCから実行を開始し、カーソルのある行の実行後にブレークします。

ソース表示モード



ソース表示モード時の[Source]ウィンドウの機能を以下に示します。

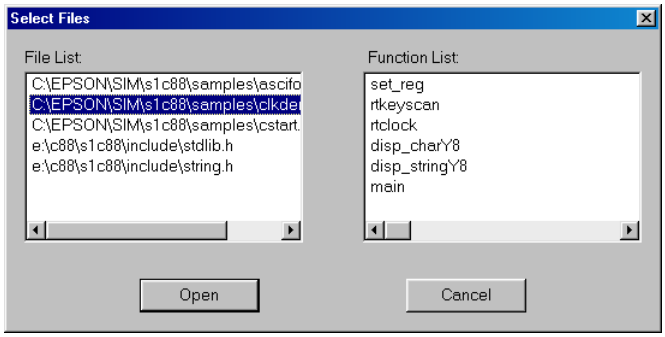
(1) プログラムコードの表示

ソースを表示します。自動的に表示されるのは、現在のPC(プログラムカウンタ)が示すアドレスを含むソースです。
コメントは緑、予約語は青、その他は黒で表示されます。
タブ幅は4文字に固定です。
プログラムの表示箇所は、スクロール以外に以下の方法で変更できます。

・ [Functions]プルダウンリストから関数名を選択します。
その関数の先頭から表示します。

・ [Current PC]ボタンをクリックします。現在のPCアドレスから表示します。

・ 他のソースファイルを表示させるには、[Source Files]ボタンをクリックして次のダイアログボックスを表示させ、リストされているソースの中から表示させるものを選択します。



※ 表示の更新
プログラムをロードして実行(g、gr、s、n、se、rstコマンド)後、実行を中断すると表示内容が更新されます。この場合、現在のPCアドレスを含むソースがウィンドウ内に表示されます。このとき、対応するソースが見つからなかった場合は上図の[Select Files]ダイアログボックスが表示され、表示するソースの選択が必要になります。

(2) カレントPCの表示

現在のPC(プログラムカウンタ)が示すアドレスを含むソース行は、先頭に黄色の矢印を表示します。

(3) PCブレークポイントの表示

ブレークポイントに設定されたアドレスを含むソース行は、先頭に赤の●マークを表示します。

(4) カーソル位置でブレーク設定

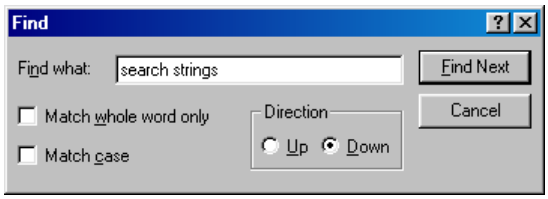
ブレークポイントを設定したいソースの行にカーソルを置きます。そこで、[Break]ボタンをクリックすると、そのソース行(ソースに対応する有効なオブジェクトコードの先頭のアドレス)がブレークポイントに設定されます(行内をダブルクリックすることによっても設定可能)。PCブレークポイントに設定されたソース行で同じ操作をすると、そのブレークポイントは解除されます。ブレークポイントはソース行単位で複数設定可能です。ただし、実コードを持たないソース行にはブレークポイントを設定することはできません。また、通常はコードが生成されるCステートメントでも、Cコンパイラの最適化によりコードが生成されないこともあります。ブレークポイントに設定できない場合はミックス表示モードに切り換えて確認してください。

[Go to Cursor]ボタンをクリックすると、プログラムが現在のPCから実行を開始し、カーソルのある行でブレークします。この場合も、カーソルを実コードを持つソース行に置く必要があります。実コードがない場合、[Go to Cursor]の操作は無効です。

(5) 文字列の検索

ソース表示モードでは、[Source]ウィンドウに以下の検索用のボタンが表示され、文字列の検索を行うことができます。

[Find]ボタンをクリックすると、検索文字列を指定するダイアログボックスが表示されます。



[Find what]エディットボックスに検索文字列を入力して[Find Next]ボタンをクリックすると、現在のカーソル位置から[Source]ウィンドウの下方向(プログラムの後方)に検索を行います。指定文字列が[Source]ウィンドウ内で見つかると、その文字列を選択状態にします。

そこでさらに[Find Next]ボタンをクリックすると、その位置から次の検索を開始します。

ウィンドウの上方向(プログラムの前方)に検索したい場合は、[Direction]の[Up]ボタンを選択してください。指定文字列に完全に一致するもののみを検索する場合は[Match whole word only]チェックボックスを、大文字と小文字を区別して検索する場合は[Match case]チェックボックスを、[Find Next]ボタンをクリックする前に選択してください。

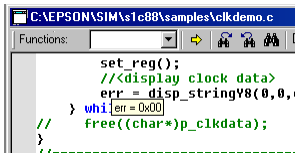
[Source]ウィンドウ内で文字列をドラッグして選択し、[Source]ウィンドウ上の[Find Next]ボタンをクリックすると、その選択位置から[Source]ウィンドウの下方向(プログラムの後方)にその文字列の検索を行います。文字列が見つかると、新たに見つかった方を選択状態にします。そこでさらに[Find Next]ボタンをクリックすると、その位置から次の検索を開始します。この検索においては、大文字と小文字は区別されません。また完全に一致しない文字列も検索の対象です。

[Find Previous]ボタンは、検索方向がウィンドウの上側(プログラムの前方)に変わる以外、上記の[Find Next]ボタンと同じ機能です。

(6) [Watch]ウィンドウへのシンボルの登録

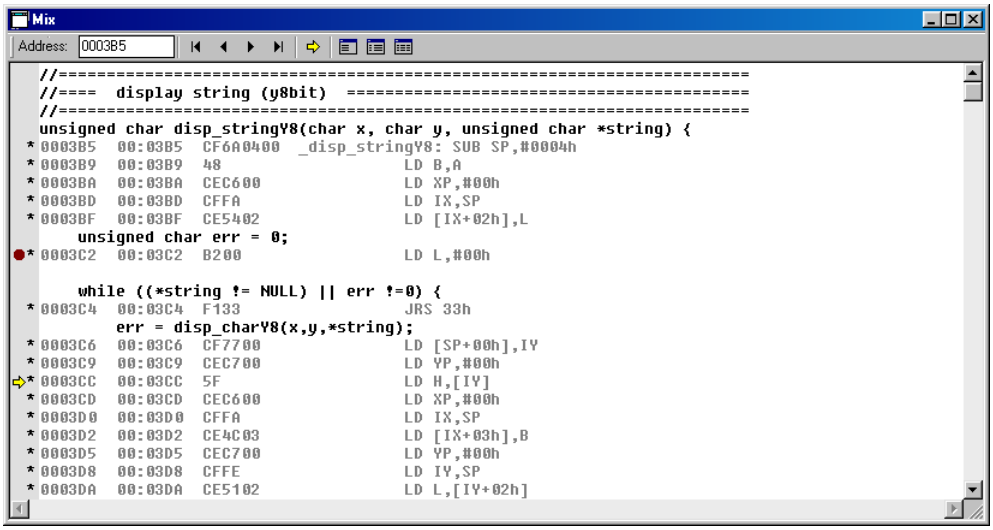
ウィンドウ内のシンボル名をドラッグして選択(反転表示)し[Watch]ボタンをクリックすると、そのシンボルが[Watch]ウィンドウのシンボルリストに登録されます。その後、[Watch]ウィンドウでそのシンボルの値を確認することができます。

(7) 変数の値を表示



表示されているソース中の変数名の上にマウスカーソルを置くと(クリックは不要)、変数の値(ポインタ変数の場合はアドレス)が表示されます。(signed/unsigned)int/long/short は10進数で、アドレス、構造体、共用体は16 進数で表示されます。構造体のメンバの値を表示するには、変数名をマウスで選択する必要があります。配列の要素の場合もマウスで選択してください。スコープを外れた変数は表示されません。

ミックス表示モード

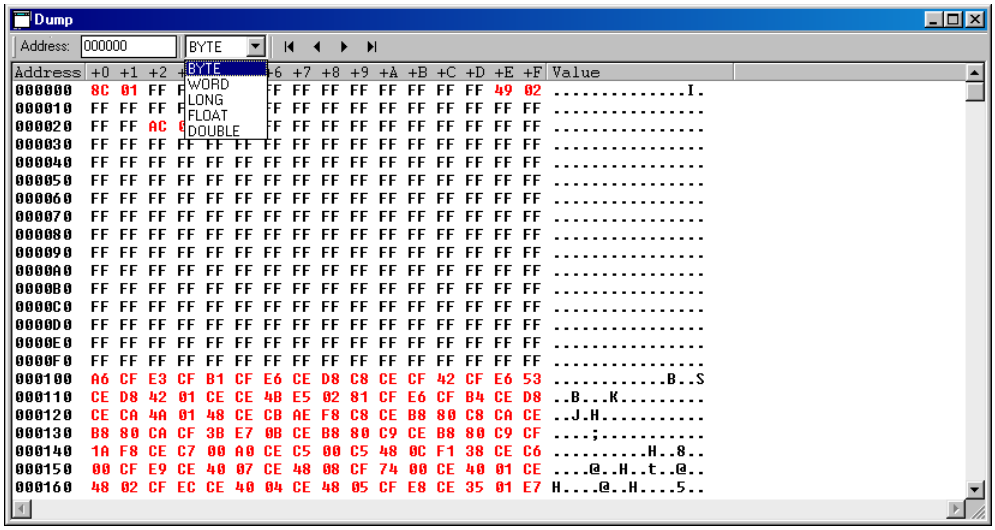


ミックス表示モードの機能は逆アセンブル表示モードと同様です。違いは各ソース行と対応するオブジェクトコードの逆アセンブル内容(物理/論理アドレス、オブジェクトコード、ニーモニック)が上下に並んで表示されることのみです。ただし、ミックス表示モードはデバッグ情報を含むIEEE-695形式のアブソリュートオブジェクトファイル(.abs)を読み込んだ場合のみ選択可能です。

表示されたソース行に対しては、ブレーク設定等の操作は一切行えません。各種表示、操作は逆アセンブル表示内容に対してのみ有効です。ミックス表示モードが対応している機能については、逆アセンブル表示モードの説明を参照してください。

ソース行は黒、逆アセンブル内容はグレーで表示されます。

4.6.6 [Dump]ウィンドウ



(1) メモリ内容の表示

データメモリのダンプ結果を16進数で表示します。
デフォルトではバイト単位に表示されますが、プルダウンボックスでその他のサイズにも変更可能です。
メモリの表示箇所は、スクロール以外に以下の方法で変更できます。

- [Address]テキストボックスにアドレスを入力またはddコマンドでアドレスを指定
  そのアドレスを先頭に表示します。
- 
  メモリ領域の先頭、または最後を表示します。
  現在のウィンドウサイズで1ページ前、または後を表示します。

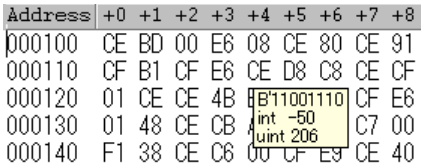
※ 表示の更新
メモリ内容をコマンド(de、df、dmコマンド)で変更すると、[Dump]ウィンドウの表示内容が更新されます。また、プログラムを実行(g、gr、s、n、se、rstコマンド)した場合も更新されます。
これ以外で、最新の内容を表示させるには、ddコマンドを実行するか、垂直スクロールバーをクリックしてください。
プログラムの実行を中断すると、実行前から変更された値は赤で表示されます。

(2) データメモリ内容の直接変更

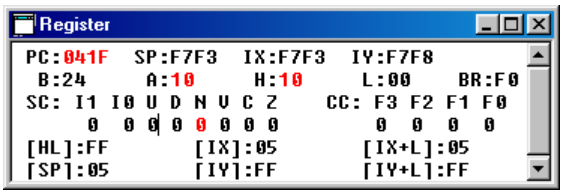
[Dump]ウィンドウ上で、データメモリを直接変更することができます。変更するデータの直前にカーソルを置くか、データをダブルクリック後、16進の数値(0～9、a～f)を入力してください。そのアドレスのデータが変更されます。カーソルは次のアドレスのデータに移動し、連続的なデータ変更を可能にしています。

(3) 10進データの表示

[BYTE]、[WORD]、[LONG]で表示中にマウスカーソルをデータの上に重ねると(クリックは不要)、10進データ(signed int/unsigned int)が表示されます。[BYTE]の場合はビットデータも表示されます。



4.6.7 [Register]ウィンドウ



(1) レジスタ内容の表示

S1C88 CPU内の全レジスタおよびコンディションフラグの内容、[HL]、[SP]、[IX]、[IY]、[IX+L]、[IY+L]で指定されるメモリの内容を表示します。

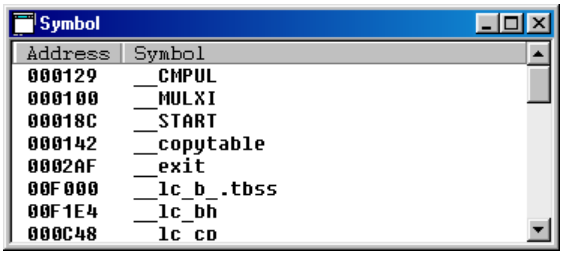
※ 表示の更新

レジスタダンプ時(rdコマンド)、レジスタデータ変更時(rsコマンド)、CPUリセット時(rstコマンド)、プログラムの実行(g、gr、s、n、seコマンド)終了後に更新されます。
プログラムの実行を中断すると、実行前から変更された値は赤で表示されます。

(2) レジスタ内容の直接変更

[Register]ウィンドウ上で、レジスタの内容を直接変更することができます。変更するデータを選択(ハイライト)後、16進の数値(0～9、a～f)を入力し[Enter]キーを押してください。そのレジスタの内容が変更されます。

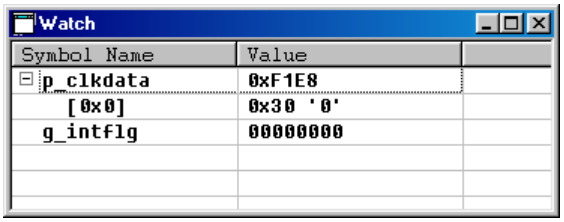
4.6.8 [Symbol]ウィンドウ



シンボルファイル(.sy)が読み込まれている場合に、シンボルの一覧を表示します。
デフォルトではアルファベット順に表示されます。"sy /a"コマンドにより、アドレス順に表示させることもできます。

* シンボルファイルはターゲットプログラム(モトローラS2形式)のロード時に自動的に読み込まれます。ただし、そのためには、ターゲットプログラムと同じ名称のシンボルファイルをターゲットプログラムと同じディレクトリに用意しておく必要があります。IEEE-695形式のプログラムファイルの場合、シンボルファイルは読み込まれません。

4.6.9 [Watch]ウィンドウ



wコマンドまたは[Source]ウィンドウの[Watch]ボタンで登録したシンボルの名称と現在の値を表示します。値はwコマンドで指定した形式で表示されます。シンボルが配列、構造体、共用体の場合、+アイコンが表示されます。それをクリックすることにより、メンバが階層表示されます。

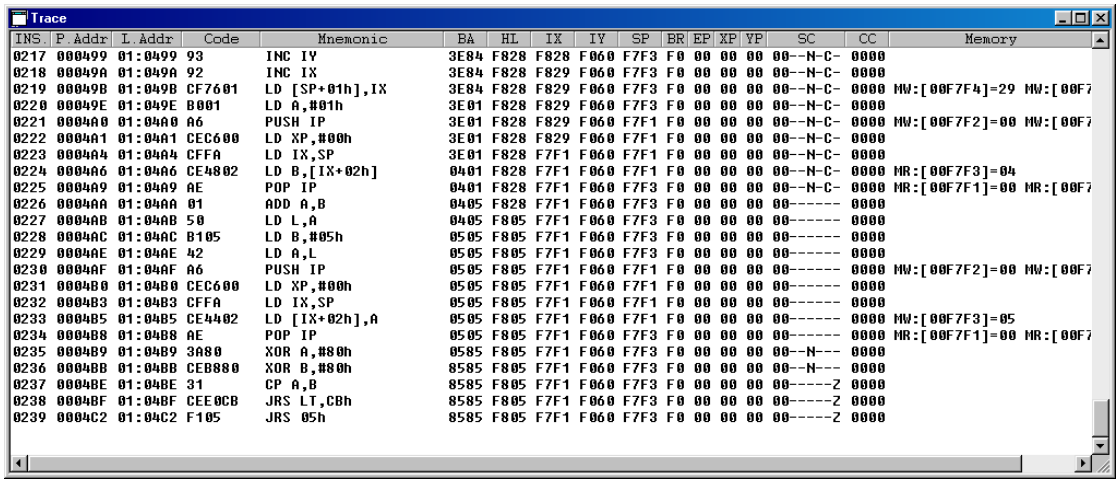
また、右クリックにより表示されるメニューで、登録したシンボルの削除や表示形式変換(16進数→10進数など)ができます。ただし、変換できるのはint、char、long、short などの型のみで、アドレスは16 進数表記固定です。
なお、このウィンドウを使用したシンボル表示は、指定シンボルの情報を含むIEEE-695形式のアブソリュートオブジェクトファイル(.abs)を読み込んだ場合のみ可能です。

注: コンパイル時に-O1オプションを指定すると、コードの最適化により不要なシンボルが削除され、シンボル情報が生成されない場合があります。そのようなシンボルは、[Watch]ウィンドウに登録することはできません。

※ 表示の更新

プログラムの実行(g、gr、s、n、seコマンド)終了後に更新されます。

4.6.10 [Trace]ウィンドウ



mdコマンドでトレース機能をONに設定すると、それ以降のプログラム実行時にトレース情報を採取します。トレース用のバッファは8192命令分の容量があり(容量を越えた分は先頭から上書き)、この中に記録した情報を[Trace]ウィンドウに表示することができます。
表示されるトレース内容は、次のとおりです。
・実行命令番号
・フェッチコードと逆アセンブル内容
・レジスタおよびコンディションフラグの内容
・メモリのアクセス内容(R/W、アドレス、データ)

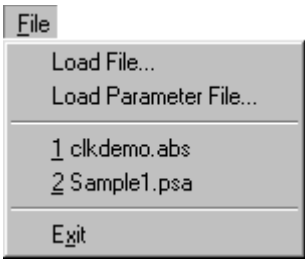
[Trace]ウィンドウは、tsコマンドによるトレースデータの検索結果の表示にも使用します。

※ 表示の更新
ターゲットプログラムの実行により、[Trace]ウィンドウの内容はクリアされます。プログラム実行を中断すると、[Trace]ウィンドウはトレースバッファの内容を表示します。

4.7 メニュー

ここでは、メニューバーの概要を説明します。
シミュレータのメニューバーには8つのメニュー項目があり、頻繁に使用するコマンドが設定されています。

[File]メニュー



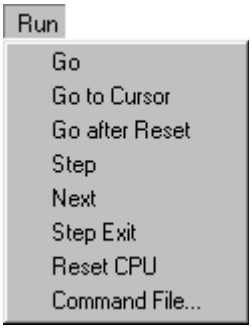
リストされているファイル名は最近ロードしたファイルです。ここからの選択でも、そのファイルを開くことができます。

[Load File...]
IEEE-695形式のアブソリュートオブジェクトファイル、モトローラS2形式のプログラムファイルまたはファンクションオプションファイルを読み込みます。この選択はlfコマンドを実行するのと同じ働きがあります。

[Load Parameter File...]
パラメータファイルを読み込みます。この選択はparコマンドを実行するのと同じ働きがあります。

[Exit]
シミュレータを終了します。この選択はqコマンドを実行するのと同じ働きがあります。

[Run]メニュー



[Go]
現在のPC(プログラムカウンタ)からターゲットプログラムを実行します。この選択はgコマンドを実行するのと同じ働きがあります。

[Go to Cursor]
現在のPCから、[Source]ウィンドウのカーソル位置(その行のアドレス)までターゲットプログラムを実行します。この選択はg <address>コマンドを実行するのと同じ働きがあります。このメニュー項目を選択するには、[Source]ウィンドウを開き、ブレークするアドレスの行をクリックしておく必要があります。

[Go after Reset]
CPUをリセット後、リセットベクタをフェッチしてターゲットプログラムを実行します。この選択はgrコマンドを実行するのと同じ働きがあります。

[Step]
現在のPCからターゲットプログラムを1ステップ実行します。この選択はsコマンドを実行するのと同じ働きがあります。

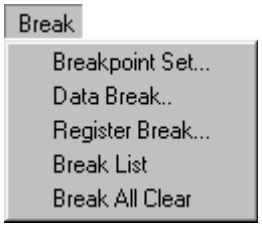
[Next]
現在のPCからターゲットプログラムを1ステップ実行します。実行命令がcars、carl、call、int命令の場合は、次のアドレスにリターンするまでを1ステップとみなし、それらのサブルーチンのステップをすべて実行します。この選択はnコマンドを実行するのと同じ働きがあります。

[Step Exit]
現在のPCからターゲットプログラムを実行します。開始位置がサブルーチン内の場合、親ルーチンにリターンしたところで実行を中断します。この選択はseコマンドを実行するのと同じ働きがあります。

[Reset CPU]
CPUをリセットします。この選択はrstコマンドを実行するのと同じ働きがあります。

[Command File...]
コマンドファイルを読み込み、記述されているコマンドを実行します。この選択はcom、cmwコマンドを実行するのと同じ働きがあります。

[Break]メニュー



[Breakpoint Set...]
PCブレークポイントをダイアログボックスを使用して設定/解除します。この選択はbpコマンドを実行するのと同じ働きがあります。

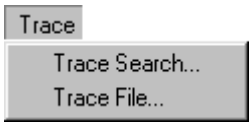
[Data Break...]
データブレーク条件をダイアログボックスを使用して設定/解除します。この選択はbdコマンドを実行するのと同じ働きがあります。

[Register Break...]
レジスタブレーク条件をダイアログボックスを使用して設定/解除します。この選択はbrコマンドを実行するのと同じ働きがあります。

[Break List]
設定されているすべてのブレーク条件を表示します。この選択はblコマンドを実行するのと同じ働きがあります。

[Break All Clear]
すべてのブレーク条件を解除します。この選択はbacコマンドを実行するのと同じ働きがあります。

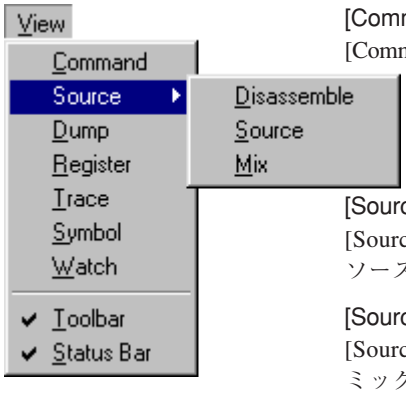
[Trace]メニュー



[Trace Search...]
トレースデータバッファ内のトレース情報を検索します。検索条件はダイアログボックスで指定します。この選択はtsコマンドを実行するのと同じ働きがあります。

[Trace File...]
[Trace]ウィンドウに表示したトレース情報の指定範囲をファイルに保存します。この選択はtfコマンドを実行するのと同じ働きがあります。

[View]メニュー



[Command]
[Command]ウィンドウをアクティブにします。

[Source - Disassemble]
[Source]ウィンドウを開いてアクティブにします。[Source]ウィンドウは逆アセンブル表示モードでプログラムを表示します。

[Source - Source]
[Source]ウィンドウを開いてアクティブにします。[Source]ウィンドウはソース表示モードでプログラムを表示します。

[Source - Mix]
[Source]ウィンドウを開いてアクティブにします。[Source]ウィンドウはミックス表示モードでプログラムを表示します。

[Dump]
[Dump]ウィンドウを開いてアクティブにします。[Dump]ウィンドウはメモリの内容をメモリの先頭から表示します。

[Register]
[Register]ウィンドウを開いてアクティブにします。[Register]ウィンドウは各レジスタの内容を表示します。

[Trace]
[Trace]ウィンドウを開いてアクティブにします。[Trace]ウィンドウはトレースデータバッファの内容を表示します。

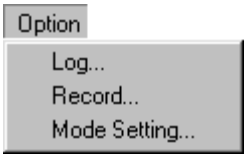
[Symbol]
[Symbol]ウィンドウを開いてアクティブにします。シンボルファイルが読み込まれていれば、その内容を表示します。

[Watch]
[Watch]ウィンドウを開いてアクティブにします。シンボルが登録されていれば、その内容を表示します。

[Toolbar]
ツールバーの表示/非表示を切り換えます。

[Status Bar]
ステータスバーの表示/非表示を切り換えます。

[Option]メニュー



[Log...]
ログ出力のON/OFFを切り換えます。この選択はlogコマンドを実行するのと同じ働きがあります。

[Record...]
実行コマンドのファイルへの記録を制御します。この選択はrecコマンドを実行するのと同じ働きがあります。

[Mode Setting...]
トレース/カバレッジ機能のON/OFF、ステップ表示モード、cmwコマンドの実行間隔を設定します。この選択はmdコマンドを実行するのと同じ働きがあります。

[Window]メニュー

[Cascade]
開いているウィンドウを斜めに整列させます。

[Tile]
開いているウィンドウを縦に整列させます。

このメニューには、現在開いているウィンドウ名が表示されます。いずれかを選択すると、そのウィンドウがアクティブになります。

[Help]メニュー

[About SIM88...]
シミュレータのアバウトダイアログボックスを表示します。

4.8　ツールバー

ここでは、ツールバーの概要を説明します。
シミュレータのツールバーには12個のボタンがあり、頻繁に使用するコマンドが設定されています。

クリックすることにより、指定の機能を実行します。

[Load File]ボタン
IEEE-695形式のアブソリュートオブジェクトファイル、モトローラS2形式のプログラムファイルまたはファンクションオプションファイルを読み込みます。この選択はIfコマンドを実行するのと同じ働きがあります。

[Load Parameter]ボタン
パラメータファイルを読み込みます。この選択はparコマンドを実行するのと同じ働きがあります。

[Key Break]ボタン
ターゲットプログラムの実行を強制的にブレークします。この機能はプログラムが永久ループ状態になった場合などにブレークさせることができます。

[Break]ボタン
[Source]ウィンドウ上のカーソルが置かれた行のアドレスに対し、ブレークポイントの設定と解除を行うのに使用します。[Source]ウィンドウが開いているときのみ有効です。

[Break All Clear]ボタン
すべてのブレーク条件を解除します。この選択はbacコマンドを実行するのと同じ働きがあります。

[Go]ボタン
現在のPC(プログラムカウンタ)からターゲットプログラムを実行します。この選択はgコマンドを実行するのと同じ働きがあります。

[Go to Cursor]ボタン
現在のPCから、[Source]ウィンドウのカーソル位置(その行のアドレス)までターゲットプログラムを実行します。この選択はg <address>コマンドを実行するのと同じ働きがあります。
このボタンを選択するには、[Source]ウィンドウを開き、ブレークするアドレスの行をクリックしておく必要があります。

[Go after Reset]ボタン
CPUをリセット後、リセットベクタをフェッチしてターゲットプログラムを実行します。この選択はgrコマンドを実行するのと同じ働きがあります。

[Step]ボタン
現在のPCからターゲットプログラムを1ステップ実行します。この選択はsコマンドを実行するのと同じ働きがあります。

[Next]ボタン
現在のPCからターゲットプログラムを1ステップ実行します。実行命令がcars、carl、call、int命令の場合は、次のアドレスにリターンするまでを1ステップとみなし、それらのサブルーチンのステップをすべて実行します。この選択はnコマンドを実行するのと同じ働きがあります。

[Step Exit]ボタン
現在のPCからターゲットプログラムを実行します。開始位置がサブルーチン内の場合、親ルーチンにリターンしたところで実行を中断します。この選択はseコマンドを実行するのと同じ働きがあります。

[Reset CPU]ボタン
CPUをリセットします。この選択はrstコマンドを実行するのと同じ働きがあります。

4.9 コマンド実行方法

デバッグ機能はすべてデバッグコマンドによって実行できます。ここでは、コマンドを実行させる方法を説明します。

4.9.1 コマンドのキーボード入力

[Command]ウィンドウを選択してください([Command]ウィンドウ上をクリック)。その中の最終行にプロンプト">"が表示され、その後にカーソルが点滅していればコマンドが入力できる状態にあります。
そこに、デバッグコマンドを入力してください。コマンドは大文字でも、小文字でも受け付けます。

● コマンド入力の一般形

＞コマンド [パラメータ [パラメータ... パラメータ]] ↲

・コマンドとパラメータ間にはスペースが必要です。
・パラメータ間にはスペースが必要です。

入力ミスの修正には矢印キー、[Back Space]キー、[Delete]キーが使用できます。
最後に[Enter]キーを入力すると、そのコマンドを実行します(ガイダンス付きのコマンドは、表示に従って必要なデータを入力した時点で実行されます)。
入力例:

```
>g↲                    (コマンドのみの入力)
>com test.cmd↲         (コマンドとパラメータの入力)
```

● ガイダンス付きのコマンド入力

パラメータが指定されないと実行できないコマンドや、既存のデータを変更するコマンドはコマンドのみの入力でガイダンスモードとなります。ガイダンスが表示されますので、そこにパラメータを入力してください。
入力例:

```
>lf↲
File name   ? :test.psa↲        ...ガイダンスに従ってデータ(アンダーライン部)を入力
>
```

・パラメータ入力が必須のコマンド
上記例のlfコマンドはプログラムファイルを読み込むコマンドです。このような、パラメータの入力が必須のコマンドはパラメータを入力後、[Enter]キーを押すと実行されます。複数のパラメータを持つコマンドでは、次のガイダンスが表示されますので、最後のパラメータまで順次入力してください。いずれかのガイダンスで[Enter]キーのみを入力すると、そのコマンドはキャンセルされ実行されません。

・既存のデータを確認して置き換えるコマンド
メモリやレジスタを1つずつ書き換えるコマンドではガイダンスをスキップ(その内容を変更しない)、1つ前のガイダンスに戻す、および途中で入力を終了することができます。

[Enter]キー	入力をスキップ
[^]キー	1つ前のガイダンスに戻す
[q]キー	入力を終了する

入力例:

```
>de↲                                  ...メモリを変更するコマンド
  Data enter address ? :001000↲       ...開始アドレスを入力
  001000 A:1↲                         ...001000H番地を1に変更
  001001 A:^↲                         ...1つ前のアドレスに戻す
  001000 1:0↲                         ...001000H番地の入力をし直す
  001001 A:↲
  001002 A:↲
  001001 A:q↲                         ...入力を終了
  >
```

● パラメータの数値データ形式

パラメータとして入力する数値は、ほとんどのコマンドが16進数のみを受け付けます。
ただし、一部のコマンドのパラメータは、10進数または2進数で指定します。
数値として有効な文字は次のとおりです。
16進数:　0～9、a～f、A～F、*
10進数:　0～9
2進数:　0、1、*
(*はデータパターン指定でマスクするビットに使用します。)

● シンボルによる指定

シンボル情報を含んだIEEE-695形式のアブソリュートオブジェクトファイル(.abs)、またはシンボルファイル(.sy)が読み込まれている場合、シンボルを使用してアドレスを指定することができます。
入力例:

```
>u  Main↵                    ...プログラムをMainというラベルから表示
```

＊ シンボルファイル(.sy)はターゲットプログラム(モトローラS2)のロード時に自動的に読み込まれます。ただし、そのためには、ターゲットプログラムと同じ名称のシンボルファイルをターゲットプログラムと同じディレクトリに用意しておく必要があります。IEEE-695形式のプログラムファイルを指定した場合は読み込まれません。

注:・指定したシンボルがない場合、sim88はその指定文字列を16進数として扱います(ABCなど)。ただし、文字列が16進数の指定文字以外を含んでいる場合はエラーとなります。

・Cソースのコンパイル時に-O1オプションを指定すると、ソースに記述されているシンボルがコードの最適化によって実際には使用されない場合が発生します。この場合、-gオプションを指定してもそのシンボルのデバッグ情報は.absファイルに出力されません。

例:
```
int x,y,xy;
x = GLOBAL_X * 100;
y = GLOBAL_Y * 100;
xy = x * y;
```

この例では最適化により変数xyは存在しなくなりますので、デバッグ時にxyの内容を参照することはできません。
-O0オプション(最適化OFF)を指定して作成した実行ファイルを評価後、-O1オプション(最適化ON)を指定して作成し直した場合は動作保証できませんので、再検証を行ってください。

● [Enter]キーによる連続実行

以下のコマンドは一度実行すると、その後は[Enter]キーのみで連続して実行できます。プログラム実行コマンドは同じ動作を繰り返します。表示コマンドは前に表示した次の部分を連続して表示します。
実行コマンド: g, s, n, se, com
表示コマンド: u, dd, td,

この連続実行機能は他のコマンドを実行すると解除されます。

4.9.2 メニュー、ツールバーからの実行

4.7項、4.8項に示したように、メニューとツールバーには頻繁に使用するコマンドが登録されています。メニューコマンドを選択するか、ツールバーボタンをクリックするだけで、指定のコマンドを実行できます。表4.9.2.1に登録されているコマンドの一覧を示します。

表4.9.2.1　メニュー、ツールバーで指定可能なコマンド

コマンド	機能	メニュー	ボタン
lf	プログラムファイルのロード	[File I Load File...]	
par	パラメータファイルのロード	[File I Load Parameter File...]	
g	プログラムの連続実行	[Run I Go]	
g <address>	プログラムを<address>まで連続実行	[Run I Go to Cursor]	
gr	CPUリセット後、プログラムの連続実行	[Run I Go after Reset]	
s	ステップ実行	[Run I Step]	
n	ステップ&サブルーチンスキップ	[Run I Next]	
se	サブルーチンの終了	[Run I Step Exit]	
com	コマンドファイルのロード、実行	[Run I Command File...]	–
cmw	コマンドファイルウェイト付き実行	[Run I Command File...]	–
rst	CPUリセット	[Run I Reset CPU]	
bp, bc (bpc)	PCブレークポイント設定/解除	[Break I Breakpoint Set...]	
bd, bdc	データブレーク条件の設定/解除	[Break I Data Break...]	–
br, brc	レジスタブレーク条件の設定/解除	[Break I Register Break...]	–
bl	ブレーク条件の表示	[Break I Break List]	–
bac	全ブレーク条件の解除	[Break I Break All Clear]	
ts	トレース情報の検索	[Trace I Trace Search...]	–
tf	トレース情報のファイルへの保存	[Trace I Trace File...]	–
u	逆アセンブル表示	[View I Source I Disassemble]	 *
sc	ソース表示	[View I Source I Source]	 *
m	ミックス表示	[View I Source I Mix]	 *
dd	メモリダンプ	[View I Dump]	–
rd	レジスタ値の表示	[View I Register]	–
td	トレース情報の表示	[View I Trace]	–
sy	シンボル一覧の表示	[View I Symbol]	–
w	シンボル情報の表示	[View I Watch]	–
	シンボルの登録	–	 *
log	ログ出力ON/OFF	[Option I Log...]	–
rec	実行コマンドの記録	[Option I Record...]	–
md	モード設定	[Option I Mode Setting...]	–

* [Source]ウィンドウ上のボタン

4.9.3 コマンドファイルによる実行

一連のデバッグコマンドを記述したコマンドファイルを読み込んで、それらのコマンドを実行させることができます。

● コマンドファイルの作成

コマンドファイルはエディタでテキストファイルとして作成してください。
ファイル名の拡張子は".cmd"とします。
コマンドファイルは、recコマンドによっても作成できます。recコマンドを使用すると、シミュレータで実際に実行したコマンドをコマンドファイルに記録することができます。

● コマンドファイル例

次の例は、プログラムファイルを読み込み、ブレークポイントを設定して実行させるためのコマンド群です。
例: ファイル名=start.cmd

```
lf test.psa
bp 0004d7
g
```

コマンドファイルにはガイダンスモードに対応した記述も可能です。その場合は、ガイダンス入力の各項目ごとに改行して記述してください。

● コマンドファイルの読み込み/実行

コマンドファイルの実行用に、comコマンドとcmwコマンドが用意されています。
comコマンドは指定されたファイルを読み込み、その中のコマンドを記述順に指定された間隔(0～256秒)で実行します。cmwコマンドも同様ですが、個々のコマンドはmdコマンドで指定された間隔(1～256秒)で実行されます。
例: `com start.cmd`

```
cmw test.cmd
```

コマンドファイルに記述されたコマンドは[Command]ウィンドウに表示されます。

● 制限事項

コマンドファイル内から別のコマンドファイルを読み込むことも可能です。ただし、最大5階層までに制限されます。6階層目のcom/cmwコマンドが現れるとエラーとなり、それ以後の実行を中止します。

4.9.4 ログファイル

実行したコマンドと実行結果を、テキスト形式のログファイルとして保存することができます。これによって、後からデバッグの手順と内容を確認することができます。
保存の対象となるのは[Command]ウィンドウに表示された内容です。

● コマンド例

```
>log tst.log
```

logコマンドでログモードに設定後(出力開始後)は、logコマンドがトグル動作(ログモード/出力ON↔通常モード/出力OFF)となりますので、必要な部分のみの出力が簡単に行えます。

● ログモードでの[Command]ウィンドウの表示

ログモードでは、[Command]ウィンドウに表示される内容が通常の場合と異なります。

(1)各ウィンドウが開いている場合のコマンド実行時
(ウィンドウに結果が表示されるコマンドをそのウィンドウが開いている状態で実行した場合)
通常モード: 表示先のウィンドウの内容が更新されます。[Command]ウィンドウに実行結果は表示されません。
ログモード: ウィンドウに表示される情報と同等の内容が[Command]ウィンドウにも表示されます。ただし、ウィンドウのスクロール操作、ウィンドウを開いたことによる表示内容は、[Command]ウィンドウには表示されません。

(2)各ウィンドウが閉じている場合のコマンド実行時
表示先のウィンドウが閉じている場合、ログモード/通常モードにかかわらず実行結果が[Command]ウィンドウに表示されます。

4.10 デバッグ機能

ここでは、デバッグ機能の概要を機能別に説明します。

4.10.1 ファイルの読み込み

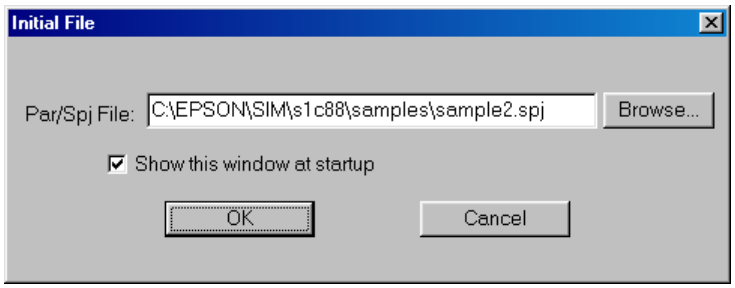
表4.10.1.1にシミュレータが読み込むファイルの一覧と読み込みコマンドを示します。

表4.10.1.1　ファイルと読み込みコマンド一覧

ファイル	拡張子	生成ツール	コマンド	メニュー	ボタン
1. パラメータファイル	.par	–	par	[File	Load Parameter File...]	
2. LCDパネル定義ファイル	.lcd	LcdUtil	–	–	–
3. コンポーネントマップファイル	.cmp	–	–	–	–
4. ポート設定ファイル	.prt	PrtUtil	–	–	–
5. シミュレータプロジェクトファイル	.spj	–	par	[File	Load Parameter File...]	
6. IEEE-695アブソリュート オブジェクトファイル	.abs	lc88	lf	[File	Load File...]	
7. モトローラS2プログラムファイル	.psa	fil88xxx	lf	[File	Load File...]	
8. ファンクションオプションファイル	.fsa	fog88xxx or winfog	lf	[File	Load File...]	
9. シンボルファイル	.sy	sy88, sym88	–	–	–
10. コマンドファイル	.cmd	–	com/cmw	[Run	Command File...]	–

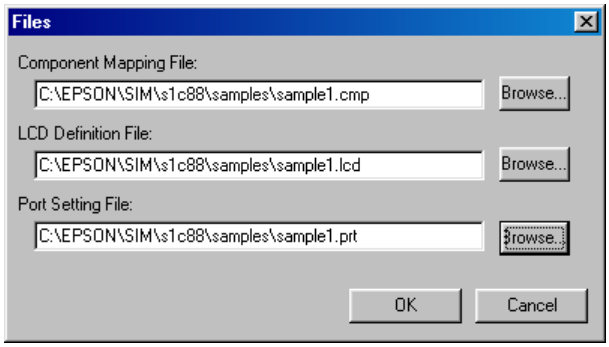
1～4はシミュレータが起動するために必要です。

シミュレータの最初の起動時は、次のダイアログボックスで1のパラメータファイルまたは5のシミュレータプロジェクトファイルを選択する必要があります。



パラメータファイルまたはシミュレータプロジェクトファイルの名称をテキストボックスに入力するか、[Browse...]ボタンで選択してください。
[Show this window at startup]チェックボックスをオフにすると、このダイアログボックスは次回の起動時から表示されなくなり、同じファイルが選択されます。
パラメータファイルをロードすると、その時点でシミュレータがリセットされます。パラメータファイルで設定されたメモリマップ情報はmaコマンドで表示させることができます。パラメータファイルについては、S5U1C88000C Manual IIの"88xxx.parファイル"を参照してください。

パラメータファイルを選択すると2～4のファイルを選択する次のダイアログボックスが表示されます。



シミュレータプロジェクトファイルを選択した場合は、その中で2～4のファイルが指定されているためダイアログボックスは表示されません。

コンポーネントマップファイル、LCDパネル定義ファイル、ポート設定ファイルをそれぞれのテキストボックスに入力するか、[Browse...]ボタンで選択してください。これらのファイルを一度選択すると、次回からはテキストボックスに同じファイル名が表示され、[OK]のクリックのみで済みます。

lfコマンドはIEEE-695形式のアブソリュートオブジェクトファイル(.abs)、モトローラS2形式のプログラムファイル(.psa)、ファンクションオプションHEXファイル(.fsa)をロードします。シミュレータはこれらのファイルを指定された拡張子で区別します。ソースレベルデバッグを行うためには、IEEE-695形式でデバッグ情報を含んだファイルを読み込む必要があります。

シンボルファイルはモトローラS2形式のプログラムファイルをデバッグする際に、アドレスをソース作成時のシンボルを使用して指定するために必要で、ロードしなくてもデバッグは行えます。このファイルはlfコマンドによるモトローラS2形式プログラムファイルのロード時に同時に読み込まれます。ただし、プログラムファイルと同じ名称(拡張子は.sy)のシンボルファイルをプログラムファイルと同じディレクトリに用意しておく必要があります。シンボルファイルが読み込まれている場合、その内容は[Symbol]ウィンドウまたはsyコマンドで確認できます。

IEEE-695形式でシンボル情報を含んだアブソリュートオブジェクトファイルを読み込んだ場合は、そのファイルのみでシンボルが使用可能なため、シンボルファイルは読み込まれません。

コマンドファイルは4.9.3項で説明したとおりです。

4.10.2 ソース表示およびシンボリックデバッグ機能

本シミュレータは、Cソースを表示させたデバッグが可能です。また、シンボル名を使用してアドレスの指定も行えます。

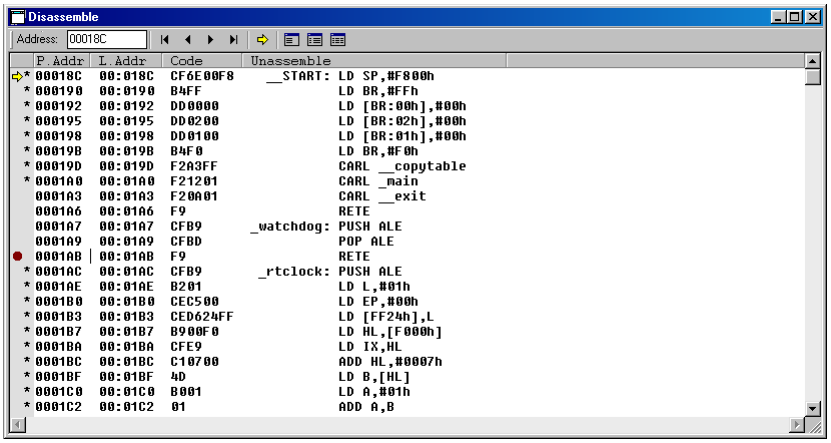
● プログラムコードの表示

[Source]ウィンドウは指定された表示モードでプログラムを表示します。表示モードは、逆アセンブル表示モード、ソース表示モード、ミックス表示モードの3種類から選択できます。

表4.10.2.1　表示モード切り換えコマンド

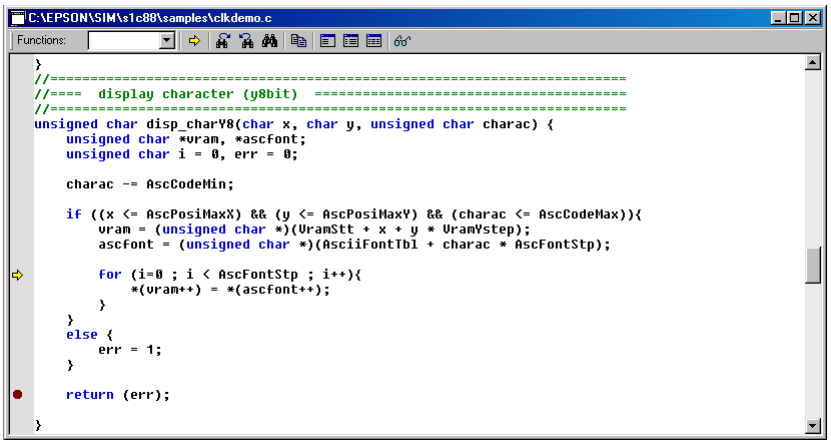
機能	コマンド	メニュー	ボタン
逆アセンブル表示モード	u	[View I Source I Disassemble]	
ソース表示モード	sc	[View I Source I Source]	
ミックス表示モード	m	[View I Source I Mix]	

(1) 逆アセンブル表示モード



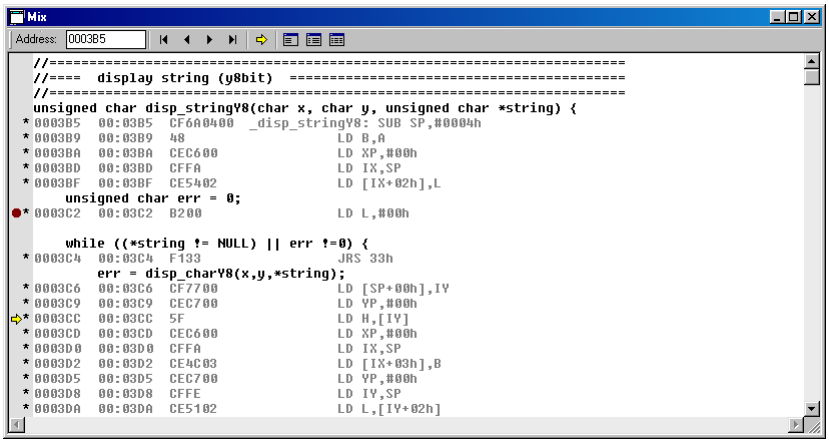
このモードでは、オブジェクトコードをニーモニックに逆アセンブルして表示します。

(2) ソース表示モード



このモードでは、現在のPCアドレス上のコードを含むソースが表示されます。このモードは、ソースデバッグ情報を含むIEEE-695形式アブソリュートオブジェクトファイルをロードしている場合にのみ指定可能です。

(3) ミックス表示モード



このモードでは、プログラムソースとそれに対応するオブジェクトの逆アセンブル内容が表示されます。このモードは、ソースデバッグ情報を含むIEEE-695形式のアブソリュートオブジェクトファイルをロードしている場合にのみ指定可能です。

表示内容とウィンドウ上の操作については、"4.6.5 [Source]ウィンドウ"を参照してください。

● シンボル参照

IEEE-695形式のオブジェクトファイル(.abs)を読み込んでデバッグを行う場合、ソースファイルで定義されたシンボルを使用してアドレスを指定することができます。パラメータに<address>を持つコマンドを[Command]ウィンドウ上で入力する際、あるいはアドレスをダイアログで指定する際に使用することができます。ただし、オブジェクトファイルにデバッグ情報が含まれている必要があります。

モトローラS2形式のプログラムファイル(.psa)を読み込んでシンボルを使用したデバッグを行う場合は、プログラムファイルと同名のシンボルファイルを同じディレクトリに用意しておく必要があります。シンボルファイルはプログラムファイルのロード時に自動的に読み込まれます。

デバッグ中のプログラムで使用しているシンボルと定義されたアドレスは、[Command]ウィンドウまたは[Symbol]ウィンドウに表示させることができます。

表4.10.2.2　シンボルリスト表示コマンド

機能	コマンド	メニュー	ボタン
シンボルリストの表示	sy	[View I Symbol]	–

4.10.3 メモリデータ、レジスタの表示と変更

シミュレータはメモリ、レジスタに対する操作機能を持っています。メモリ領域はパラメータファイルで与えられるマップ情報に従ってシミュレータに設定されます。

● メモリの操作

メモリ（ROM領域、RAM領域、表示メモリ、I/Oメモリ）に対しては以下の操作が行えます。

表4.10.3.1　メモリ操作コマンド

機能	コマンド	メニュー	ボタン
メモリダンプ	dd	[View	Dump]	–
データの入力/変更	de	–	–
指定領域の書き換え	df	–	–
指定領域のコピー	dm	–	–
データの検索	ds	–	–

(1) メモリダンプ

メモリの内容を指定サイズ（Byte, Word, Long, Float, Double）の16進ダンプ形式で表示します。[Dump]ウィンドウが開いていれば[Dump]ウィンドウの内容を更新し、開いていなければ[Command]ウィンドウに表示します。

(2) データの入力/変更

16進データを入力して、指定アドレスのデータを書き換えます。[Dump]ウィンドウ上で直接変更することもできます。

(3) 指定領域の書き換え

指定した領域を指定したデータですべて書き換えます。

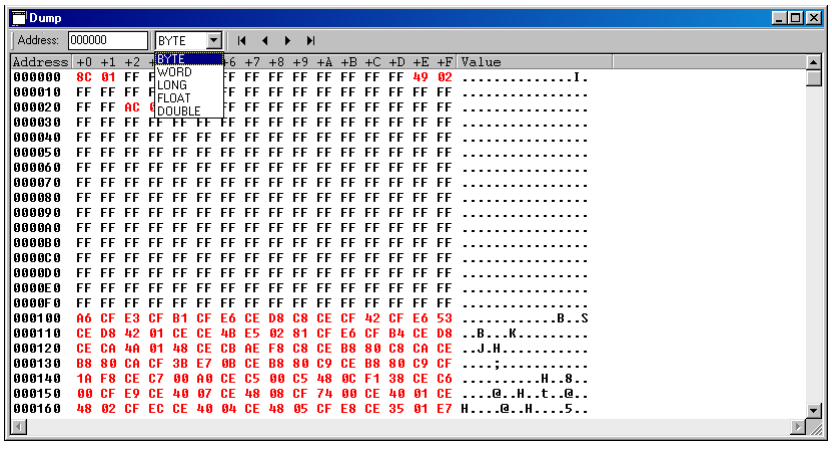
(4) 指定領域のコピー

指定した領域の内容を、別の領域にコピーします。

(5) データの検索

指定の領域内で指定のデータを検索できます。検索結果は最大256個まで[Command]ウィンドウに表示できます。また、[Dump]ウィンドウが現在表示している範囲内で指定データが見つかった場合、[Dump]ウィンドウ上の該当データは緑色で表示されます。

[Dump]ウィンドウの表示内容とウィンドウ上の操作については、"4.6.6 [Dump]ウィンドウ"を参照してください。



● レジスタの操作

レジスタに対しては以下の操作が行えます。

表4.10.3.2　レジスタ操作コマンド

機能	コマンド	メニュー	ボタン
レジスタの表示	rd	[View I Register]	–
レジスタ値の変更	rs	–	–

(1) レジスタの表示

レジスタおよびレジスタで間接アドレッシングされるメモリの内容を[Register]ウィンドウまたは[Command]ウィンドウに表示させることができます。

レジスタ: PC, SP, IX, IY, A, B, H, L, BR, CB, NB, EP, XP, YP, SC (I1, I0, U, D, N, V, C, Z), CC (F3, F2, F1, F0)

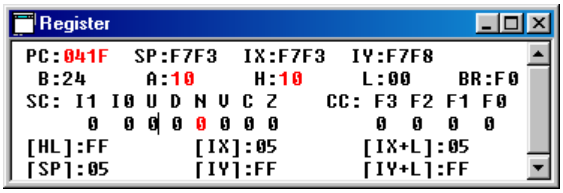
メモリ:　[HL], [SP], [IX], [IY], [IX+L], [IY+L]

(2) レジスタ値の変更

上記レジスタの内容を任意の値に設定できます。

レジスタ値は[Register]ウィンドウ上で直接変更することもできます。

[Register]ウィンドウの表示内容とウィンドウ上の操作については、"4.6.7 [Register]ウィンドウ"を参照してください。



4.10.4 プログラムの実行

シミュレータにはターゲットプログラムを連続実行およびシングルステップ実行させる機能があります。

● 連続実行

(1) 連続実行の種類

2種類の連続実行機能があります。
・現在のPCアドレスから連続実行
・CPUをリセット後に連続実行

表4.10.4.1　連続実行コマンド

機能	コマンド	メニュー	ボタン
現在のPCアドレスから連続実行	g	[Run | Go]	
CPUをリセット後に連続実行	gr	[Run | Go after Reset]	

(2) 連続実行の停止

連続実行コマンド(g <アドレス1><アドレス2>)によって、その実行中にのみ有効なテンポラリブレークアドレスを2ヵ所まで指定することができます。
テンポラリブレークアドレスは[Source]ウィンドウで指定することもできます。[Source]ウィンドウ上のブレークさせる行にカーソルを置いて[Go to Cursor]ボタンをクリックすると、プログラムは現在のPCから実行を開始し、カーソル位置の命令を実行後にブレークします。

なお、ソース表示モードでCソースを表示している場合は、有効なオブジェクトコードに展開されているソース行にカーソルを置くことが必要です。コメント行や宣言文など、オブジェクトコードにコンパイルされないソース行にカーソルがある場合、[Go to Cursor]ボタンをクリックしてもプログラムは実行されません(PCによるブレーク機能の説明も参照してください)。

このテンポラリブレーク以外では、実行中のプログラムは次のいずれかの要因によってブレークするまで停止しません。
・ブレーク設定コマンドで設定したブレーク条件が成立
・[Key Break]ボタンのクリック
・ 未定義領域への不当アクセス

 [Key Break]ボタン

※プログラムが停止しない場合は、このボタンで強制ブレークさせることができます。

注: Cソース表示モードでプログラムの実行を停止すると、停止したアドレスが含まれるオブジェクトのソースが表示されます。ただし、停止したアドレスにソースがない場合、[Source Files]ダイアログボックスが表示され、ソースファイルの選択が必要になります。

(3) LCDパネル、外部入出力のシミュレーション

シミュレータ起動時にターゲットに合わせて作成したコンポーネントマップファイル、LCDパネル定義ファイル、ポート設定ファイルを読み込んでおくことにより、[LCD]ウィンドウはプログラムの制御に従った表示を行います。また、コンピュータのキーボードを使用してキー入力もシミュレートできます。これらの機能については、"4.6.3 [LCD]ウィンドウ"を参照してください。また、[I/O Terminal]ウィンドウからIOTファイルを読み込んで、シリアル/汎用ポートの入出力、A/D変換をシミュレートすることもできます。詳細については、"4.6.4 [I/O Terminal]ウィンドウ"を参照してください。

● シングルステップ実行

(1) シングルステップ実行の種類

3種類のシングルステップ実行機能があります。

・Cステートメント/命令をシングルステップ実行(STEP)
Cソース表示モードではCソース行単位にシングルステップで実行します。
逆アセンブル表示またはミックス表示モードでは命令単位にシングルステップで実行します。

・関数、サブルーチン以外をシングルステップ実行(NEXT)
Cソース表示モードでは、関数呼び出しがあっても関数内には入らず、リターンするまでを1ステップとして実行します。関数呼び出し以外は通常のシングルステップ実行と同様です。
逆アセンブル表示またはミックス表示モードでは、cars、carl、call、int命令を、リターン命令で次のステップに戻るまでを1ステップとして実行します。他の命令は、通常のシングルステップ実行と同様です。

・関数、サブルーチンの終了(STEP EXIT)
Cソース表示モードでは、現在の関数から上位レベルの関数に戻るまでを連続実行し、リターン後に停止します。メイン関数内では実行しないでください。
逆アセンブル表示またはミックス表示モードでは、現在のサブルーチンからリターン命令で上位レベルに戻るまでを連続実行し、リターン後に停止します。最上位レベルではgコマンドと同様の機能です。下位レベルのサブルーチンをコールしてリターンしても停止しません。

いずれの場合も現在のPCから実行します。

表4.10.4.2　シングルステップコマンド

機能	コマンド	メニュー	ボタン
シングルステップ実行	s	[Run | Step]	
関数/サブルーチン以外を シングルステップ実行	n	[Run | Next]	
関数/サブルーチンの終了	se	[Run | Step Exit]	

s およびnコマンド入力では、実行するステップ数を最大65,535ステップまで指定することができます。
メニューコマンド、ツールバーでは1ステップずつ実行します。
以下の場合には、シングルステップ実行が指定のステップ数の実行前に終了します。
・[Key Break]ボタンのクリック
・ 未定義領域への不当アクセス

PCブレーク、データブレーク等のユーザ設定ブレークでは停止しません。

 [Key Break]ボタン

※プログラムが停止しない場合は、このボタンで強制ブレークさせることができます。

(2) ステップ動作中の表示

シミュレータの初期設定では、次のように表示を更新します。
[Source]、[Register]、[Dump]ウィンドウの表示内容は最終ステップを実行後に更新されます。[Register]ウィンドウが閉じている場合は、レジスタの値が最終ステップを実行後に[Command]ウィンドウに表示されます。
この表示モードを、各ステップごとに更新するように、mdコマンドで切り換えることもできます。

(3) ステップ動作中のキー入力シミュレーション

[LCD]ウィンドウの[Key List]ボタンで表示される[Key List]ウィンドウ上でキー入力状態が設定でき、その設定状態はステップ動作中も保持されます。

(4) Cソースのシングルステップ実行に関する注意事項

Cソース表示モードでのシングルステップ実行は基本的にソース行単位になります。ただし、対応するオブジェクトコードのないソース行やユーザーソースのない箇所(インラインアセンブルやコンパイラが自動生成した関数など)は、その次の行まで実行されます。
これに伴い、Cステートメントの書き方でもステップ回数が変わります。
例: `for(x=0; x<10; x++) a[x]=x;`　　... 1ステップで実行されます。

```
for(x=0; x<10; x++)
    a[x]=x;
```
... for文を抜けるのに20ステップの実行が必要です。

● CPUのリセット

表4.10.4.3　CPUリセットコマンド

機能	コマンド	メニュー	ボタン
CPUリセット	rst	[Run	Reset CPU]	
CPUをリセット後に連続実行	gr	[Run	Go from Reset]	

CPUはgrコマンドの実行時、およびrstコマンドの実行によりリセットされます。
CPUのリセットによる初期設定内容は以下のとおりです。

(1) CPUの内部レジスタ、メモリ

イニシャルリセットによりCPUの内部レジスタは以下のように初期化されます。

表4.10.4.4　初期設定値

レジスタ名称	記号	ビット長	初期値
データレジスタA	A	8	不定
データレジスタB	B	8	不定
インデックス(データ)レジスタL	L	8	不定
インデックス(データ)レジスタH	H	8	不定
インデックスレジスタIX	IX	16	不定
インデックスレジスタIY	IY	16	不定
プログラムカウンタ	PC	16	不定*
スタックポインタ	SP	16	不定
ベースレジスタ	BR	8	不定
ゼロフラグ	Z	1	0
キャリーフラグ	C	1	0
オーバーフローフラグ	V	1	0
ネガティブフラグ	N	1	0
デシマルフラグ	D	1	0
アンパックフラグ	U	1	0
インタラプトフラグ0	I0	1	1
インタラプトフラグ1	I1	1	1
ニューコードバンクレジスタ	NB	8	01H
コードバンクレジスタ	CB	8	不定*
エクスパンドページレジスタ	EP	8	00H
IX用エクスパンドページレジスタ	XP	8	00H
IY用エクスパンドページレジスタ	YP	8	00H

* リセット例外処理によって、0バンクのメモリの先頭(000000H～000001H)に格納されている値がPCにロードされます。また、このとき同時にNBの初期値01HがCBにロードされます。

内蔵RAMおよび外部RAMはイニシャルリセット時に初期化されません。
内蔵のI/Oメモリについては、それぞれの初期値に設定されます。

(2) [Source]ウィンドウ、[Register]ウィンドウを再表示

PCが再設定されるため、[Source]ウィンドウはそのアドレスから再表示されます。
[Register]ウィンドウも上記の設定で再表示されます。

4.10.5 ブレーク機能

ターゲットプログラムは次の要因により実行を中断します。
・ ブレーク設定コマンドの条件成立
・ [Key Break]ボタンのクリック
・ 未定義領域への不当アクセス

● コマンドによるブレーク機能

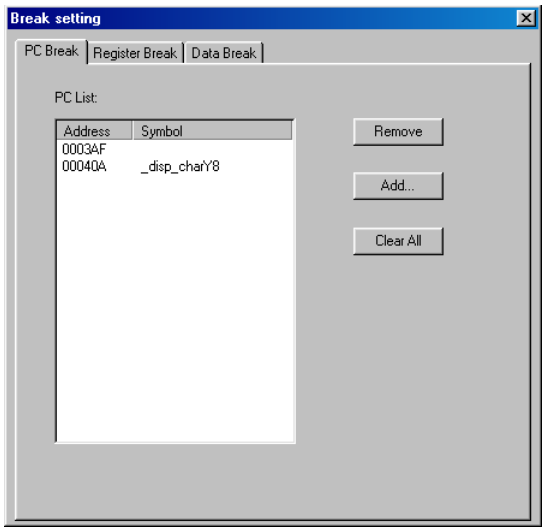
シミュレータはコマンドによってブレーク条件が設定できる3種類のブレーク機能を持っています。それぞれ、条件が成立すると実行中のプログラムはブレークします。

(1) PCによるブレーク機能

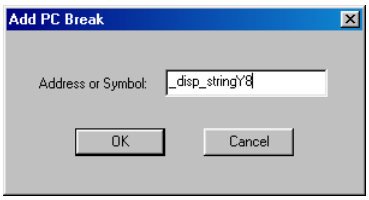
PCが設定したアドレスに一致するとブレークする機能です。プログラムは、そのアドレスの命令を実行後にブレークします。PCブレークポイントは最大64ヵ所のアドレスに設定できます。

表4.10.5.1　ブレークポイント設定コマンド

機能	コマンド	メニュー	ボタン
ブレークポイントの設定	bp	[Break	Breakpoint Set...]	
ブレークポイントの解除	bc (bpc)	[Break	Breakpoint Set...]	



[Break]メニューから[Breakpoint Set...]を選択すると、[Break setting]ダイアログボックスが現れ、[PC Break]タブの画面に、設定されているPCブレークポイントの一覧が表示されます。
PCブレークポイントを追加するには、[Add...]ボタンをクリックします。
[Add PC Break]ダイアログボックスが表示されますので、PCブレークポイントに設定するアドレス(16進数)またはシンボルを入力して[OK]をクリックします。



ブレークポイントを解除するには、一覧からそのアドレスを選択し、[Remove]ボタンをクリックします。[Clear All]ボタンをクリックすると、設定されているすべてのPCブレークポイントが解除されます。

※ 連続実行コマンド(g)で指定可能なテンポラリブレークアドレスは、ブレークポイントリストに設定されたアドレスには影響を与えません。

PCブレークポイントに設定されたアドレスは、[Source]ウィンドウ上の行の先頭に●が表示されます。
Cソース表示の例

```
    }

    return (err);

}
```

逆アセンブル表示の例

```
  0001A7  00:01A7  CFB9      _watchdog: PUSH ALE
  0001A9  00:01A9  CFBD                 POP ALE
● 0001AB | 00:01AB  F9                   RETE
* 0001AC  00:01AC  CFB9       _rtclock: PUSH ALE
```

[Break]ボタンを使用して簡単にブレークポイントの設定と解除を行うこともできます。

 [Break]ボタン

[Source]ウィンドウ上で、ブレークポイントに設定する行をクリック(カーソルを移動)し、[Break]ボタンをクリックします。その行の先頭に●マークが表示され、そのアドレスはブレークポイントリストに登録されます。●で始まる行をクリックして[Break]ボタンをクリックすると、ブレーク設定が解除され、アドレスがブレークポイントリストから削除されます。

ソース表示モード時のブレークポイント設定

ソース表示モードの[Source]ウィンドウ上では、ブレークポイントを設定できる行と、できない行があります。実コードが生成されていないソース行には設定できません。

```
例: 1    void func(void)        // NG
   2     {                      // OK
   3           int a;           // NG
   4           int x=0;         // OK
   5           a = x;           // OK
   6     }                      // OK
```

1は関数の宣言で、実コードがない(アセンブラのラベル宣言と同じです)ため設定できません。
3は変数の宣言で、実コードがないため設定できません。
4は変数の宣言ですが、初期化のコードが生成されるため設定できます。
2は設定できます。ただし、ブレークポイントは4 (その関数の先頭の命令)に設定されます。
5は実コードのある有効な行ですので、設定できます。
6は関数の終了(ニーモニックのretと等価)ですので、設定できます。

ただし、コンパイル時に最適化を行うとブレークポイントが設定できなくなることがあります。上記例では、各行の実行により得られるものが何もない(ローカル変数の書き換えのみでグローバル変数の書き換えがない)ため、最適化により実コードがなくなる可能性があります。

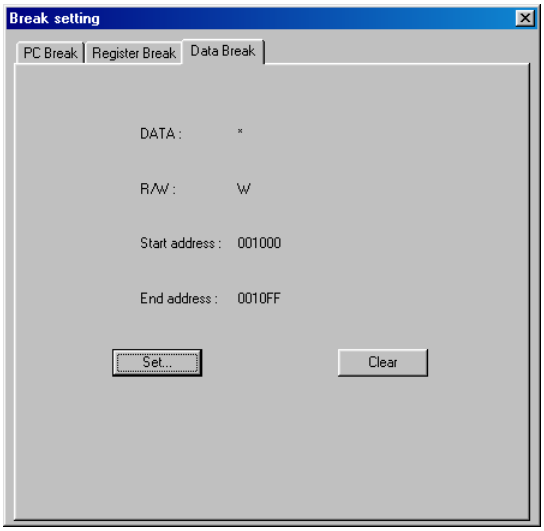
[Go to Cursor]ボタンで停止できる行の条件も同じです。

(2) データブレーク機能

データブレークは、指定したメモリ領域内をアクセスした場合にブレークを発生させる機能です。アクセスを監視するメモリ領域のほかに、読み出し、書き込みのどちらが行われた場合にブレークさせるか、またそのデータの内容まで指定することができます。読み出し/書き込み条件はマスク可能で、どちらが行われた場合でもブレークさせることができます。同様にデータ条件もビット単位のマスクが可能です。
ブレークは上記の条件を満たす動作が行われたサイクルの終了時に発生します。

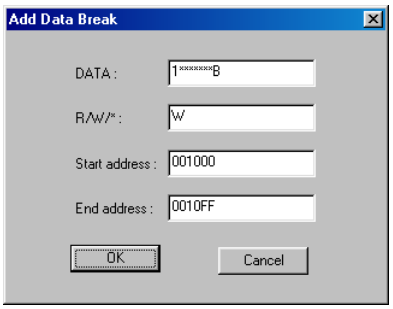
表4.10.5.2　データブレーク設定コマンド

機能	コマンド	メニュー	ボタン
データブレーク条件の設定	bd	[Break | Data Break...]	–
データブレーク条件の解除	bdc	[Break | Data Break...]	–



[Break]メニューから[Data Break...]を選択すると、[Break setting]ダイアログボックスが現れ、[Data Break]タブの画面に、設定されているデータブレーク条件が表示されます。

データブレーク条件を設定するには、[Set...]ボタンをクリックします。



[Add Data Break]ダイアログボックスが表示されますので、条件を入力して[OK]をクリックします。たとえば、アドレスを001000H～0010FFH、データパターンを1******(*はマスク)、読み出し/書き込み条件を書き込みに設定してプログラムを実行すると、プログラムがアドレス001000H～0010FFHの領域に、ビット7="1"のデータを書き込んだ場合にブレークが発生します。

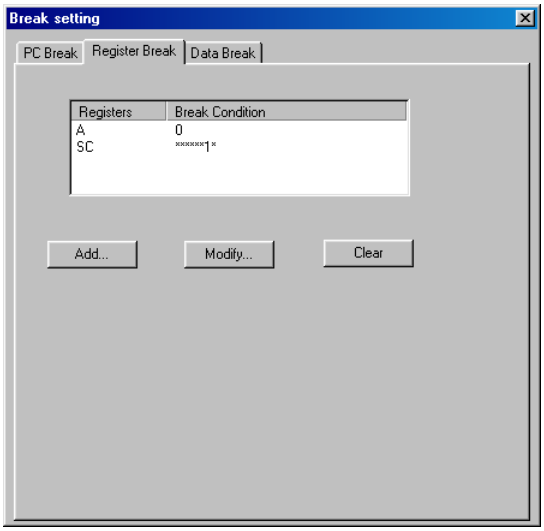
データブレーク条件を解除するには、[Clear]ボタンをクリックします。

(3) レジスタブレーク

レジスタブレークは、PC、SP、IX、IY、A、B、HL、BR、CB、NB、EP、XP、YPレジスタおよびフラグ(I1、I0、U、D、N、V、C、Z)がそれぞれ指定した値になった場合にブレークを発生させる機能です。各レジスタはマスク(ブレーク条件に含めない設定)が可能です。フラグレジスタはビットごとにマスク可能です。ブレークは上記のレジスタがすべて設定条件を満たすように変更された時点で発生します。

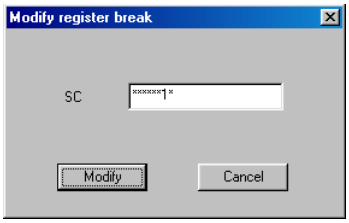
表4.10.5.3　レジスタブレーク設定コマンド

機能	コマンド	メニュー	ボタン
レジスタブレーク条件の設定	br	[Break	Register Break...]	–
レジスタブレーク条件の解除	brc	[Break	Register Break...]	–

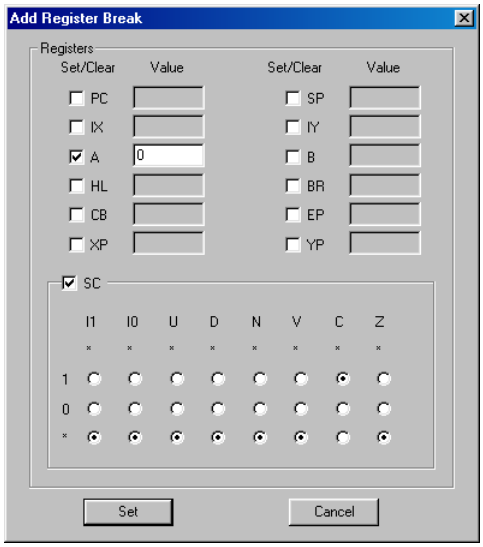


[Break]メニューから[Register Break...]を選択すると、[Break setting]ダイアログボックスが現れ、[Register Break]タブの画面に、設定されているレジスタブレーク条件が表示されます。

設定済みのレジスタブレーク条件を修正するには、Break setting]ダイアログボックスの一覧からレジスタ名を選択し、[Modify...]ボタンをクリックします。[Modify register break]ダイアログボックスが現れ、現在の設定条件を表示しますので、その値を修正して[Modify]ボタンをクリックします。



レジスタブレーク条件を解除するには、[Clear]ボタンをクリックします。



レジスタブレーク条件を新規設定あるいは追加するには、[Break setting]ダイアログボックスの[Add...]ボタンをクリックします。
[Add Register Break]ダイアログボックスが表示されますので、条件を入力して[Set]をクリックします。
たとえば、Aレジスタのデータを0、Cフラグのデータを1、他をすべてマスクに設定してプログラムを実行させると、Aレジスタが0になり、かつCフラグが1になった時点でブレークします。

(4) その他のブレークコマンド

設定されている全ブレーク条件を[Command]ウィンドウに表示するコマンドと、全ブレーク条件を解除するコマンドが用意されています。

表4.10.5.4　その他のブレークコマンド

機能	コマンド	メニュー	ボタン
全ブレーク条件の表示	bl	[Break	Break List]	–
全ブレーク条件の解除	bac	[Break	Break All Clear]	

● [Key Break]ボタンによる強制ブレーク

[Key Break]ボタンは、プログラムが永久ループまたはスタンバイ(HALT、SLEEP)状態から抜け出せない場合など、実行中のプログラムを強制的に終了させます。

 [Key Break]ボタン

● 不当アクセスによるブレーク

プログラム実行中に以下のエラーが発生した場合もブレークします。

未定義プログラム領域へのアクセス

パラメータファイルで設定されたプログラムメモリマップの未定義領域にアクセスした場合にブレークします。
このブレークが発生したときは、次のメッセージを[Command]ウィンドウに表示してコマンド入力待ちとなります。

```
Break by accessing no map program area
>
```

スタック領域外へのアクセス

パラメータファイルで設定されたスタック領域外に対してスタック操作が行われた場合にブレークします。
このブレークが発生したときは、次のメッセージを[Command]ウィンドウに表示してコマンド入力待ちとなります。

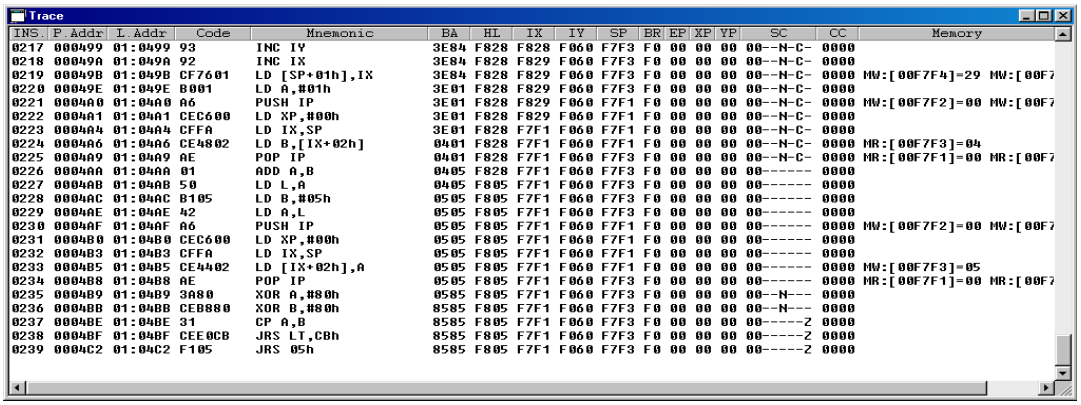
```
Out of stack area
>
```

4.10.6 トレース機能

シミュレータには、プログラムの実行をトレースする機能があります。
トレース機能は、mdコマンド([Option ¦ Mode setting...])の設定により有効になります。初期状態では無効になっています。

●トレースデータバッファとトレース情報

シミュレータはトレースデータバッファを持っています。プログラムを実行すると各実行命令ごとのトレース情報がこのバッファに取り込まれます。トレースデータバッファは8,192命令分の容量があり、トレース情報がこの容量を越えると、古いデータから上書きしていきます。したがって、トレースデータバッファには常に8,192命令以内のトレース情報が記録されています。プログラムの実行によりトレースデータバッファはクリアされ、新しい実行データをトレースします。



各命令の実行でトレースデータバッファに取り込まれるトレース情報は次のとおりです。[Trace]ウィンドウの表示に対応させて示します。

INS: 実行命令番号(0～8,191、10進数)0000が最も古いトレースデータです。
P Addr: PCアドレス(16進物理アドレス)
L Addr: PCアドレス(16進論理アドレス)
Code: 命令コード(16進表示)
Mnemonic: 逆アセンブルコード
BA～CC: CPUレジスタ値(16進数)
Memory: メモリリード/ライト状態(MR:またはMW:)、アクセスしたメモリアドレス(16進数)、データ(16進数)

●トレース情報の表示と検索

採取したトレース情報はプログラムの実行を中断すると[Trace]ウィンドウに表示されます。[Trace]ウィンドウではスクロールによってトレースデータバッファの全データを見ることができます。コマンドにより指定したサイクルから表示させることもできます。表示内容は前記のとおりです。[Trace]ウィンドウが閉じている場合はコマンドで[Command]ウィンドウに表示することもできます。

表4.10.6.1 トレース情報表示コマンド

機能	コマンド	メニュー	ボタン
トレース情報の表示	td	[View ¦ Trace]	–

検索条件を指定し、条件に合ったトレース情報のみを表示させることができます。
検索条件は次の3種類から選択できます。
1. プログラムの実行アドレス
2. データを読み出したアドレス
3. データを書き込んだアドレス

上記条件とアドレスを1ヵ所指定して検索を行います。条件に合ったトレース情報が見つかると、件数を[Command]ウィンドウに表示します。検索データは[Trace]ウィンドウ(閉じている場合は[Command]ウィンドウ)に表示します。

表4.10.6.2　トレース情報検索コマンド

機能	コマンド	メニュー	ボタン
トレース情報の検索	ts	[Trace | Trace Search...]	–

●トレース情報の保存

トレース情報の指定サイクル範囲をファイルに保存することができます。

表4.10.6.3　トレース情報保存コマンド

機能	コマンド	メニュー	ボタン
トレース情報の保存	tf	[Trace | Trace File...]	–

4.10.7 カバレッジ

シミュレータはカバレッジ情報(プログラムを実行したアドレスの情報)を保持し、それを読み出して[Command]ウィンドウに表示することができます。
この機能は、mdコマンド([Option | Mode setting...])の設定により有効になります。初期状態では無効になっています。
次のように実行したアドレス範囲を表示しますので、実行していない領域が分かります。

```
000100 - 000108
000200 - 00020f
       :
```

表4.10.7.1　カバレッジコマンド

機能	コマンド	メニュー	ボタン
カバレッジ情報の表示	cv	–	–
カバレッジ情報のクリア	cvc	–	–

また、カバレッジ機能をONにしておくと、[Source]ウィンドウの実行したアドレスの前に*が表示されます(ソース表示モードを除く)。

4.11 コマンド一覧

表4.11.1 コマンド一覧

分類	コマンド		機能
メモリ操作	**dd**	[<addr1> [<addr2>] [{-Bl-Wl-Ll-Fl-D}]] [<addr1> <@size>] [{-Bl-Wl-Ll-Fl-D}]]	メモリダンプ
	de	[<addr. <data1> [..<data16>]]	データの入力
	df	[<addr1> <addr2> <data>] [<addr1> <@size> <data>]	領域のフィル
	dm	[<addr1> <addr2> <addr3>] [<addr1> <@size> <addr2>]	領域のコピー
	ds	[<addr1> <addr2> <data>] [<addr1> <@size> <data>]	データの検索
レジスタ操作	**rd**		レジスタの表示
	rs	[<reg> <value> [..<reg> <value>]]	レジスタ値の変更 reg={PC|SP|IX|IY|A|B|HL|BR|CB|EP|XP|YP|SC|I1|I0|U|D|N|V|Z|C}
プログラム実行	**g**	[<addr1> [<addr2>]]	カレントPCから連続実行
	gr	[<addr1> [<addr2>]]	CPUリセット後に連続実行
	s	[<step>]	カレントPCからシングルステップ実行
	n	[<step>]	サブルーチン以外をシングルステップ実行
	se		サブルーチン終了
CPUリセット	**rst**		CPUをリセット
ブレーク	**bp**	[<addr1> [..<addr16>]]	ブレークポイントの設定
	bc **bpc**	[<addr1> [..<addr16>]]	ブレークポイントの解除
	bd	[<data> {r|w|*} <addr1> <addr2>]	データブレーク条件の設定
	bdc		データブレーク条件の解除
	br	[<reg> <value> [..<reg> <value>]]	レジスタブレーク条件の設定 reg={PC|SP|IX|IY|A|B|HL|BR|CB|EP|XP|YP|SC}
	brc		レジスタブレーク条件の解除
	bl		全ブレーク条件の表示
	bac		全ブレーク条件の解除
プログラム表示	**u**	[<addr>]	逆アセンブル表示
	sc	[<addr>]	ソース表示
	m	[<addr>]	ミックス表示
シンボル情報	**sy**	[/a]	シンボル一覧の表示
	w	<symbol> [;HlD|Q|B] [/A]	シンボル情報の表示
ファイルロード	**lf**	[<file name>]	プログラム/オプションHEXファイルのロード
	par	[<file name>]	パラメータファイルのロード
トレース	**td**	[<index>]	トレース情報の表示
	ts	[{pc|dr|dw} <addr>]	トレース情報の検索
	tf	[[<index1> [<index2>]] <file name>]	トレース情報の保存
その他	**cv**	[<addr1> [<addr2>]]	カバレッジ情報の表示
	cvc		カバレッジ情報のクリア
	com	[<file name> [<interval>]]	コマンドファイル読み込み/実行
	cmw	[<file name>]	ウェイト付きコマンドファイル読み込み/実行
	rec	[<file name>]	実行コマンドの記録
	log	[<file name>]	ログ出力
	ma		マップ情報の表示
	md	[<option> <num> [..<option> <num>]]	モード設定 option={-tm|-cv|-s|-cm}
	q		デバッガ終了
	?		コマンドusageの表示

4.12 コンポーネントマップファイル(.cmp)

コンポーネントマップファイル(ペリフェラル設定ファイル)は、シミュレータでCPU、キー入力、LCD表示などをシミュレートするために必要な情報を記録したテキストファイルです。各機種の内部周辺回路のデータを記述したファイルが用意されていますので、テキストエディタで必要な情報を追加して使用してください。なお、シミュレータ上で変更したLCDパネルの色などはこのファイルに保存されますので、書き込み禁止(ReadOnly)属性を設定しないでください。

例:

```
[Internal]
CPU=CPU.bmc

LCD=LcdDrv88.bmc
K/P/R port=KPRport.bmc
SVD=SVD88.bmc
Serial=Serial88.bmc (注1)
Adc=Adc88.bmc (注2)

[Settings]
CpuType=88348
ChipMode=MCU
CpuModel=3
OSC1=32.768KHz
OSC3=4.9152MHz

[External]
Ext0=S1D15210.bmc,080000,080002
```

[Internal] 内蔵周辺回路の設定

この部分はユーザが変更することはできません。ただし、すでにS1C63/S1C88シミュレータをインストール済みの環境では、シリアルインタフェースに対応したコンポーネント(注1)をここに追加する必要があります。A/D変換器を使用する場合も同様です(注2)。シリアルインタフェースまたはA/D変換器を搭載していない機種を使用する場合、注1はSerial=NullDev.bmc、注2はAdc=NullDev.bmcとなります。
ROM/RAMサイズの設定はパラメータファイル(.par)にて行います。

[Settings] CPUの設定

これらはすべて必須項目です。下記を参考に変更してください。

```
[Settings]          ←Settingsセクション開始
CpuType=88348       ←ファミリ名
ChipMode=MCU        ←チップタイプ(MCU or MPU)
CpuModel=3          ←CPUモデル(1 or 3)
OSC1=32.768KHz      ←OSC1クロック周波数
OSC3=4.9152MHz      ←OSC3クロック周波数
```

[External] 外部デバイスの設定

ROM/RAMサイズの設定はパラメータファイル(.par)にて行います。
ROM/RAM以外のデバイスを接続する場合は、アドレスデコーダ(CE出力)の設定を参考にしてください。現在は外付けLCDドライバ、バックライト(BkLight)のみサポートしています。使用する場合は以下のように記述してください。

```
[External]          ←Externalセクション開始
Ext0=S1D15210.bmc,080000,080001
Ext1=BkLight.bmc,180004,180004,4L
```

定義形式: Ext<N>=<device>.bmc, <Start_addr>, <End_addr>, <Option>

Ext<N>: Nは0から始まる連番で個々のデバイス番号です。
<device>.bmc: 周辺デバイス定義ファイル名
<Start_addr>: I/Oマップ開始アドレス
<End_addr>: I/Oマップ終了アドレス
<Option>: ペリフェラル固有のオプション

外付けLCDドライバの設定

例に示したS1D15210はアドレスバスの最下位ビットA0に接続され、2バイトの空間にマップされます。例では080000Hでデコードされる CE信号を使用するものとして、080000H（A0=0）と080001H（A0=1）を設定しています。このアドレスに書き込んだ値がS1D15210のD0～D7へ出力され、S1D15210が出力したD0～D7はこのアドレスから読み込むことができます。

BkLightの設定

バックライトを接続したI/Oアドレスを指定します。開始・終了アドレスは同じになります。バックライトを点灯させるビット、およびH/Lのどちらで ONにするかをオプションとして指定します。例では4ビット目がLowのときにバックライトを点灯するように定義しています。

4.13 IOTファイル(.iot)

IOTファイルは、汎用ポートおよびシリアルインタフェースの入出力、A/D変換をシミュレートするための設定および入力データを記述しておくテキストファイルです。これを[I/O Terminal]ウィンドウのメニューで読み込み、入出力シミュレーションを行います。[I/O Terminal]ウィンドウおよび入出力シミュレーションの操作方法については、"4.6.4 [I/O Terminal]ウィンドウ"を参照してください。
入出力シミュレーションを行う場合は、以下の説明に従い、テキストエディタでIOTファイルを作成します。作成したデータは、拡張子を".iot"として保存してください。

ファイルの内容

入力データは[General I/O]、[A/D Converter] のようにセクション単位で記述します。セクションには順序はありませんが、セクション内部の各データは、記述した順に処理されます。使用しない機能のセクションは記述する必要はありません。

例: test.iot

```
[General I/O]
watch P00, P01, P02, P10, P11, P20
5.0 K00=1
5.5 K00=0
8.0 K00=H
8.5 K00=L
[A/D Converter]
AVdd=3.0
AVss=0.2
AVref=2.9
[A/D CH2]
2.40,2.39,2.38,2.37,2.36
2.00, 2.10, 2.20, 2.30
0.5
0.65
0.80
[Serial CH1]
"aBcDeFg"
0x25,0x20,0x41
```

処理される順序

セクション(汎用入出力ポート)

セクション(A/D変換器電圧設定)

処理される順序

セクション(A/D Ch2入力設定)

処理される順序

セクション(シリアルI/F Ch1)

汎用入出力ポート

汎用入出力ポートの設定は、[General I/O]セクションに記述します。

監視するポートの指定

入出力を監視する汎用入出力ポートをwatch文で指定します。

例: `watch P00, P01, P02, P10, P11, P20`

watch文には、入出力状態をチェックするポートの名称(Kxx、Rxx、Pxx)を記述します。ここに記述したポートの入出力がHighからLowまたはLowからHighに変化すると、[I/O Terminal]ウィンドウにログが表示されます。ポートの監視を行わない場合は、記述する必要はありません。

入力の指定

汎用ポートへ入力を行う場合は、入力を行う時刻(秒単位)とポート名、およびポートへの入力レベルを1行単位で記述します。入力シミュレーションは記述した順に上から行われます。

例:
```
5.0 K00=1
8.5 K00=0
+0.7 P02=H, P10=L
```

各行の先頭は、入力を行う時刻(秒単位)を表します。小数点を用いて1msの桁まで設定することができます。基本的に、値はリセットからの経過時間です。なお、先頭に"+"を付けることにより、前回の入力からの経過時間を指定することができます。
入力内容は、"ポート名=入力レベル"の形式で記述します。Highレベル入力はHまたは1、Lowレベル入力はLまたは0で指定します。
上記例の場合、リセット実行から5.0秒後にK00ポートにHigh、8.5秒後にK00ポートにLow、9.2秒後にP02ポートにHigh、同時にP10ポートにLowが入力されます。
入力ポート(Kポート)割り込みが許可されている場合、入力のタイミングで割り込みが発生します。

A/D変換器

A/D変換器の設定は、[A/D Converter]セクションと[A/D CHn]セクションに記述します。

A/D変換器の電圧設定

[A/D Converter]セクションには、変換器の電圧設定を記述します。

例:
```
[A/D Converter]
AVdd=3.0
AVss=0.2
AVref=2.9
```

AVdd、AVss、AVrefの各行は、それぞれA/D変換器の+電源電圧、-電源電圧、基準電圧を表します。電圧値はボルト単位で記述します。値は小数点を用いて1mVの桁まで設定することができます。A/D変換器を使用する場合、この3行を含む[A/D Converter]セクションは必ず記述しなければなりません。

アナログ入力電圧の指定

[A/D CHn]セクションには、A/D変換器への入力電圧を記述します。指定はチャンネル単位に行います。未使用のチャンネルは記述する必要はありません。

例:
```
[A/D CH4]
2.40,2.39,2.38,2.37,2.36
2.00, 2.10, 2.20, 2.30
[A/D CH5]
0.5
0.65
0.80
```

入力電圧はボルト単位で記述します。値は小数点を用いて1mVの桁まで設定することができます。
記述したデータはターゲットプログラムがA/D変換を実行するごとに、上から順に読み込まれます。1行に記述する場合はカンマ(,)で区切ります。この場合は左から順に読み込まれます。
割り込みが許可されている場合、入力を開始してから一定時間(サンプリング時間 + A/D 変換時間)経過後に割り込みが発生します。

データを指定しなかったチャンネルのA/D変換を行った場合は、レベル0の電圧(AVssと等価)が入力されているものとして処理されます。指定したデータ数を超えるA/D変換を行った場合は、最後に指定したデータが繰り返し使用されます。
また、AVssよりも小さな値を指定した場合、レベル0(AVssと等価)として読み出されます。AVrefよりも大きな値を指定した場合、最大レベル(AVrefと等価)として読み出されます。

シリアルインタフェース

シリアルインタフェースの設定は[Serial CHn]セクションに記述します。設定はチャンネル単位で行います。1チャンネルのみを内蔵している機種は"CH1"(チャンネル1)を指定します。未使用のチャンネルは記述する必要はありません。

例:
```
[Serial CH1]
"aBcDeFg"
0x25,0x20,0x41,FER
```

入力データは16進数表記または文字列表記(" "で囲む)のどちらかで指定します。1行に混在させることはできません。記述したデータはシリアルインタフェースによる受信を行うごとに、上から順に読み込まれます。複数の16進データを1行に記述する場合はカンマ(,)で区切ります。この場合は左から順に読み込まれます。
割り込みが許可されている場合、入力サンプリング終了のタイミングで割り込みが発生します。
データを指定しなかったチャンネルで受信を行った場合、割り込みは発生しません。

サンプリングに必要な時間
調歩同期式: I/Oレジスタで選択したクロック源の周期×ビット数(+パリティビット)
クロック同期式(マスタ): I/Oレジスタで選択したクロック源の周期×ビット数
クロック同期式(スレーブ): ユーザ定義の転送レート(bps)

クロック同期式(スレーブ)の場合は、転送レート(bps)を以下のように指定します。データよりも前に指定してください。この設定はクロック同期式(スレーブ)の場合のみ有効です。

例:
```
[Serial CH1]
bps=2400
"aBcDeFg"
0x25,0x20,0x41,FER
```

16進データの代わりに、以下のデータを指定することができます。
FER: フレーミングエラー
PER: パリティエラー
OER: オーバーランエラー

これらのデータは、シリアルインタフェースがエラー割り込みをサポートしている場合のみ有効です。サポートしていない機種の場合は指定しないでください。
これらのデータを読み込んだ際、割り込みが許可されているとエラー割り込みが発生します。

シリアルインタフェースの出力に関しては、特に何も指定する必要はありません。各チャンネルへの出力が行われると、ログが[I/O Terminal]ウィンドウに表示されます。

4.14 シミュレータプロジェクトファイル(.spj)

シミュレータプロジェクトファイルは、使用するパラメータファイル、LCDパネル定義ファイル、コンポーネントマップファイルおよびポート設定ファイルを記述したテキストファイルです。テキストエディタで作成してください。このファイルにより、上記のファイルの選択を一度に行うことができます。
シミュレータの起動時、またはparコマンド([File¦Load Parameter File])実行時に表示されるダイアログボックスでパラメータファイルかシミュレータプロジェクトファイルのどちらか一方を選択します。パラメータファイルを選択する場合は、シミュレータプロジェクトファイルを用意しておく必要はありません。
ファイルの内容は次のとおりです。
例:

```
[Setting]
PAR=88348.par       ←パラメータファイル
LCD=88348dmt.lcd    ←LCDパネル定義ファイル
CMP=88348dmt.cmp    ←コンポーネントマップファイル
PRT=88348dmt.prt    ←ポート設定ファイル
```

4.15 制限事項

- 本シミュレータの対応機種につきましてはReadme_J.txtの、"サポート機種および周辺機器"を参照してください。
- 外部デバイスは、ROM、RAM、以下のLCDドライバ、モノクロLCDパネル、バックライト、キー、キーマトリクスのみをサポートしています。
  S1D15210, S1D15600, S1D15601, S1D15602, S1D15605, S1D15606, S1D15607, S1D15608
- 外部ROM、RAM、LCDドライバは、外部バスに接続する方法のみサポートします。外付けLCDドライバとのシリアル接続や汎用ポートを使用した外部ROM、RAMのアクセスはサポートしていません。
- 本シミュレータはインストラクションレベルでシミュレーションを行っています。よってインストラクションサイクルよりも短いサイクルのシミュレーションを行うことはできません。
- インストラクションサイクルを元にタイマをシミュレーションしていますので、実際のハードウェアとはタイミングが同じではありません。
- 外部周辺デバイスアクセス時のウェイト挿入は正確にシミュレートされません。
- 以下の機能はサポートしていません。
  1. ブザー出力
  2. TOUT/FOUT出力
  3. D/Aコンバータ、アナログコンパレータ
  4. プログラマブルタイマのイベントカウンタモード
- サウンドジェネレータをシミュレートするには、PCM音源の再生が可能なサウンドカードが必要です。
- PC上で複数のシミュレータを同時に起動してシミュレーションを行うことはできません。
- Kポート同時Low入力によるリセットを行う際、割り当てた4つのキーを同時に押し下げることによるリセットは実行できません。1+3または2+2の組み合わせでわずかにタイミングをずらして押し下げを行ってください。またICEとは異なり、2秒間の検定時間を待たずにリセットが行われます。

最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_sim88_J.txtを参照してください。
また最新バージョンのサポート機種については、S5U1C88000QのReadme_J.txtを参照してください。

4.16 シミュレータメッセージ

● ステータスメッセージ一覧

メッセージ	内容	該当コマンド
Break by accessing no map program area	未定義プログラム領域のアクセスによりブレーク	
Break by data break	データブレーク発生	
Break by PC break	PCブレーク発生	
Break by register break	レジスタブレーク発生	
Key Break	[Key Break]ボタンによりブレーク	
No coverage data	カバレッジ情報がありません	cv
No matched trace data	指定条件に合うトレース情報がありません	ts
No trace data	トレース情報がありません	td, tf, ts
Out of stack area	スタック領域外へのスタック操作によりブレーク	
Set to log off mode	ログ出力をOFF	log
Set to log on mode	ログ出力をON	log
Set to record off mode	レコード出力をOFF	rec
Set to record on mode	レコード出力をON	rec
Simulator initialization error!	シミュレータの初期化に失敗しました	
Symbol file does not exist	プログラムファイルのディレクトリに対応するシンボルファイルがありません	lf
Symbol file is loaded	プログラムファイルと共に対応するシンボルファイルがロードされました	lf

● エラーメッセージ一覧

メッセージ	内容	該当コマンド
Address out of range, use 0 - 0x7FFFFF	プログラムメモリ領域外のアドレスが指定されました	g, bp, bc(bpc), u, ts, cv
Address out of range, use 0 - 0xFFFFFF	データメモリ領域外のアドレスが指定されました	dd, de, df, dm, bd, ts
Cannot open file	ファイルがオープンできません	lf, par, com, cmw
Data out of range, use 0 - 0xFF	指定の数値はデータの有効範囲外です	de, df
End address < start address	開始アドレスより小さい終了アドレスが指定されました	dd, df, dm, cv
End index < start index	開始サイクルより小さい終了サイクルが指定されました	tf
Error file type (extension should be CMD)	指定のファイル拡張子はコマンドファイルとして無効です	com, cmw
Error file type (extension should be PAR)	指定のファイル拡張子はパラメータファイルとして無効です	par
Incorrect number of parameters	コマンドのパラメータ数が不正です	
Incorrect r/w option, use r/w/*	無効なR/Wオプションが指定されました	bd
Incorrect register name, use PC/SP/SP/IX/IY/A/B/HL/BR/CB/EP/XP/YP/SC	レジスタブレーク条件に無効なレジスタ名が指定されました	br
Incorrect register name, use PC/SP/SP/IX/IY/A/B/HL/BR/CB/EP/XP/YP/SC/I1/I0/U/D/N/V/Z/C	無効なレジスタ名が指定されました	rs
Index out of range, use 0 - 8191	指定のトレースサイクル番号は有効範囲外です	td
Input address does not exist	未設定のブレークポイントアドレスが指定されました	bp, bc (bpc)
Invalid command	無効なコマンドが入力されました	
Invalid data pattern	入力データパターンが不正です	bd, br
Invalid display unit, use -B/-W/-L/-F/-D	表示単位の指定が無効です	dd
Invalid file name	指定のファイル拡張子はプログラムファイルまたはファンクションオプションファイルとして無効です	lf
Invalid option, use -tm/-cv/-s/-cm	モード設定用オプションが無効です	md
Invalid value	入力した値が不正です	
Maximum nesting level(5) is exceeded, cannot open file	コマンドファイルのネストレベルが制限を越えました	com. cmw
No symbol information	シンボル情報がありません (シンボルファイルがロードされていません)	sy
Number of steps out of range, use 0 - 65535	指定ステップ数は制限を越えています	s, n
This command is not supported in current mode	トレース/カバレッジコマンドはトレースOFF/カバレッジOFF時は無効です	td, ts, tf, cv, cvc

● ワーニングメッセージ一覧

メッセージ	内容	該当コマンド
64 break addresses are already set	64ヵ所を越えるブレークポイントが指定されました	bp
Break address already exists	指定のアドレスはすでにブレークポイントに設定されています	bp
Identical break address input	コマンドラインに同じアドレスが2回以上指定されています	bp

5 LCDパネルカスタマイズユーティリティ

5.1 概要

LCDパネルカスタマイズユーティリティ(LcdUtil.exe、以下LcdUtilと記述)はシミュレータ(sim63.exe、sim88.exe)でモノクロLCDパネルの表示をシミュレートするための、パネルレイアウトおよびCOM/SEGポートの割り付けデータを作成します。アイコンなどはビットマップファイル(.bmp)から読み込むため汎用ペイントソフトで作成可能で、実際の製品と同様の画面をシミュレートすることができます。本ツールは機種に依存しない共通のツールで、機種で異なるLCDドライバの設定は機種情報定義ファイルにより行います。なお、このファイルによりS1Cチップの内蔵LCDドライバのほかセイコーエプソン製外付けLCDドライバにも対応します。

5.2 入出力ファイル

図5.2.1にLcdUtilの入出力ファイルを示します。

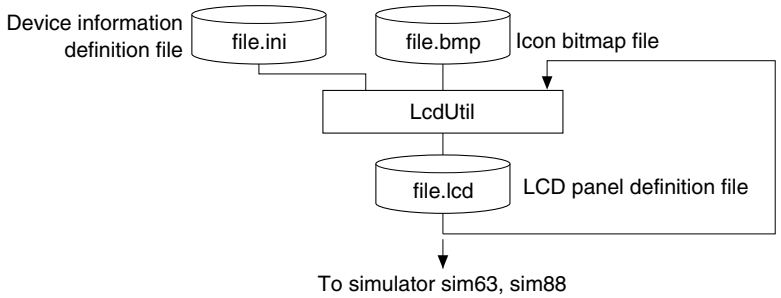


図5.2.1 LcdUtilの入出力ファイル

● 機種情報定義ファイル(file_name.ini)

S1C63xxx/S1C88xxxのLCD端子などの情報が記録されています。必ずセイコーエプソンが提供するファイルを使用してください。このファイルはファイル名で示される機種にのみ有効です。ファイルの内容を修正したり、他の機種で使用しないでください。

● ビットマップファイル(file_name.bmp)

アイコンの図形をこの形式(白黒ビットマップ)のファイルで読み込みます。複数のパーツを個々に読み込んで、LcdUtil上でレイアウトを編集することができます。

● LCDパネル定義ファイル(file_name.lcd)

パネルレイアウトのビットマップと共にCOM/SEG端子の割り付け情報が記録されます。このファイルをシミュレータにロードしてLCD表示のシミュレーションを行います。

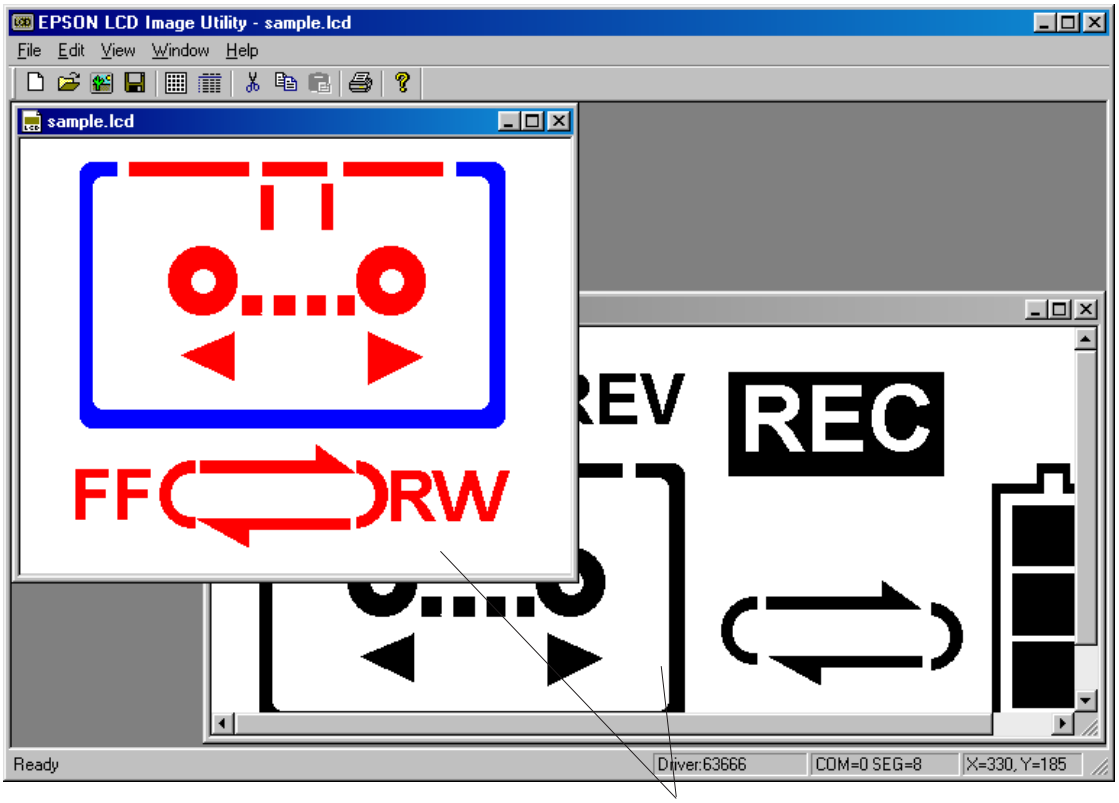
5.3 起動と終了

LcdUtil.exe

このアイコンをダブルクリックすると、LcdUtilが起動します。

終了するには、[File]メニューから[Exit]を選択します。

5.4 ウィンドウ



パネル編集ウィンドウ

● パネル編集ウィンドウ

ビットマップファイル(.bmp)またはLCDパネル定義ファイル(.lcd)を開くと、このウィンドウに表示されます。このウィンドウ上でパネルのレイアウト、COM/SEGの割り付けを行います。
同時に複数のウィンドウを開くことができます。2つのウィンドウ間でのドラッグ&ドロップにも対応しています。

● ウィンドウ自体の基本操作

クローズとアクティブ化

ウィンドウを閉じるには、ウィンドウの[閉じる]ボタンをクリックしてください。
開いているウィンドウは[Window]メニューにリストされます。そこからウィンドウ名を選択することで、そのウィンドウがアクティブになります。ウィンドウ上をクリックすることでも同様です。また、[Ctrl]+[Tab]のキー操作によってもアクティブウィンドウの切り替えが行えます。

サイズの変更と移動

それぞれのウィンドウサイズは、ウィンドウの境界をドラッグすることによって任意の大きさに変更できます。[最小化]ボタン、[最大化]ボタン等も一般のWindowsアプリケーションと同様です。各ウィンドウはタイトルバーをドラッグすることによって、表示位置を変更できます。ただし、サイズ変更、移動ともに、アプリケーションウィンドウの範囲内に限られます。表示する画像がウィンドウより大きい場合はスクロールバーが表示されます。

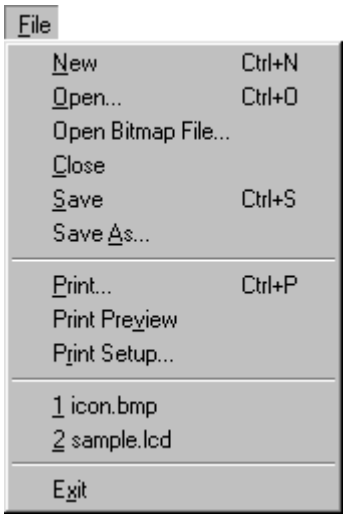
その他

開いているウィンドウは、[Window]メニューの[Cascade]または[Tile]を選択することで整列させることができます。

5.5 メニューとツールバー

5.5.1 メニュー

[File]メニュー



[New] ([Ctrl]+[N])
新しいパネル編集ウィンドウを開きます。

[Open...] ([Ctrl]+[O])
LCDパネル定義ファイル(.lcd)を開きます。

[Open Bitmap File...]
ビットマップファイル(.bmp)を開きます。

[Save] ([Ctrl]+[S])
アクティブなパネル編集ウィンドウの内容をLCDパネル定義ファイル(.lcd)に保存(上書き)します。

[Save As...]
アクティブなパネル編集ウィンドウの内容を別の名称でLCDパネル定義ファイル(.lcd)に保存します。

[Print...] ([Ctrl]+[P])
アクティブなパネル編集ウィンドウのビットマップを印刷します。

[Print Preview]
アクティブなパネル編集ウィンドウの印刷イメージを表示します。

[Print Setup...]
用紙やプリンタを選択するダイアログボックスを表示します。

ファイルリスト
ここにリストされているファイル名は最近開いたファイルです。ここからの選択でも、そのファイルを開くことができます。

[Exit]
LcdUtilを終了します。

[Edit]メニュー

[Cut] ([Ctrl]+[X])
パネル編集ウィンドウ内で選択されているパーツを、クリップボードにカットします。

[Copy] ([Ctrl]+[C])
パネル編集ウィンドウ内で選択されているパーツを、クリップボードにコピーします。

[Paste] ([Ctrl]+[V])
クリップボードにコピーされているパーツをパネル編集ウィンドウの左上端にペーストします。

[Insert dot matrix] ([Ctrl]+[M])
パネル編集ウィンドウにドットマトリクスを挿入します。ダイアログボックスでサイズなどを指定できます。

[Icon List] ([Ctrl]+[I])
アクティブなパネル編集ウィンドウにあるアイコンの一覧を表示します。ここで表示されるダイアログボックスでCOM/SEGの割り付けも行えます。

[Resize LCD]
LCDパネルのサイズを設定します。新規にパネル編集ウィンドウを開いた場合のデフォルトサイズは640×480です。

[Group Icon]
複数のアイコンを1つにグループ化します。

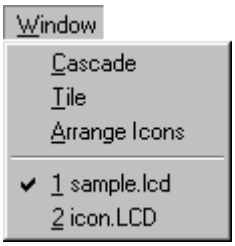
[Release Group]
グループを解除し、個々のアイコンに戻します。

[View]メニュー

[Toolbar]
ツールバーの表示/非表示を切り替えます。

[Status Bar]
ステータスバーの表示/非表示を切り替えます。

[Window]メニュー



[Cascade]
開いているパネル編集ウィンドウを斜めに重ねて表示します。

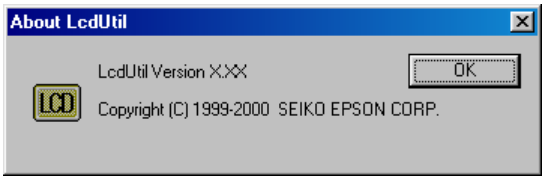
[Tile]
開いているパネル編集ウィンドウを上下に並べて表示します。

[Arrange Icons]
最小化されたパネル編集ウィンドウアイコンをアプリケーションウィンドウ領域下部に整列させます。

ウィンドウリスト
現在開いているパネル編集ウィンドウ名がリストされます。ここでウィンドウ名を選択すると、対応するパネル編集ウィンドウがアクティブになります。

[Help]メニュー

[About LcdUtil...]
LcdUtilのバージョン情報を表示します。



5.5.2 ツールバーボタン

[New]ボタン
新しいパネル編集ウィンドウを開きます。

[Open]ボタン
LCDパネル定義ファイル(.lcd)を開きます。

[Bitmap]ボタン
ビットマップファイル(.bmp)を開きます。

[Save]ボタン
アクティブなパネル編集ウィンドウの内容をLCDパネル定義ファイル(.lcd)に保存(上書き)します。

[Dot Matrix]ボタン
パネル編集ウィンドウにドットマトリクスを挿入します。ダイアログボックスでサイズなどを指定できます。

[Icon List]ボタン
アクティブなパネル編集ウィンドウにあるアイコンの一覧を表示します。ここで表示されるダイアログボックスでCOM/SEGの割り付けも行えます。

[Cut]ボタン
パネル編集ウィンドウ内で選択されているパーツを、クリップボードにカットします。

[Copy]ボタン
パネル編集ウィンドウ内で選択されているパーツを、クリップボードにコピーします。

[Paste]ボタン
クリップボードにコピーされているパーツをパネル編集ウィンドウの左上端にペーストします。

[Print]ボタン
アクティブなパネル編集ウィンドウのビットマップを印刷します。

[About]ボタン
LcdUtilのバージョン情報を表示します。

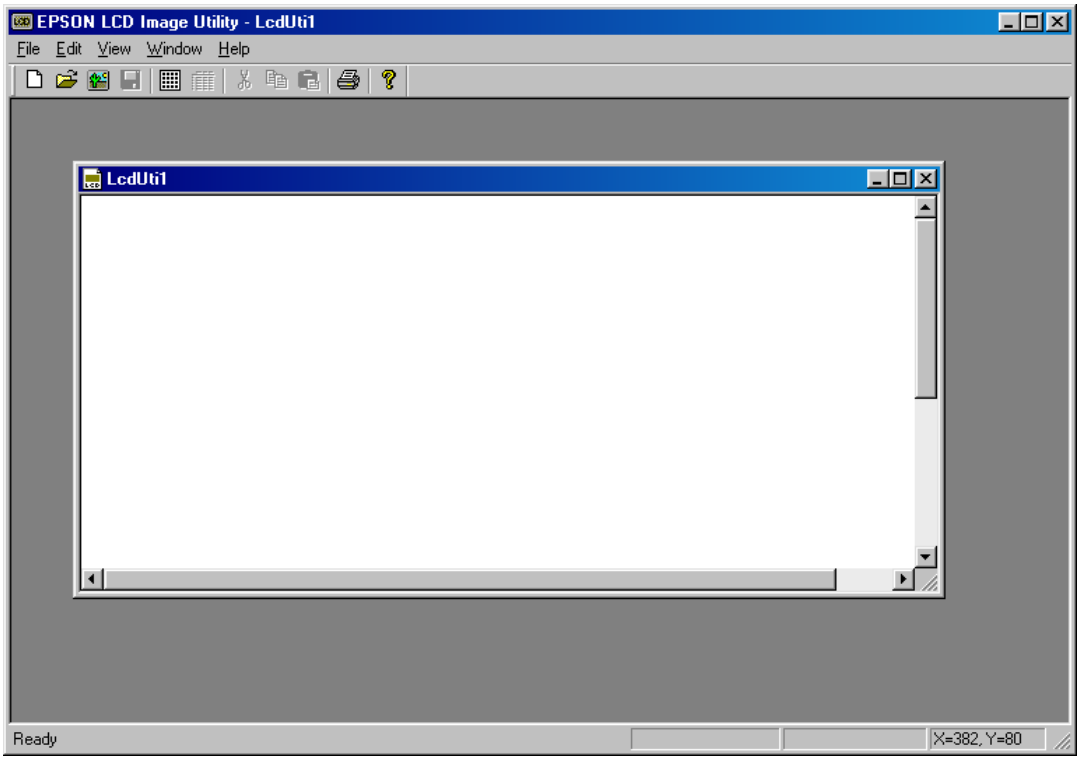
5.6 LCDパネルデータの作成

パネル編集ウィンドウにはアイコンとドットマトリクスブロックを実際のパネルのようにレイアウトすることができ、COM/SEGの割り付けも行えます。以下、その方法を説明します。

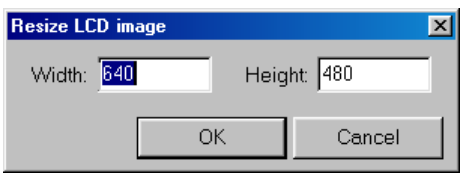
5.6.1 新規パネルの作成とパネルサイズの設定

空白のパネル編集ウィンドウにドットマトリクスやアイコンを配置してLCDパネル定義ファイルを作成する場合は、最初に以下の手順で新規パネルを設定します。

1) [File]メニューから[New]を選択(または[New]ボタンをクリック)します。
空白のパネル編集ウィンドウが開きます。



2) [Edit]メニューから[Resize LCD]を選択します。
[Resize LCD image]ダイアログボックスが表示されます。



ここにパネルのサイズを入力し、[OK]をクリックします。
この数値はシミュレータでシミュレートする際のLCDパネルのサイズでコンピュータ画面上のドット数で指定します。新規作成時のデフォルトサイズは640×480ドットです。また、配置するアイコンのビットマップやドットマトリクスのサイズも考慮する必要があります。パネルの最大サイズは1280×1024ドットです。
なお、サイズの変更は、アイコンやドットマトリクスの配置後でも行えます。

以上でパネルのベースができます。ここに、ドットマトリクスや別途作成したアイコンのビットマップを配置します。

アイコンのビットマップを読み込み、それをベースにLCDパネル定義ファイルを作成する場合は、ここで説明した新規作成の操作は必要ありません。

5.6.2 アイコンの作成

ここでは、アイコンのポート割り付けについて説明します。

● ビットマップデータの作成

アイコンのビットマップはスキャナ入力や一般のペイントソフトで作成します。
この際、以下の点に注意してください。

色数とファイル形式

背景を白、アイコンを黒で作成し、白黒ビットマップ形式(.bmp)で保存してください。
65535色以下のビットマップはLcdUtilで読み込み可能ですが、読み込み時に白黒の2値データに変換されます。

サイズ

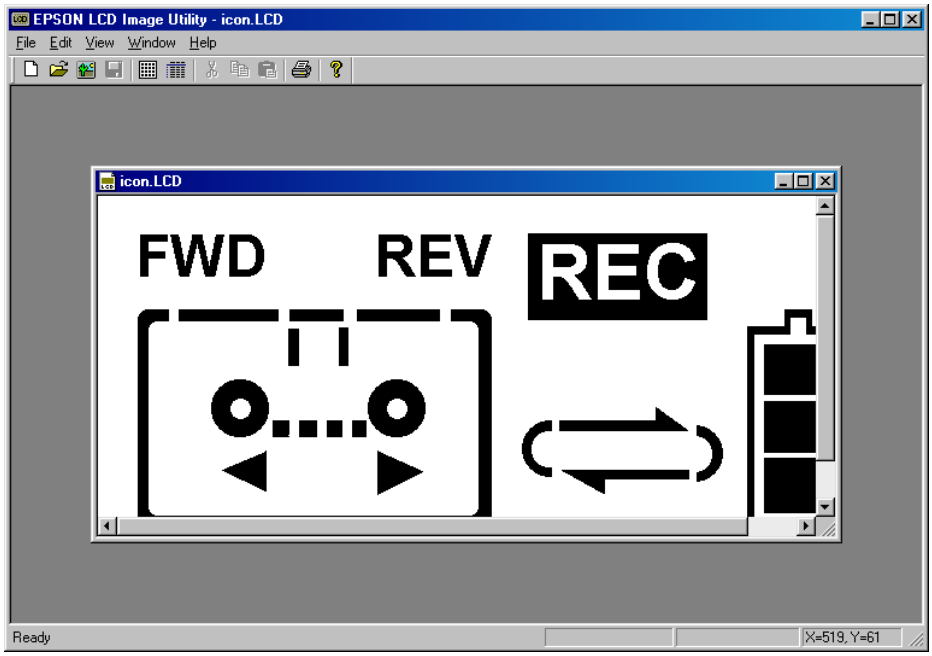
LcdUtilではシミュレータでシミュレートする際のLCDパネルのサイズをコンピュータ画面上のドット数で指定できます([Edit]メニューの[Resize LCD])。ビットマップデータの最大サイズは1280×1024ドットです。
個々のアイコンはシミュレートする際のパネルサイズに合わせて相対的に大きさを決めて作成します。LcdUtil上でアイコンのサイズを調整することはできません。
ドットマトリクスも混在するパネルをシミュレートする場合、ドットのサイズをコンピュータ画面上の何ドットにするか指定できますので、マトリクスのサイズを考慮してパネルサイズやアイコンのサイズを決めてください。なお、シミュレータ上でも25%～200%まで数段階の拡大縮小が行えます。
プログラムの動作確認のみを目的とする場合は特に厳密な形と大きさで作成しておく必要はありませんが、LCDパネルのデザイン評価も行えるように正確な表示シミュレーションが可能です。

ファイル数

アイコンすべてを1つのファイルに作成することも、いくつかに分けて作成することもできます。
ただし、LcdUtilには簡易的な編集機能しかありませんので、できるだけビットマップ作成時に1つのファイルとしてレイアウトも決定しておくことを推奨します。

● ビットマップデータの読み込み

LcdUtilを起動し、[File]メニューから[Open Bitmap File...]を選択(または[Bitmap]ボタンをクリック)します。標準の[開く]ダイアログボックスが表示されますので、作成したビットマップファイルを選択してロードします。
パネル編集ウィンドウが開き、読み込んだビットマップを表示します。



● アイコンのレイアウト編集

読み込んだビットマップは黒が連続するそれぞれの部分がパーツとして切り出されアイコンと見なされます。このパーツに対して以下の操作が可能です。

選択

パーツをクリックすると、色が青に変わり選択されたことを示します。
以下の操作やCOM/SEGポート割り付けは、この選択状態で行います。

移動

選択したパーツをドラッグします。
他のパネル編集ウィンドウに移動することもできます。

コピー

メニューまたはツールバーボタンでコピー&ペーストします。

削除

[delete]キーで削除できます。

グループ化

[Ctrl]キーを押しながらクリックすることで複数のパーツを選択できます。
その状態で[Edit]メニューから[Group Icon]を選択すると、それらのパーツがグループ化され1つのアイコンとして扱えます。
グループ化されたアイコンは[Edit]メニューから[Release Group]を選択することによって元の複数のパーツに戻ります。

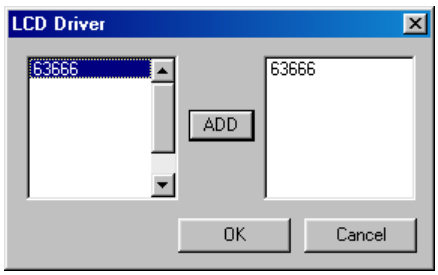
注: LcdUtilの編集機能は簡易的なものです。できるだけビットマップ作成時に1つのファイルとしてレイアウトも決定しておくことを推奨します。

● ポート割り付け

パネル編集ウィンドウ上でCOM/SEGポート割り付けを行うことができます。
その手順を以下に示します。

1）ポート割り付けを行うアイコンをダブルクリックします。

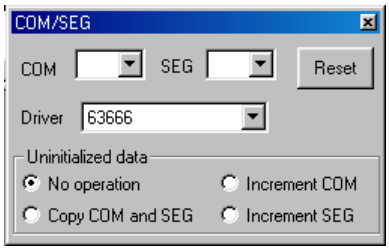
2）[LCD Driver]ダイアログボックスが表示されます。



開発する機種をリストから選択し、[ADD]ボタンをクリックします。
S1C88 Familyでリストにある外部LCDドライバも使用する場合は、同じように選択/追加してください。
選択終了後、[OK]ボタンをクリックします。中止する場合は[Cancel]をクリックしてください。

* このダイアログボックスは、初めてポート割り付けを行う場合に表示され、一度ドライバを選択すると、その後は表示されません。
ポート割り付け後にドライバを変更することはできませんので、選択には注意してください。

3）[COM/SEG]ダイアログボックスが表示されます。



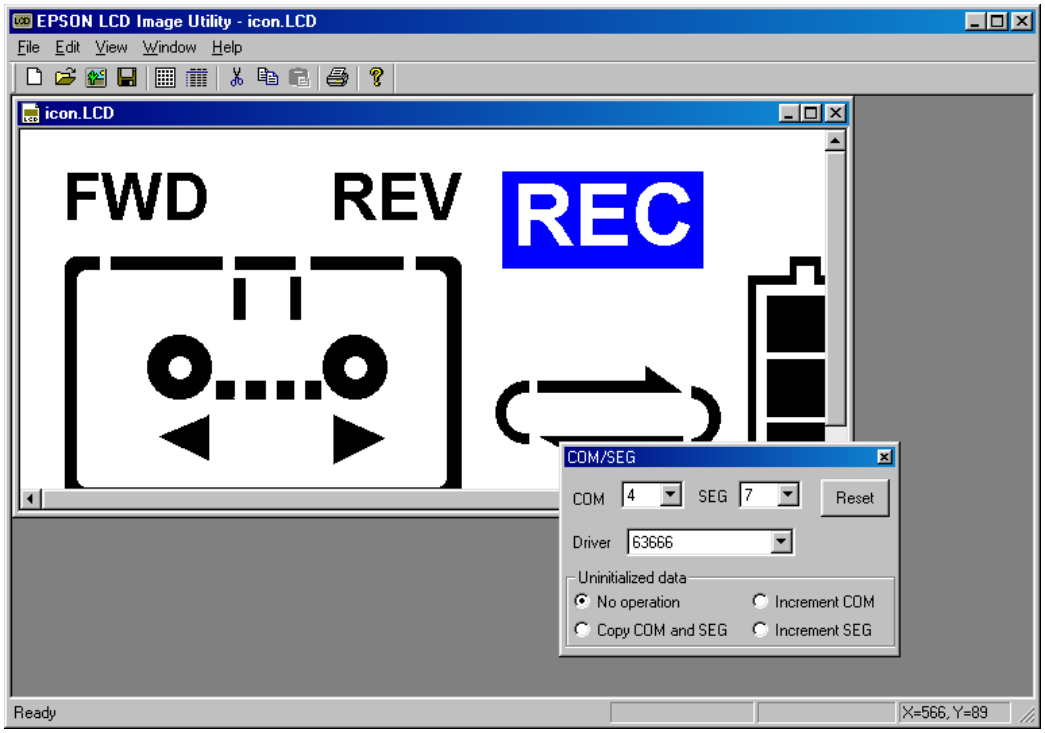
COMポートとSEGポートの番号を、それぞれのプルダウンリストから選択します。
[Reset]ボタンをクリックするとCOM/SEGの選択がクリアされ、[Driver]の選択もデフォルトのドライバに戻ります。

選択後、パネル編集ウィンドウ上をクリックするか、このダイアログを閉じて次のアイコンの割り付けを行います。

[Uninitialized data]のラジオボタンで、以下のように選択が行えます。

No operation	未定義のアイコンを選択するごとにCOMとSEGの番号はクリアされます。
Copy COM and SEG	未定義のアイコンを選択してもCOMとSEGの番号はクリアされずに直前の設定が残ります。
Increment COM	未定義のアイコンを選択するとSEGの番号はクリアされずに直前の設定が残ります。COM番号はインクリメント(+1)されます。
Increment SEG	未定義のアイコンを選択するとCOMの番号はクリアされずに直前の設定が残ります。SEG番号はインクリメント(+1)されます。

ポートが割り付けられたアイコンは赤で表示されます。また、その上にマウスポインタを重ねると、ステータスバーに割り付けられたポートとドライバの情報を表示します。また、マウスカーソルの位置にもポート情報を表示します。



複数のパーツで1つのアイコンを構成する場合は、それぞれのパーツに同じCOM/SEGポートを割り付けます。あるいは、あらかじめグループ化しておくと1回の選択で済みます。

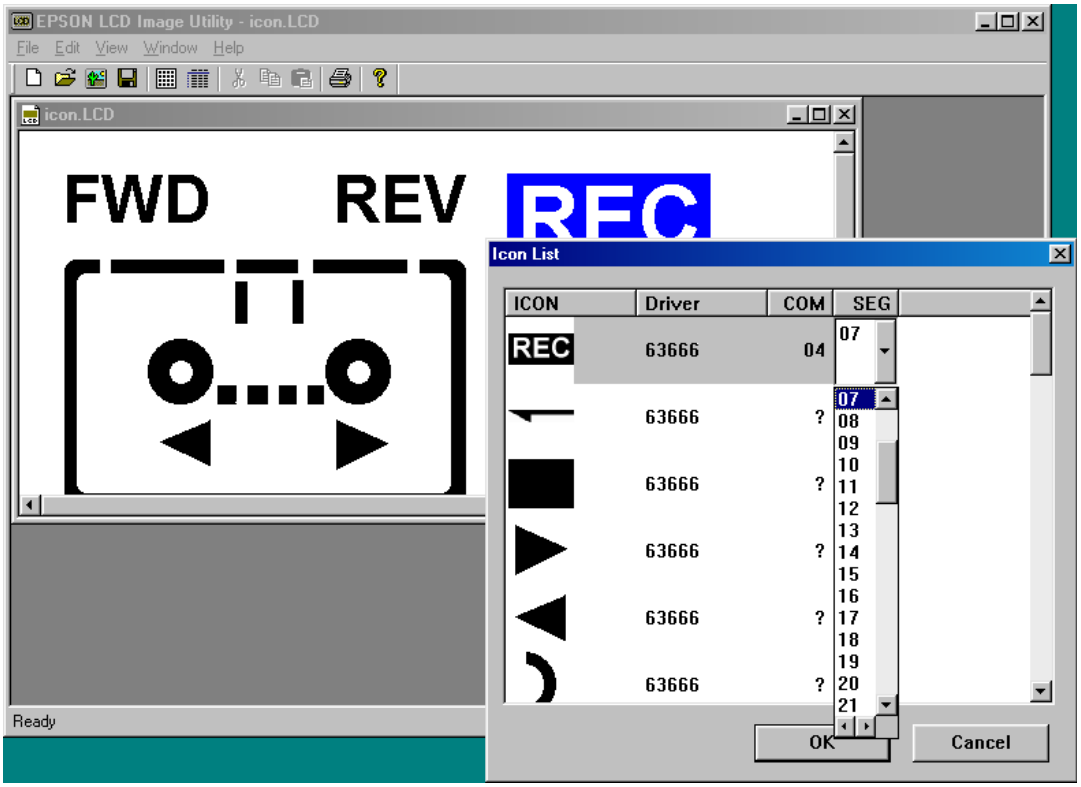
ポートが割り付けられているアイコンをダブルクリックした場合もこのダイアログボックスが表示されますので、後から割り付けを修正することは可能です。

注: ポート割り付けの済んだアイコンをコピー&ペースト、あるいは他のパネル編集ウィンドウに移動した場合、ポート割り付け情報はクリアされます。したがって、他のパネル編集ウィンドウでレイアウト編集をしている場合は、アイコンをそちらに移動してからポート割り付けを行ってください。
また、異なるポートを割り付けたパーツをグループ化すると、グループ化のために最後に選択したパーツの割り付けが選択したすべてのパーツに適用されます。この設定はその後グループを解除しても元には戻りません。

4) [File]メニューから[Save]または[Save As...]を選択し、設定内容をLCDパネル定義ファイルとして保存してください。

● アイコンリストの表示

[Edit]メニューから[Icon List]を選択(または[Icon List]ボタンをクリック)すると、現在アクティブなパネル編集ウィンドウ内のアイコンの一覧が表示されます。



リスト内のアイコンをクリックして選択すると、パネル編集ウィンドウ内のアイコンも青く表示され、どのアイコンに対応しているか分かります。
また、ドライバ、COM/SEG番号の部分をクリックするとプルダウンリストが現れ、設定内容を変更することもできます。

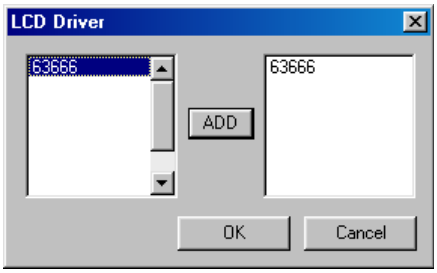
5.6.3 ドットマトリクスの作成

ここでは、ドットマトリクスの作成とポート割り付けについて説明します。

1）5.6.1項に示した手順で新規パネルを作成します。
あるいは、アイコンを含むビットマップファイルかLCDパネル定義ファイルを開いてください。
パネルのサイズはドットマトリクスのサイズを計算し、それより大きく設定してください。

2）[Edit]メニューから[Insert dot matrix]を選択（または[Dot Matrix]ボタンをクリック）します。

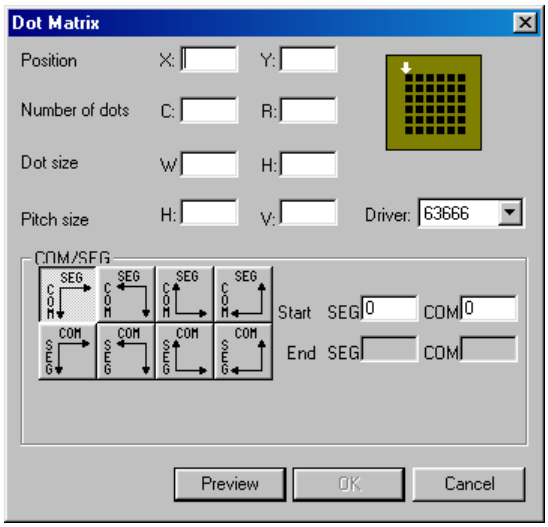
3）[LCD Driver]ダイアログボックスが表示されます。



開発する機種をリストから選択し、[ADD]ボタンをクリックします。
さらに、リストにある外部LCDドライバも使用する場合は、同じように選択/追加してください。
選択終了後、[OK]ボタンをクリックします。中止する場合は[Cancel]をクリックしてください。

* このダイアログボックスは、LCDドライバがまだ選択されていない場合にのみ表示されます。すでにアイコンにポートを割り付けてある場合は表示されません。

4）[Dot Matrix]ダイアログボックスが表示されます。



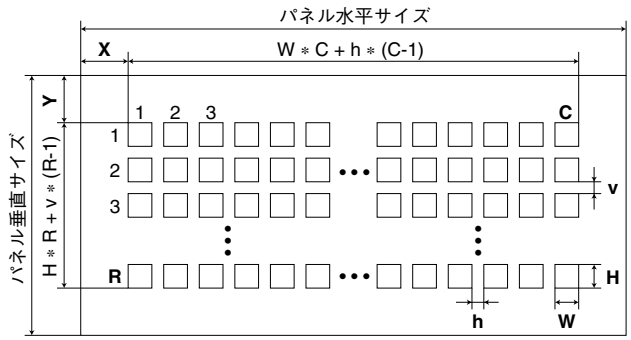


図5.6.3.1　ドットマトリクスの設定

ダイアログ内で以下の設定を行ってください。

Position
パネルの左上端からドットマトリクスの左上端までの距離をコンピュータ画面のドット数で指定します。(図のXとY)

Number of dots
ドットマトリクスの水平方向、垂直方向のドット数を指定します。(図のCとR)

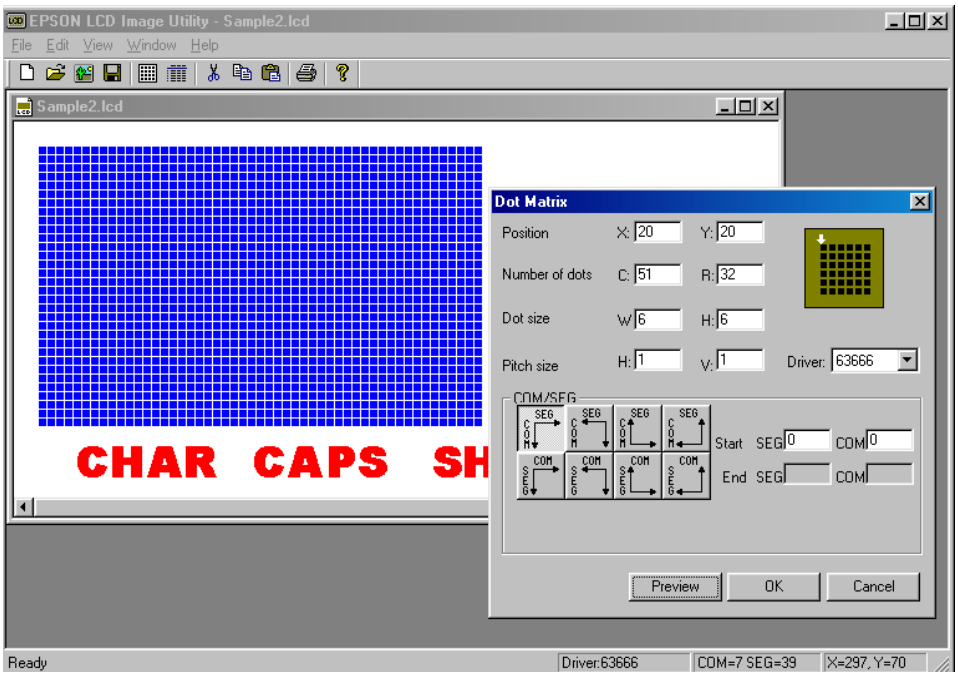
Dot size
1つのドットをコンピュータ画面上の何ドットで表示するか指定します。(図のWとH)

Pitch size
各ドットの表示間隔をコンピュータ画面のドット数で指定します。(図のhとv)

Driver
使用するLCDドライバを選択します。複数のLCDドライバで1つのドットマトリクスを駆動する場合は、ドライバごとにドットマトリクスを作成してください。

COM/SEG
8個のボタンの1つで割り付けの開始点を選択し、[Start]テキストボックスにそのドットに割り付けるCOM/SEGポートの番号を設定します。
その他のドットはボタンの選択にしたがって、自動的に割り付けられます。

上記の各項目を設定後[Preview]ボタンをクリックすると、パネル編集ウィンドウ内の指定位置に指定サイズのドットマトリクスが表示されます。これは設定内容の確認用で、実際にマトリクスが作成されるのは[OK]ボタンをクリックした時点です。



作成されたドットマトリクスは赤で表示されます。また、個々のドットの上にマウスポインタを重ねると、ステータスバーに割り付けられたポートとドライバの情報を表示します。また、マウスカーソルの位置にもポート情報を表示します。

ドットマトリクスをダブルクリックすると[Dot Matrix]ダイアログボックスが表示されますので、後から設定を変更することができます。
また、ドットマトリクスの位置はマウスのドラッグで変更することもできます。

注: ドットマトリクスをコピー&ペーストした場合、位置とサイズの情報は保持されますが、ポート割り付け情報は破棄されます。

5.7 注意事項

最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_utility_J.txtを参照してください。

6 ビットマップユーティリティ

6.1 概要

ビットマップユーティリティ(BmpUtil.exe、以下BmpUtilと記述)はドットマトリックスLCDの表示に使用するキャラクタジェネレータなどのビットマップイメージデータを作成します。出力データは指定のラベルが付いたアセンブリソース形式で作成されますので、そのままアセンブルしてプログラムにリンクすることができます。このビットマップユーティリティは、3.3項および4.3項に示した機種のLCDコントローラ、および表示メモリのビット配置(テクニカルマニュアルの"表示メモリ"の項を参照)に対応しています。

6.2 入出力ファイル

図6.2.1にBmpUtilの入出力ファイルを示します。

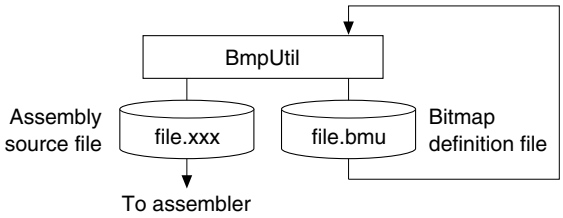


図6.2.1 BmpUtilの入出力ファイル

● ビットマップ定義ファイル(file_name.bmu)
定義したビットマップデータやCOM/SEGの出力パターンなどの情報が記録されます。拡張子は".bmu"に固定です。

● アセンブリソースファイル(file_name.xxx)
S1C63/S1C88 Family用のアセンブラで読み込み可能なテキストファイルです。ビットマップデータはデータ設定擬似命令(S1C63 Family: .word、S1C88 Family: DB)で書き出されます。拡張子は任意に指定可能です。

6.3 起動と終了

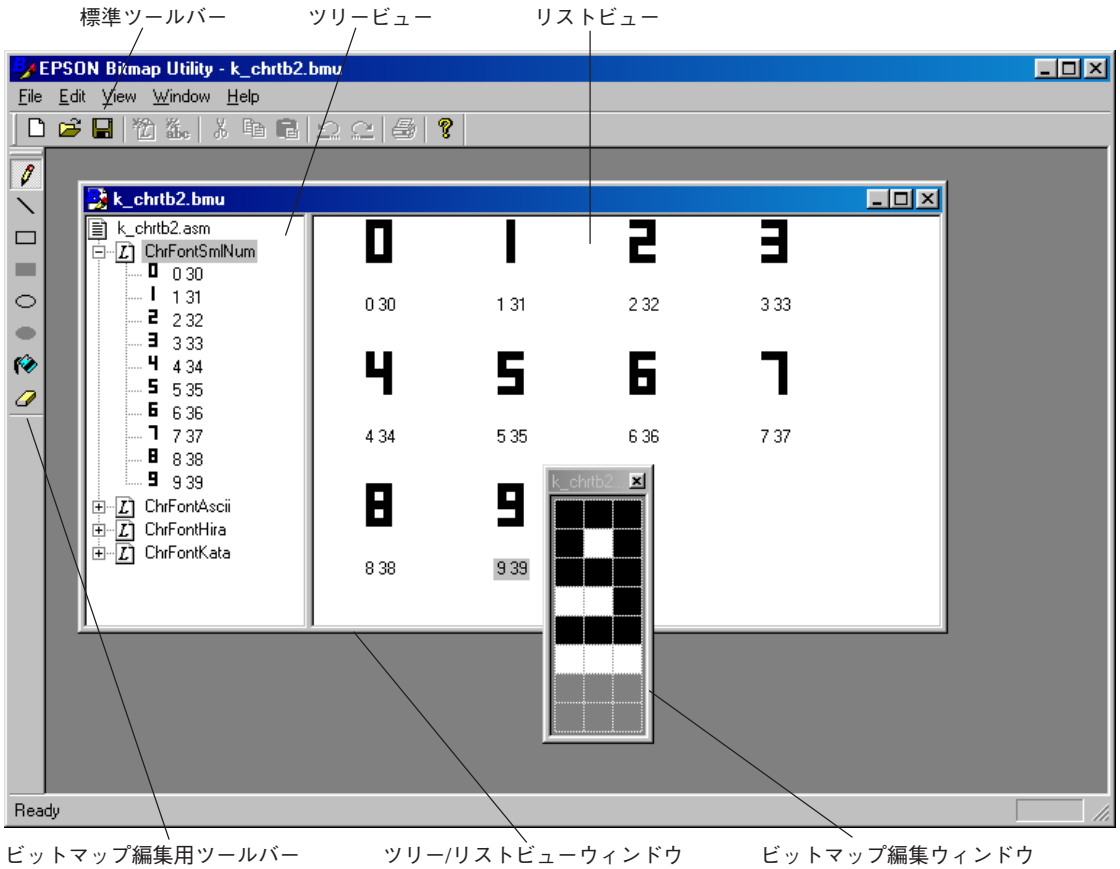

BmpUtil.exe

このアイコンをダブルクリックすると、BmpUtilが起動します。

終了するには、[File]メニューから[Exit]を選択します。

6.4　ウィンドウ

BmpUtilのウィンドウの構成を以下に示します。



● ツリー/リストビューウィンドウ

このウィンドウは個々のアセンブリリストファイルに対応し、ファイルの新規作成を選択、あるいはビットマップ定義ファイルを読み込むことにより表示されます。複数のファイルを開いて編集することができます。ウィンドウはツリービューとリストビューに分かれています。それぞれの領域のサイズは境界をドラッグすることで変更可能です。

ツリービュー

ここには、出力するアセンブリソースファイル名、ラベル、データの3階層のツリー構造が表示されます。ここでは、各項目の選択、名称の変更、ラベルやデータの追加や削除を行います。

リストビュー

ここには、ツリービューで選択されているラベルに定義されているデータの一覧が表示されます。データの名称は、アセンブリソースファイルに出力される際に付加されるコメントになります。
ここでは、データの選択、名称の変更、データの追加や削除を行います。

● ビットマップ編集ウィンドウ

このウィンドウは、リストビュー内のビットマップアイコンをダブルクリックすることにより開きます。このウィンドウ上でビットマップの作成/編集を行います。

● ツールバー

標準ツールバーとビットマップ編集用ツールバーが用意されています。標準ツールバーには[File]や[Edit]メニューにあるコマンドが登録されています。ビットマップ編集用ツールバーはビットマップイメージの編集用ツールで、ビットマップ編集ウィンドウを開くとアクティブになります。

● ウィンドウ自体の基本操作

クローズとアクティブ化

ウィンドウを閉じるには、ウィンドウの[閉じる]ボタンをクリックしてください。
開いているウィンドウは[Window]メニューにリストされます。そこからウィンドウ名を選択することで、そのウィンドウがアクティブになります。ウィンドウ上をクリックすることでも同様です。また、[Ctrl]+[Tab]のキー操作によってもアクティブウィンドウの切り替えが行えます(ツリー/リストビューウィンドウのみ)。

サイズの変更と移動

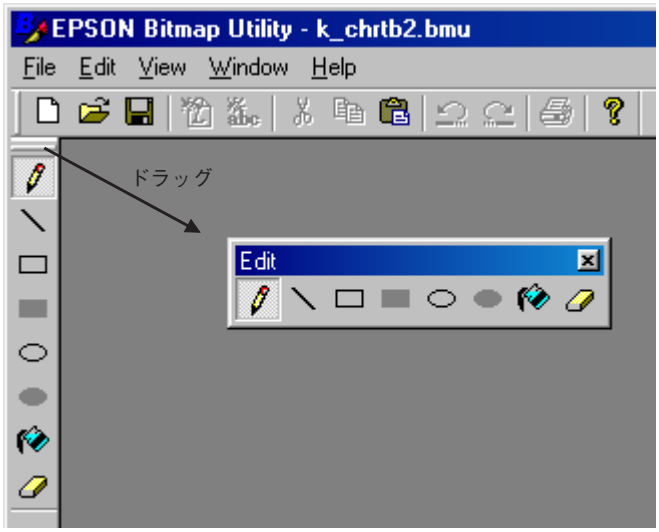
それぞれのウィンドウサイズは、ウィンドウの境界をドラッグすることによって任意の大きさに変更できます。[最小化]ボタン、[最大化]ボタン等も一般のWindowsアプリケーションと同様です。各ウィンドウはタイトルバーをドラッグすることによって、表示位置を変更できます。ただし、サイズ変更、移動ともに、アプリケーションウィンドウの範囲内に限られます。表示内容がウィンドウより大きい場合はスクロールバーが表示されます。

ウィンドウの整列

開いているウィンドウは、[Window]メニューの[Cascade]または[Tile]を選択することで整列させることができます(ツリー/リストビューウィンドウのみ)。

ツールバーの移動

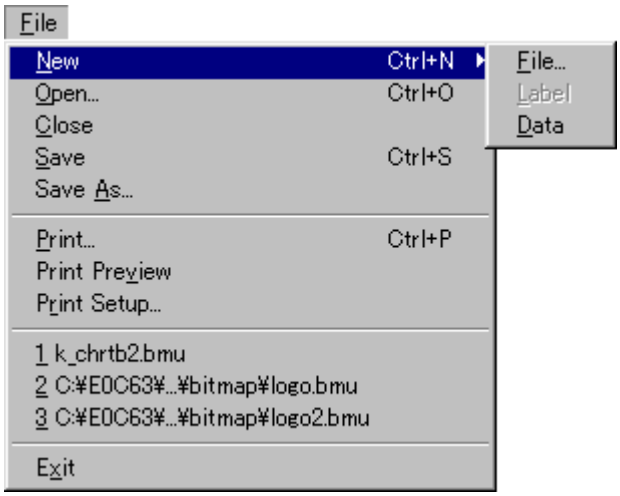
各ツールバーは、先頭部をドラッグすることで移動できます。上部にドラッグすると横位置になり、ウィンドウの左端あるいは右端にドラッグすると縦位置になります。また、ウィンドウ内にドラッグ(または先頭部をダブルクリック)すると、ツールバー自体がウィンドウになります。
なお、それぞれのツールバーは[View]メニューで隠すこともできます。



6.5 メニューとツールバー

6.5.1 メニュー

[File]メニュー



[New] ([Ctrl]+[N]) ...ツリー/リストビューウィンドウが開いている場合
[File...]
新しいビットマップ定義ファイルを作成し、作成するデータの種類やサイズなどを設定するウィザードを表示します。設定が終了するとツリー/リストビューウィンドウを開きます。

[Label]
新規ラベルを作成します。このメニュー項目はツリービューでアセンブリソースファイル名が選択されている場合にアクティブになります。

[Data]
新規ビットマップパターンを作成します。このメニュー項目はツリービューでラベルが選択されている場合にアクティブになります。

[New...] ([Ctrl]+[N]) ...ツリー/リストビューウィンドウが開いていない場合
新しいビットマップ定義ファイルを作成し、作成するデータの種類やサイズなどを設定するウィザードを表示します。設定が終了するとツリー/リストビューウィンドウを開きます。

[Open...] ([Ctrl]+[O])
ビットマップ定義ファイル(.bmu)を開きます。

[Close]
現在アクティブなツリー/リストビューウィンドウを閉じます。ウィンドウの[閉じる]ボタンと同機能です。データが変更されている場合は保存を選択するダイアログボックスが表示されます。

[Save] ([Ctrl]+[S])
アクティブなツリー/リストビューウィンドウの内容をビットマップ定義ファイル(.bmu)およびアセンブリソースファイルに保存(上書き)します。

[Save As...]
アクティブなツリー/リストビューウィンドウの内容を別の名称でビットマップ定義ファイル(.bmu)に保存します。

[Print...] ([Ctrl]+[P])
アクティブなツリー/リストビューウィンドウに定義されているすべてのビットマップイメージを一覧として印刷します。

[Print Preview]
印刷イメージを表示します。

[Print Setup...]
用紙やプリンタを選択するダイアログボックスを表示します。

ファイルリスト
ここにリストされているファイル名は最近開いたファイルです。ここからの選択でも、そのファイルを開くことができます。

[Exit]
BmpUtilを終了します。

[Edit]メニュー

[Undo] ([Ctrl]+[Z])
ビットマップ編集ウィンドウ内で行った編集操作を取り消します。
（最大4操作まで）

[Redo] ([Ctrl]+[Y])
Undoで取り消した編集操作を再実行します。（最大4操作まで）

[Cut] ([Ctrl]+[X])
選択されている項目を、クリップボードにカットします。

[Copy] ([Ctrl]+[C])
選択されている項目を、クリップボードにコピーします。

[Paste] ([Ctrl]+[V])
クリップボードにコピーされている内容をペーストします。

[Delete] ([Ctrl]+[D])
選択されている項目を、削除します。

[Rename]
ツリー/リストビューウィンドウで選択している項目の名称を変更できる状態にします。

[View]メニュー

[Toolbar]
ツールバーの表示/非表示を切り替えます。

[Status Bar]
ステータスバーの表示/非表示を切り替えます。

[Edit Bar]
ビットマップ編集用ツールバーの表示/非表示を切り替えます。

[Zoom]
ビットマップ編集ウィンドウのドット表示サイズを拡大/縮小します。

[Window]メニュー

[Cascade]
開いているツリー/リストビューウィンドウを斜めに重ねて表示します。

[Tile Horizontally]
開いているツリー/リストビューウィンドウを上下に並べて表示します。

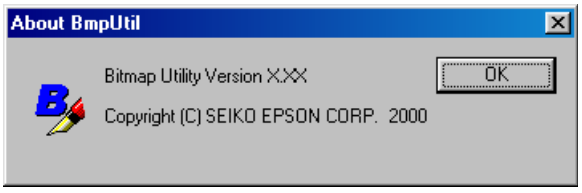
[Tile Vertically]
開いているツリー/リストビューウィンドウを左右に並べて表示します。

[Arrange Icons]
最小化されたツリー/リストビューウィンドウアイコンをアプリケーションウィンドウ領域下部に整列させます。

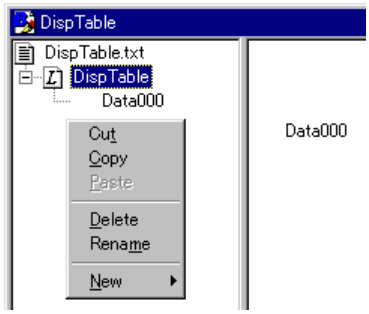
ウィンドウリスト
現在開いているツリー/リストビューウィンドウがリストされます。ここでウィンドウ名を選択すると、対応するツリー/リストビューウィンドウがアクティブになります。

[Help]メニュー

[About BmpUtil...]
BmpUtilのバージョン情報を表示します。



ポップアップメニュー



ツリービューあるいはリストビューでマウスの右ボタンをクリックすると、ポップアップメニューが表示されます。メニュー項目はそのときの選択項目に従ってアクティブになります。各項目の機能は前記のメニュー説明と同じです。

6.5.2 標準ツールバーボタン

[New File]ボタン
新しいビットマップ定義ファイルを作成し、作成するデータの種類やサイズなどを設定するウィザードを表示します。設定が終了するとツリー/リストビューウィンドウを開きます。

[Open]ボタン
ビットマップ定義ファイル(.bmu)を開きます。

[Save]ボタン
アクティブなツリー/リストビューウィンドウの内容をビットマップ定義ファイル(.bmu)およびアセンブリソースファイルに保存(上書き)します。

[New Label]ボタン
新規ラベルを作成します。このメニュー項目はツリービューでアセンブリソースファイル名が選択されている場合にアクティブになります。

[New Data]ボタン
新規ビットマップパターンを作成します。このメニュー項目はツリービューでラベルが選択されている場合にアクティブになります。

[Cut]ボタン
選択されている項目を、クリップボードにカットします。

[Copy]ボタン
選択されている項目を、クリップボードにコピーします。

[Paste]ボタン
クリップボードにコピーされている内容をペーストします。

[Undo]ボタン
ビットマップ編集ウィンドウ内で行った編集操作を取り消します。(最大4操作まで)

[Redo]ボタン
Undoで取り消した編集操作を再実行します。(最大4操作まで)

[Print]ボタン
アクティブなツリー/リストビューウィンドウに定義されているすべてのビットマップイメージを一覧として印刷します。

[About]ボタン
BmpUtilのバージョン情報を表示します。

6.5.3 ビットマップ編集用ツールバーボタン

ビットマップイメージの作成/修正に使用するボタンで、ビットマップ編集ウィンドウが開いている場合にアクティブとなります。

[Pen]ボタン
ドットをオン(黒)、オフ(白)します。

[Line]ボタン
直線を描画します。

[Rectangle]ボタン
四角形(枠)を描画します。

[Filled Rectangle]ボタン
四角形(塗りつぶし)を描画します。

[Ellipse]ボタン
円(枠)を描画します。

[Filled Ellipse]ボタン
円(塗りつぶし)を描画します。

[Fill]ボタン
同色の連続した領域を塗りつぶします。

[Erase]ボタン
指定の領域を消去(白に塗りつぶし)します。

6.6 ビットマップデータの作成

ここでは、ビットマップデータの作成方法について説明します。

6.6.1 新規作成ウィザード

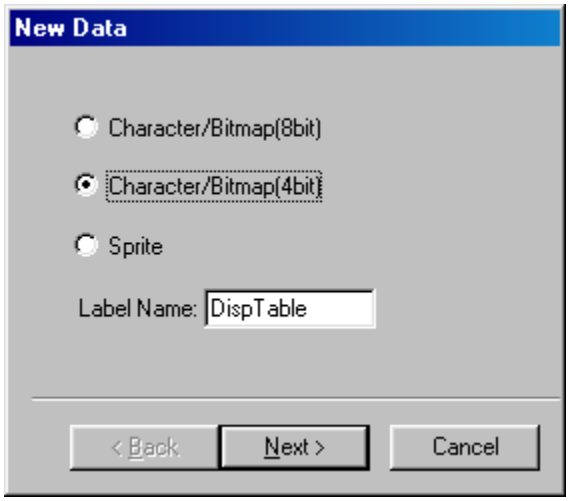
● 新規作成ウィザードの開始

ビットマップデータを新規作成するには、[File]メニューから[New...](または[New-File]*)を選択するか、[New File]ボタンをクリックします。[New Data]ダイアログボックスが表示されます。


[New File]ボタン

* [File]メニューの表示は、ツリー/リストビューウィンドウが開いていない場合は[New...]、開いている場合は[New](サブメニュー付き)になります。

● [New Data]ダイアログボックス



ここでは、作成するデータの種類とラベル名を設定します。

- データの種類

S1C63 Family用のデータを作成する場合は、Character/Bitmap(4bit)を選択してください。
S1C88 Family用のデータを作成する場合は、Character/Bitmap(8bit)とSpriteが選択可能です。

Character/Bitmap
ドットマトリクスLCDに表示するビットマップ画像やキャラクタジェネレータのフォントデータなどを作成する場合に選択します。
必ずS1C63 Familyの場合は4bitを、S1C88 Familyの場合は8bitを選択してください。

Sprite
16×16ドットのビットマップおよび透明プレーンで構成されるスプライトデータを作成する場合に選択します。
作成されるデータは、スプライト機能を持つ表示コントローラを内蔵したS1C88 Familyの機種にのみ有効です。

- ラベル

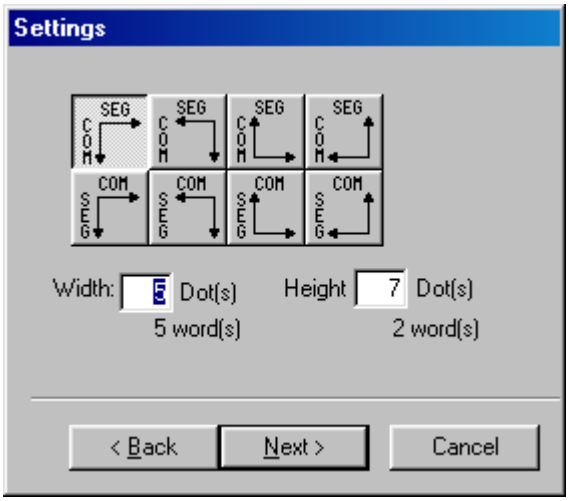
[Label Name]に、アプリケーションプログラムがデータをアクセスするために使用するラベルを入力します。この名称は、出力されるアセンブリソースファイル内にラベルとして記述されますので、アセンブラがラベルとして認識できる文字と記号を使用し、文字数もアセンブラの制限以内としてください。
また、入力内容はアセンブリソースファイル名としても使用されます。
ここで指定したラベル名、アセンブリソースファイル名はツリービュー上で後から個別に変更することもできます。
ラベルは後から追加可能で、1つのアセンブリソースファイルに複数のラベルを定義することができます。

上記の選択と入力を行い、[次へ(N)>]ボタンをクリックします。

データの種類に[Sprite]を選択した場合は、ビットマップサイズ等の設定が不要なため、[次へ(N)>]ボタンが[完了(F)]ボタンに変わります。[完了(F)]ボタンをクリックすると新規作成ウィザードは終了し、ツリー/リストビューウィンドウが表示されます。

● [Settings]ダイアログボックス([Character/Bitmap]選択時)

[New Data]ダイアログボックスで[Character/Bitmap]を選択し[次へ(N)>]ボタンをクリックすると、次の[Settings]ダイアログボックスが表示されます。[Sprite]を選択し直すには[<戻る(B)]ボタンをクリックして[New Data]ダイアログボックスに戻してください。



(S1C63 Familyの例)

ここでは、作成するビットマップイメージのサイズ、COM/SEGの配置方向を設定します。

- COM/SEGの配置方向
  ドットマトリクスLCDのCOM/SEG割り付け方向に合わせ、作成するビットマップが正しい方向で表示されるよう、8個のボタンから適切なものを1つ選択します。この選択内容はデータ作成後に変更することができませんので、注意してください。

- ビットマップイメージサイズ
  [Width]と[Height]それぞれのテキストボックスに、ビットマップイメージの水平方向、垂直方向のドット数を入力します。
  テキストボックスの下には、それぞれの方向に必要なワード数が表示されます。それらを掛け合わせることで1つのデータに必要なワード数が算出できます。

- 設定例

図6.6.1.1に5×7ドットのキャラクタジェネレータのデータを作成する場合(デフォルト設定)の例を示します。

COM/SEGの配置方向: 

Width = 5ドット、Height = 7ドット

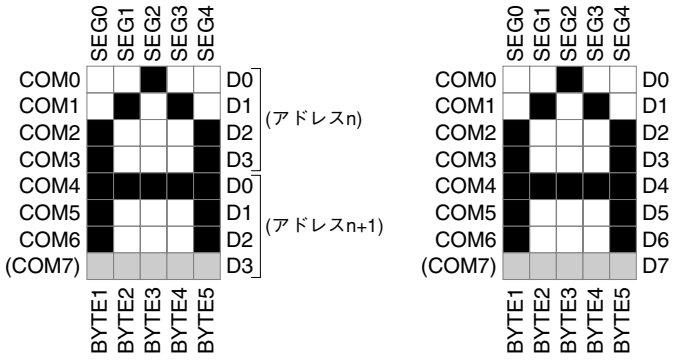


図6.6.1.1 ビットマップイメージサイズ設定例

例のCOMとSEGの番号は、選択したボタンの矢印の方向で昇順になることを示したもので、実際にデータを書き込む表示メモリのアドレスで変わります。
作成されるデータは、COM番号の昇順にデータビットに割り当てられます。
S1C63 Familyの場合、COM方向は4ビット単位となりますので、4ドットの倍数以外では無効なデータビットが発生します。例の場合はCOM0～COM6に割り付けてCOM7に対応するD3をマスクしています。これをCOM0に対応するD0をマスクするように、次に表示される[Offset]ダイアログボックスで変更することができます。
S1C88 Familyの場合、COM方向はバイト単位となりますので、8ドットの倍数以外では無効なデータビットが発生します。例の場合はCOM0～COM6に割り付けてD7をマスクしています。これをD0をマスクするように、次に表示される[Offset]ダイアログボックスで変更することができます。

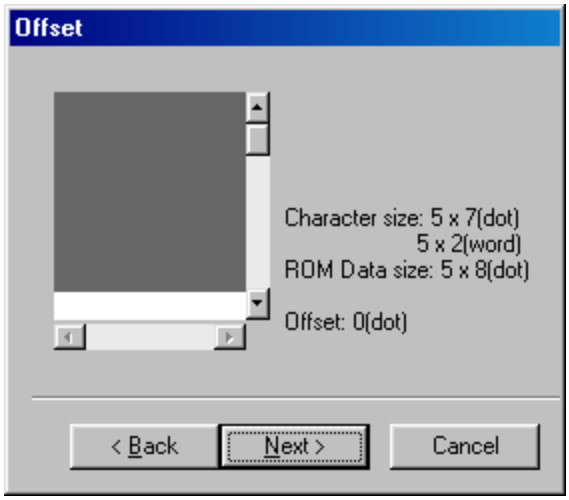
上記の選択と入力を行い、[次へ(N)>]ボタンをクリックします。

設定したサイズによっては、このダイアログで終了するため、[次へ(N)>]ボタンが[完了(F)]ボタンに変わります。[完了(F)]ボタンをクリックすると新規作成ウィザードは終了し、ツリー/リストビューウィンドウが表示されます。

● [Offset]ダイアログボックス([Character/Bitmap]選択時)

[Settings]ダイアログボックスでCOM方向に4(S1C63 Family)または8(S1C88 Family)の倍数以外を指定し[次へ(N)>]ボタンをクリックすると、次の[Offset]ダイアログボックスが表示されます。[Settings]ダイアログボックスに戻るには[<戻る(B)]ボタンをクリックしてください。

注: COM方向のドット数が4(8)の倍数の場合、[Offset]ダイアログボックスは表示されません。



(S1C63 Familyの例)

ここでは、COM方向のデータのオフセット値を指定します。

COM方向は4(8)ドット単位の表示データとなります。そのため、COM方向のドット数が4(8)の倍数以外の場合、使用しないCOM出力がでてきます。デフォルトではビットマップがCOM0から割り付けられますので、COM番号の大きな方が無効となります。
スクロールバーで囲まれたボックスは、グレーの部分がデータを配置する領域、白い部分が未使用領域を表しています。スクロールバーを操作することにより未使用領域がCOM0側に移動し、[Offset]の数値が変わります。

たとえば、前記の5×7ドットの例でオフセットを1にすると、データの配置は次のように変わります。

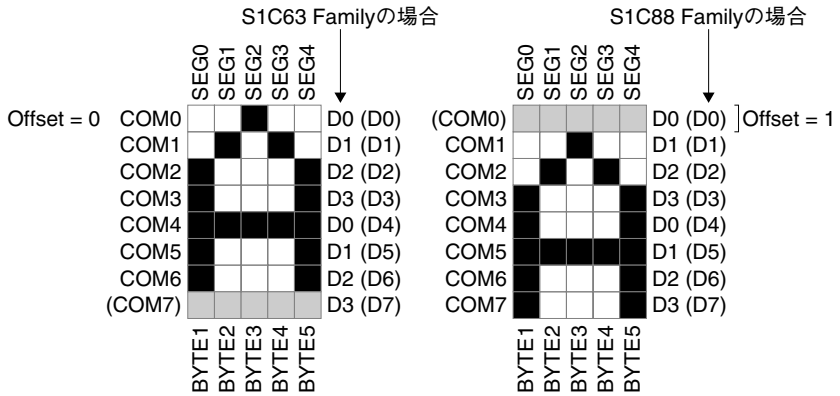


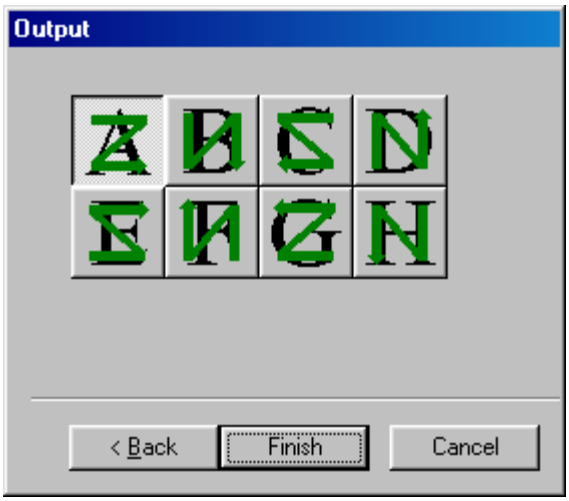
図6.6.1.2　オフセット設定例

この設定により、未使用のビットを0として最終的なデータが作成されます。
オフセット値を設定後、[次へ(N)>]ボタンをクリックします。

● [Output]ダイアログボックス([Character/Bitmap]選択時)

[Settings]ダイアログボックスでCOM方向のサイズを5ドット以上(S1C63 Family)または9ドット以上(S1C88 Family)に設定した場合、[Settings]ダイアログボックスまたは[Offset]ダイアログボックスの次に[Output]ダイアログボックスが表示されます。前のダイアログボックスに戻るには[<戻る(B)]ボタンをクリックしてください。

注: COM方向のドット数が4(8)以下の場合、[Output]ダイアログボックスは表示されません。



ここでは、データの出力順序をパターンAからHのいずれかのボタンで指定します。データはアイコンに示される順序でアセンブリソースファイルに出力されます。

16×16ドットのビットマップデータの例を図6.6.1.3に示します。

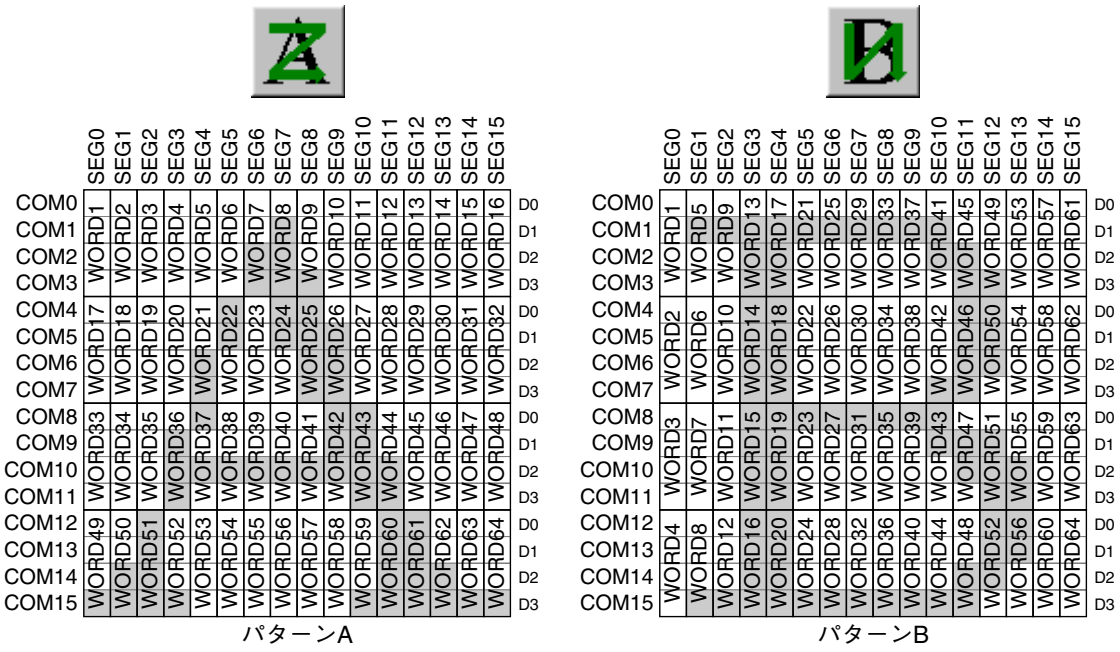


図6.6.1.3(a) データ出力パターン(S1C63 Family)

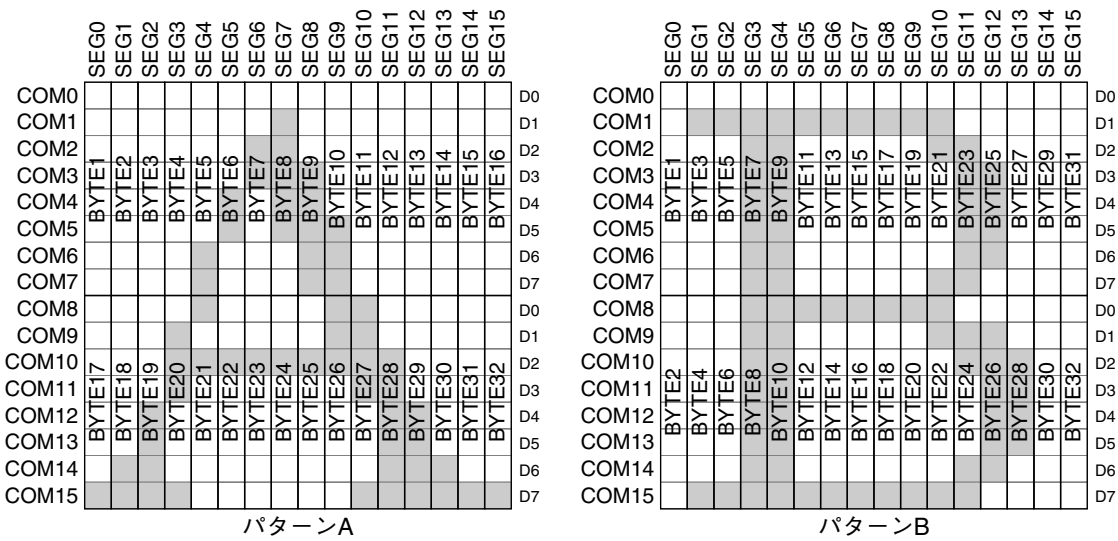
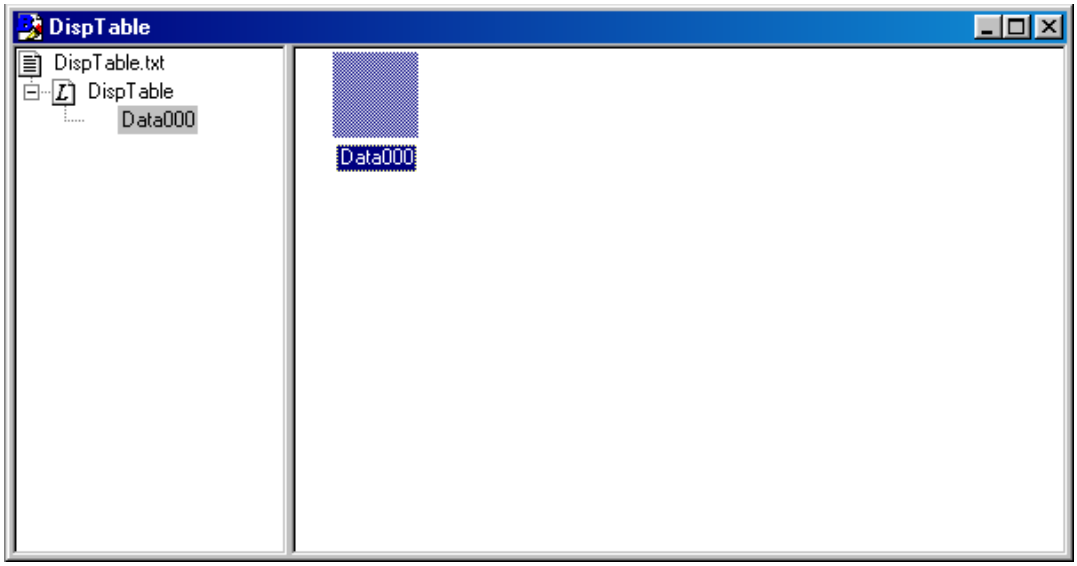


図6.6.1.3(b)　データ出力パターン(S1C88 Family)

上記の選択後、[完了(F)]ボタンをクリックすると新規作成ウィザードは終了し、ツリー/リストビューウィンドウが表示されます。

● 新規作成時のツリー/リストビューウィンドウ

新規作成ウィザードを終了すると、次のようなツリー/リストビューウィンドウが表示されます。
[Character/Bitmap]選択時の例(ラベル名はデフォルト)

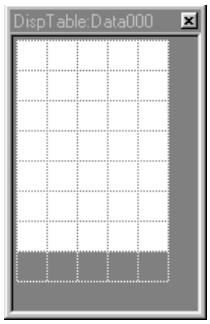


リストビューには、[New Data]ダイアログで指定した名称で、アセンブリソースファイル名とラベル名が表示されています。また、ラベル名の下にはData000の名称で新規のデータ(空白)が作成されています(ラベル名をダブルクリックすると表示されます)。これらの名称を変更する方法やラベル/データの追加方法については、"6.6.3 編集機能"を参照してください。

6.6.2 ビットマップイメージの作成

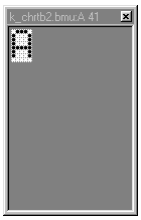
ここでは、ビットマップイメージを作成する方法を説明します。

● ビットマップ編集ウィンドウ

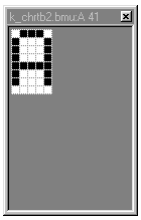


リストビューに表示されているアイコンをダブルクリックするとビットマップ編集ウィンドウが開き、ビットマップイメージの作成/編集が行えます。新規作成時はアイコン部分が空白になっていますので、データ名の上部の空間をダブルクリックしてください。

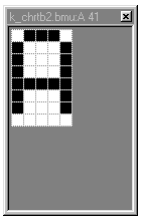
ウィンドウ内のグレー部分は、編集対象外の領域です。新規作成時、編集領域はすべて白(ドットオフ)になっています。
初期設定では、ドットのサイズが最も拡大(500%)された状態で表示されます。このサイズは[View]メニューの[Zoom]で5段階の拡大/縮小が行えます。



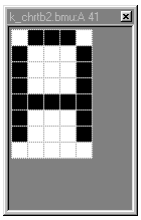
100%



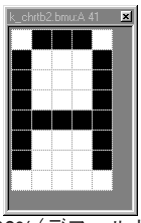
200%



300%



400%



500%(デフォルト)

ビットマップ編集ウィンドウ上にマウスカーソル(形状はビットマップ編集用ツールバーで選択したボタンで変わります)を置くと、その位置(ビットマップの左上端を0とする座標値)がステータスバーにX × Yの形式で表示されます。

後述の編集ツールでビットマップイメージを作成後は、ウィンドウの[閉じる]ボタンでビットマップ編集を終了してください。

● ビットマップ編集用ツール

ビットマップイメージの編集は、ビットマップ編集用ツールバーの以下のボタンによって行います。

[Pen]ボタン
マウスの左クリックでドットをオン(黒)、右クリックでドットをオフ(白)します。マウスのドラッグによりその軌跡をオン/オフすることもできます。

[Line]ボタン
マウスをドラッグすることにより、その始点から終点までを直線で結びます。線の色は、左ボタンを使用すると黒、右ボタンを使用すると白になります。

[Rectangle]ボタン
マウスをドラッグすることにより、その始点と終点を対角とする四角形(枠)を描画します。線の色は、左ボタンを使用すると黒、右ボタンを使用すると白になります。

[Filled Rectangle]ボタン
マウスをドラッグすることにより、その始点と終点を対角とする四角形(塗りつぶし)を描画します。線と塗りの色は、左ボタンを使用すると黒、右ボタンを使用すると白になります。

[Ellipse]ボタン
マウスをドラッグすることにより、その始点と終点を通る円(枠)を描画します。線の色は、左ボタンを使用すると黒、右ボタンを使用すると白になります。

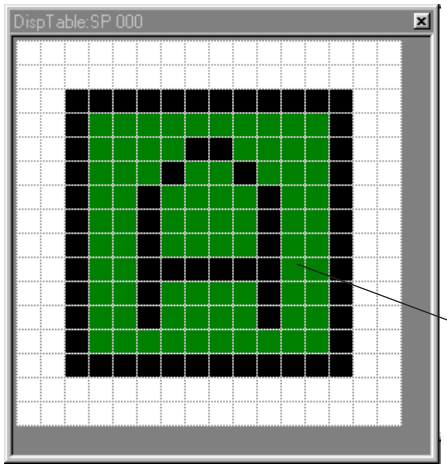
[Filled Ellipse]ボタン
マウスをドラッグすることにより、その始点と終点を通る円(塗りつぶし)を描画します。線と塗りの色は、左ボタンを使用すると黒、右ボタンを使用すると白になります。

[Fill]ボタン
マウスをクリックすると、そのドットを含み、それと同色の連続した領域を塗りつぶします。塗りの色は、左ボタンを使用すると黒、右ボタンを使用すると白になります。

[Erase]ボタン
マウスをドラッグすることにより、その始点と終点を対角とする四角形の領域を消去(白に塗りつぶし)します。

[Fill]以外のドラッグ操作を行うボタンは、マウスボタンを離す前(ドラッグ中)に[ESC]キーを押すことにより操作をキャンセルすることができます。
また、間違った操作は[Edit]メニューまたは標準ツールバーボタンの[Undo]、[Redo]により最大4操作まで取り消すことができます。

● スプライトデータの編集



スプライトデータも前記の編集ツールでビットマップイメージの編集が行えます。スプライト編集では、これに加え透明なドットが設定できます。ドットを透明にする場合は、前記の編集ツールを選択後、[Ctrl]キーを押しながらドットオンの操作(マウスの左ボタンによる操作)を行います。透明に設定されたドットは緑色で表示されます。透明の設定を解除するには、[Ctrl]キーを押しながらドットオフの操作(マウスの右ボタンによる操作)を行ってください。

6.6.3 編集機能

ここでは、前項で説明したビットマップイメージ作成を除いた編集機能を説明します。

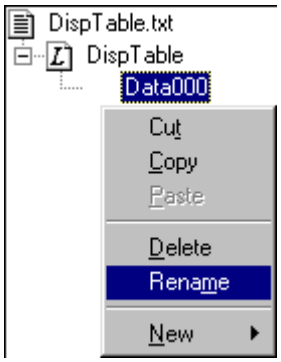
● ファイルの保存/読み込み

作成したデータは、一般のアプリケーションと同様に[File]メニューから[Save]/[Save As...] を選択(または[Save]ボタンをクリック)して、ファイルに保存します。BmpUtilは、ビットマップ定義ファイル(.bmu)とアセンブリソースファイル(ツリービューに表示されている名称)を出力します。
既存のデータに対して追加や修正を行う場合は、[File]メニューの[Open...](または[Open]ボタン)でビットマップ定義ファイルを開いてください。アセンブリソースファイルを開くことはできません。

● 名称の変更

ビットマップ定義ファイル名(.bmu)を変更するには、[File]メニューから[Save As...]を選択し別名で保存します。

アセンブリソースファイル名、ラベルやデータの名称はツリービュー上で以下の方法で変更できます。



1. 名称を変更する項目をクリックして選択します。
2. 再度、その項目をクリックします(1とあわせたダブルクリックは無効です)。または、[Edit]メニューあるいはポップアップメニュー(右クリックで表示)から[Rename]を選択します。
   名称の変更が可能な状態になりますので、全置き換えの場合はそのまま、一部を変更する場合は変更箇所を選択し直して置き換える文字を入力してください。

注:・ アセンブリソースファイル名を変更した場合、その後の最初のデータ保存時に新たな名称のアセンブリソースファイルが作成されます。

・ アセンブリソースファイル名とラベル名の長さおよび文字はアセンブラの制限に従ってください。

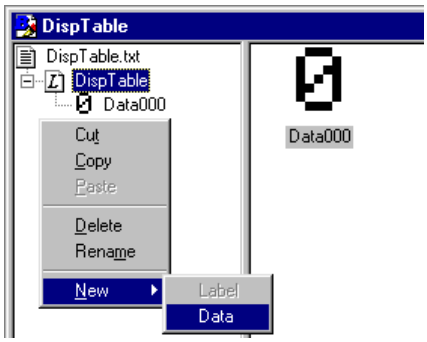
・ データ名は、出力されるアセンブリソース内の各データを区別するためのコメントとして使用されるのみで、プログラム上でその名称を制御に使用することはありません。

● データの追加、削除、コピー

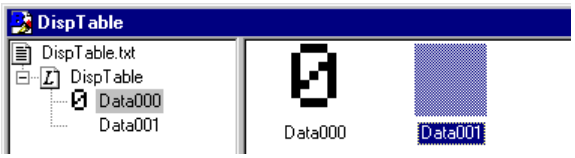
• データの追加

データの追加は、以下の手順で行います。

1. ツリービューでデータを追加するラベルを選択します。リストビューには、選択したラベルに定義されているデータの一覧が表示されます。



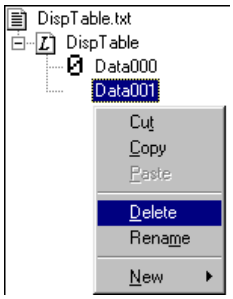
2. [File]メニューまたは右クリックにより表示されるポップアップメニューから、[New - Data]を選択します。
   ツリービューおよびリストビューに新しいデータが追加されます。必要に応じて名称を変更してください。



※ 既存のデータをもとに新たなデータを作成する場合は、データをコピー・ペーストして追加することもできます。

• データの削除

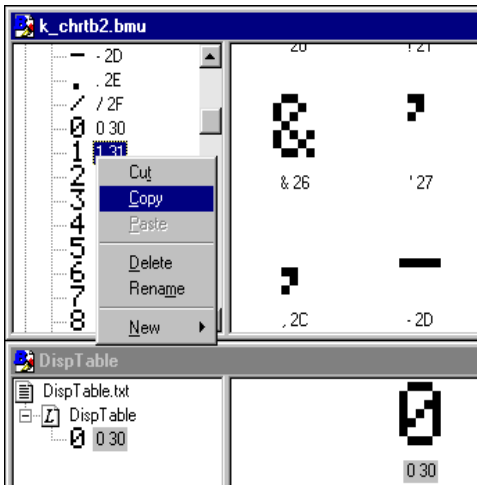
データの削除は、以下の手順で行います。



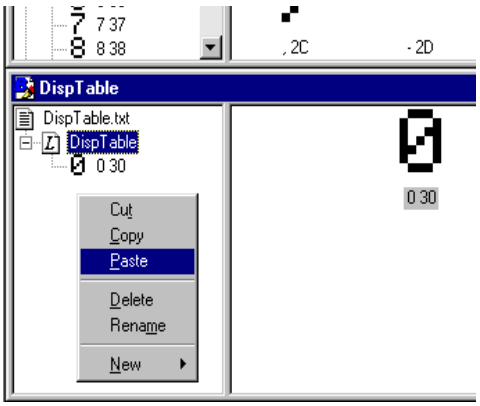
1. リストビューまたはツリービューで削除したいデータを選択します。[Shift]キーや[Ctrl]キーを使用して、複数のデータを選択することもできます。
2. [Edit]メニューまたは右クリックにより表示されるポップアップメニューから[Delete]を選択します。
   リストビューおよびツリービューから、データが削除されます。

• データのカット・コピー・ペースト

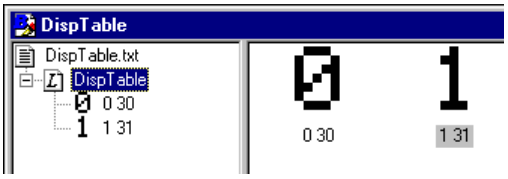
データはカット/コピー&ペーストが行えます。手順は次のとおりです。



1. リストビューまたはツリービューでカットまたはコピーしたいデータを選択します。[Shift]キーや[Ctrl]キーを使用して、複数のデータを選択することもできます。
2. [Edit]メニューまたは右クリックにより表示されるポップアップメニューから[Cut](または[Copy])を選択します(キーボードショートカット[Ctrl]+[X]、[Ctrl]+[C] も使用可能)。



3. ツリービューでペースト先のラベルを選択後(別のウィンドウのラベルも可)、[Edit]メニューまたは右クリックにより表示されるポップアップメニューから[Paste]を選択します(キーボードショートカット[Ctrl]+[V]も使用可能)。
   ツリービューおよびリストビューにデータがペーストされます。



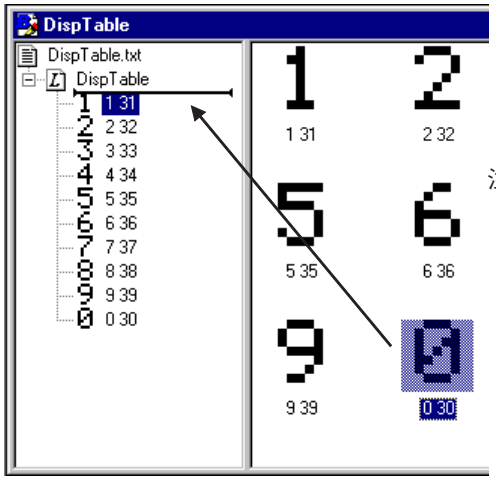
コピー先ラベルのデータサイズが異なる場合は作業の継続を確認するダイアログボックスが表示され、ペーストを中止することができます。ここでペーストを行うと、ペースト先のサイズが大きい場合は空白のドットが右側および下部に追加され、ペースト先のサイズが小さい場合はデータがクリッピングされます。

カットまたはコピーしたデータはクリップボードに保存されますので、繰り返しペーストすることができます。

ラベルを選択してペーストすると、データはそのラベル内の最後にコピーされます。一連のデータの途中にペーストする場合は、ラベルの代わりに挿入する場所のデータを選択してください。選択したデータの前にペーストされます。

- データのドラッグ＆ドロップ
  データはドラッグ＆ドロップ操作により移動またはコピーが行えます。手順は次のとおりです。

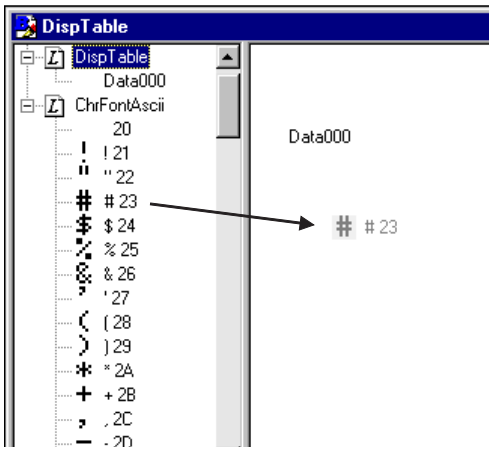
同一ラベル内でのデータの移動(順序の入れ替え)



1. ツリービューで移動させたいデータを選択します。
2. ツリービューの同じラベル内の移動先へドラッグ＆ドロップします。ドラッグ中は挿入箇所を示すカーソルが表示されます

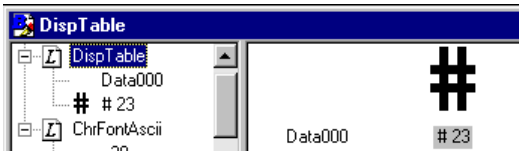
注: 他のラベルにドロップすると、移動ではなくコピーになります。
また、同じラベル内であってもリストビューとツリービューの間でドラッグ＆ドロップするとコピーになります。

データコピー1(ツリービュー→リストビュー)

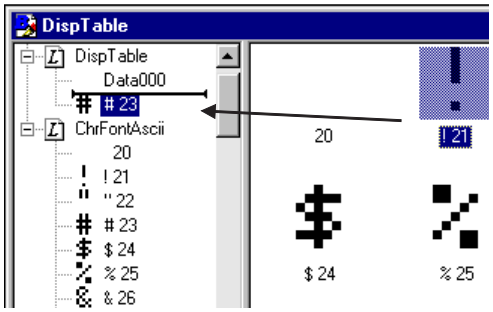


1. ツリービューでデータコピー先のラベルを選択します。リストビューにはコピー先ラベルのデータが表示されます。
2. ツリービューからコピーしたいデータをリストビュー上へドラッグします。リストビューへドロップすると、データがコピーされます。

この方法では、データはコピー先ラベル内の最後に追加されます。



データコピー2(リストビュー→ツリービュー)



1. ツリービューでコピーするデータのラベルを選択します。リストビューにはそのラベルのデータが表示されます。
2. リストビューからコピーするデータをツリービューの移動先へドラッグ＆ドロップします。ドラッグ中はツリービュー内に挿入箇所を示すカーソルが表示されます。

この方法により、データを任意の位置にコピーすることができます。

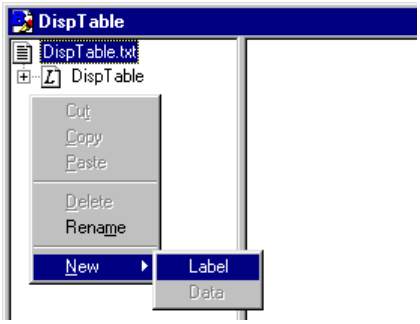
データコピー1と2で、コピーするデータとコピー先のデータのサイズが異なる場合は作業の継続を確認するダイアログボックスが表示され、コピーを中止することができます。ここでコピーを行うと、コピー先のサイズが大きい場合は空白のドットが右側および下部に追加され、コピー先のサイズが小さい場合はデータがクリッピングされます。

複数のウィンドウを開いている場合、2つのウィンドウ間でドラッグ＆ドロップによるコピーを行うこともできます。

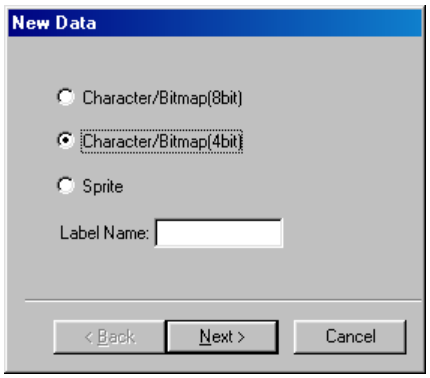
● ラベルの追加、削除、コピー

- ラベルの追加

ラベルの追加は、以下の手順で行います。



1. ツリービューの最上部にあるファイル名を選選択します。リストビューには何も表示されません。
2. [File]メニューまたは右クリックにより表示されるポップアップメニューから、[New - Label]を選択します。

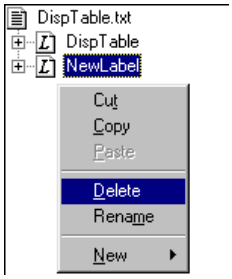


3. ラベル作成用のウィザードが表示されます。操作内容は"6.6.1 新規作成ウィザード"の説明と同じです。ラベル名、データのサイズなどを設定します。ウィザードを完了させると、1個の空白のデータを持つ新しいラベルが追加されます。

※ 既存のラベル(定義されているデータを含む)をもとに新たなラベルを作成する場合は、ラベルをコピー・ペーストして追加することもできます。

- ラベルの削除

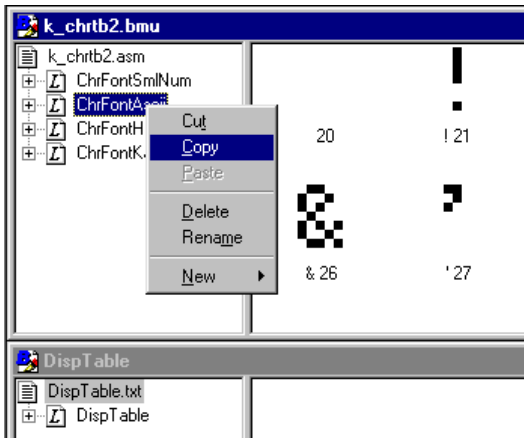
ラベルの削除は、以下の手順で行います。



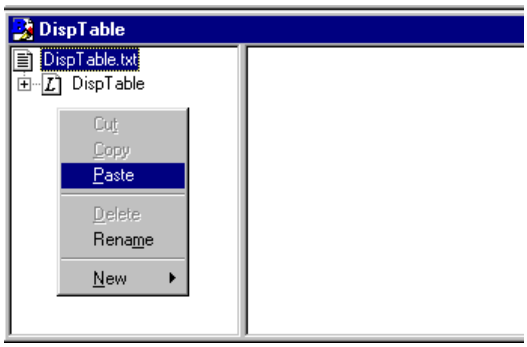
1. ツリービューで削除したいラベルを選択します。
2. [Edit]メニューまたは右クリックにより表示されるポップアップメニューから[Delete]を選択します。
   ツリービューから、ラベルおよび定義されているデータがすべて削除されます。

- ラベルのカット・コピー・ペースト

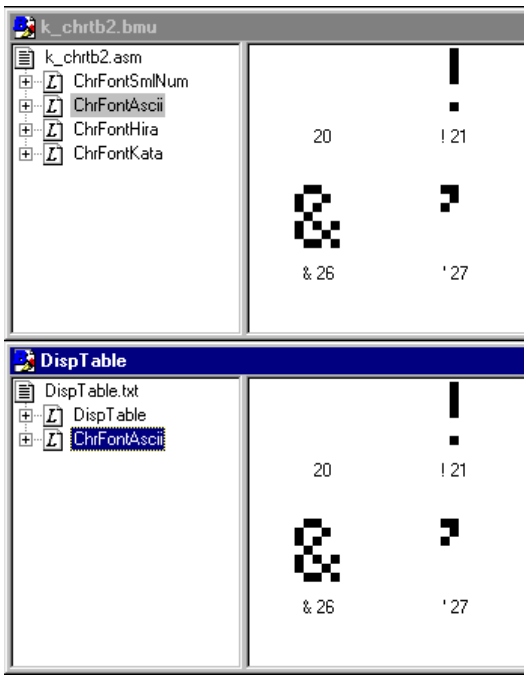
ラベルは定義されているデータも含めカット/コピー&ペーストが行えます。手順は次のとおりです。



1. ツリービューでカットまたはコピーしたいラベルを選択します。
2. [Edit]メニューまたは右クリックにより表示されるポップアップメニューから[Cut](または[Copy])を選択します(キーボードショートカット[Ctrl]+[X]、[Ctrl]+[C] も使用可能)。



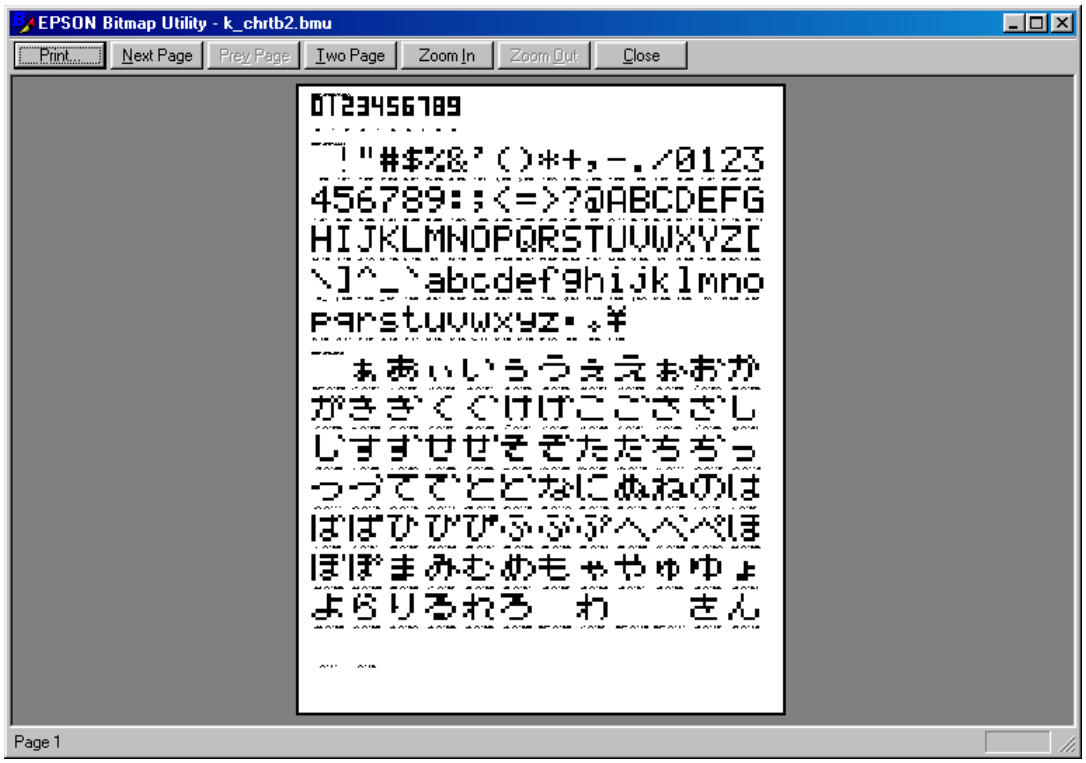
3. ツリービューでアセンブリソースファイル名を選択後(別のウィンドウのファイル名も可)、[Edit]メニューまたは右クリックにより表示されるポップアップメニューから[Paste]を選択します(キーボードショートカット[Ctrl]+[V]も使用可能)。



ツリービューの最後にラベルがペーストされ、リストビューにデータが表示されます。
ラベルはツリービューの最後にペーストされます。
カットまたはコピーしたラベル/データはクリップボードに保存されますので、繰り返しペーストすることができます。

● データの印刷

[File]メニューから[Print]を選択(またはツールバーの[Print]ボタンをクリック)することにより、作成したデータを印刷することができます。
開いているファイルに定義されているすべてのデータが出力されます。印刷のイメージは[File]メニューから[Print Preview]を選択することにより、確認できます。



6.7 アセンブリソースファイル

以下に、アセンブリソースファイルの例を示します。指定のラベルに続いてビットマップデータがバイト単位に、指定の出力順序で記述されます。それぞれのデータに付けたデータ名は、コメントになります。作成されたファイルはユーザプログラムに取り込むか、アセンブル後に他のオブジェクトといっしょにリンクしてください。

S1C63 Familyの例）

0123456789

```
ChrFontSmlNum:
      .word 0Fh,01h,0Fh ; 0 30
      .word 01h,01h,01h
      .word 00h,0Fh,00h ; 1 31
      .word 00h,01h,00h
      .word 0Dh,05h,07h ; 2 32
      .word 01h,01h,01h
      .word 05h,05h,0Fh ; 3 33
      .word 01h,01h,01h
      .word 07h,04h,0Fh ; 4 34
      .word 00h,00h,01h
      .word 07h,05h,0Dh ; 5 35
      .word 01h,01h,01h
      .word 0Fh,05h,0Dh ; 6 36
      .word 01h,01h,01h
      .word 01h,01h,0Fh ; 7 37
      .word 00h,00h,01h
      .word 0Fh,05h,0Fh ; 8 38
      .word 01h,01h,01h
      .word 07h,05h,0Fh ; 9 39
      .word 01h,01h,01h
```

COM/SEGの配置方向: 

Width = 3ドット、Height = 6ドット

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGH

IJKLMNOPQRSTUVWXYZ[¥]^_`abcdefghijklmnop

qrstuvwxyz{|}~｡｢

```
ChrFontAscii:
      .word 00h,00h,00h,00h,00h     ;   20
      .word 00h,00h,00h,00h,00h
      .word 00h,00h,0Fh,00h,00h     ; ! 21
      .word 00h,00h,04h,00h,00h
      .word 00h,07h,00h,07h,00h     ; " 21
      .word 00h,00h,00h,00h,00h
      .word 04h,0Fh,04h,0Fh,04h     ; # 23
      .word 01h,07h,01h,07h,01h
      .word 04h,0Ah,0Fh,0Ah,02h     ; $ 24
      .word 02h,02h,07h,02h,01h
      .word 03h,03h,08h,04h,02h     ; % 25
      .word 02h,01h,00h,06h,06h
      .word 06h,09h,05h,02h,00h     ; & 26
      .word 03h,04h,05h,02h,05h
      .word 00h,05h,03h,00h,00h     ; ' 27
      .word 00h,00h,00h,00h,00h
      .word 00h,0Ch,02h,01h,00h     ; ( 28
      .word 00h,01h,02h,04h,00h
      .word 00h,01h,02h,0Ch,00h     ; ) 29
      .word 00h,04h,02h,01h,00h
      .word 04h,08h,0Eh,08h,04h     ; * 2A
      .word 01h,00h,03h,00h,01h
      .word 08h,08h,0Eh,08h,08h     ; + 2B
      .word 00h,00h,03h,00h,00h
      .word 00h,00h,00h,00h,00h     ; , 2C
      .word 00h,05h,03h,00h,00h
      .word 08h,08h,08h,08h,08h     ; - 2D
      .word 00h,00h,00h,00h,00h
      .word 00h,00h,00h,00h,00h     ; . 2E
```

COM/SEGの配置方向: 

Width = 5ドット、Height = 8ドット

```
.word 00h,06h,06h,00h,00h
.word 00h,00h,08h,04h,02h     ; / 2F
.word 02h,01h,00h,00h,00h
.word 0Eh,01h,09h,05h,0Eh     ; 0 30
.word 03h,05h,04h,04h,03h
.word 00h,02h,0Fh,00h,00h     ; 1 31
.word 00h,04h,07h,04h,00h
.word 02h,01h,01h,09h,06h     ; 2 32
.word 04h,06h,05h,04h,04h
.word 01h,01h,05h,0Bh,01h     ; 3 33
.word 02h,04h,04h,04h,03h
.word 08h,04h,02h,0Fh,00h     ; 4 34
.word 01h,01h,01h,07h,01h
.word 07h,05h,05h,05h,09h     ; 5 35
.word 02h,04h,04h,04h,03h
.word 0Ch,0Ah,09h,09h,00h     ; 6 36
.word 03h,04h,04h,04h,03h
.word 01h,01h,09h,05h,03h     ; 7 37
.word 00h,07h,00h,00h,00h
.word 06h,09h,09h,09h,06h     ; 8 38
.word 03h,04h,04h,04h,03h
.word 06h,09h,09h,09h,0Eh     ; 9 39
.word 00h,04h,04h,02h,01h
.word 00h,06h,06h,00h,00h     ; : 3A
.word 00h,03h,03h,00h,00h
.word 00h,06h,06h,00h,00h     ; ; 3B
.word 00h,05h,03h,00h,00h
.word 08h,04h,02h,01h,00h     ; < 3C
.word 00h,01h,02h,04h,00h
.word 04h,04h,04h,04h,04h     ; = 3D
.word 01h,01h,01h,01h,01h
.word 00h,01h,02h,04h,08h     ; > 3E
.word 00h,04h,02h,01h,00h
.word 02h,01h,01h,09h,06h     ; ? 3F
.word 00h,00h,05h,00h,00h
.word 02h,09h,09h,01h,0Eh     ; @ 40
.word 03h,04h,07h,04h,03h
.word 0Eh,01h,01h,01h,0Eh     ; A 41
.word 07h,01h,01h,01h,07h
.word 0Fh,09h,09h,09h,06h     ; B 42
.word 07h,04h,04h,04h,03h
.word 0Eh,01h,01h,01h,02h     ; C 43
.word 03h,04h,04h,04h,02h
.word 0Fh,01h,01h,02h,0Ch     ; D 44
.word 07h,04h,04h,02h,01h
.word 0Fh,09h,09h,09h,01h     ; E 45
.word 07h,04h,04h,04h,04h
.word 0Fh,09h,09h,09h,01h     ; F 46
.word 07h,00h,00h,00h,00h
.word 0Eh,01h,09h,09h,0Ah     ; G 47
.word 03h,04h,04h,04h,07h
.word 0Fh,08h,08h,08h,0Fh     ; H 48
.word 07h,00h,00h,00h,07h
.word 00h,01h,0Fh,01h,00h     ; I 49
.word 00h,04h,07h,04h,00h
.word 00h,00h,01h,0Fh,01h     ; J 4A
.word 02h,04h,04h,03h,00h
.word 0Fh,08h,04h,02h,01h     ; K 4B
.word 07h,00h,01h,02h,04h
.word 0Fh,00h,00h,00h,00h     ; L 4C
.word 07h,04h,04h,04h,04h
.word 0Fh,02h,0Ch,02h,0Fh     ; M 4D
.word 07h,00h,00h,00h,07h
.word 0Fh,04h,08h,00h,0Fh     ; N 4E
.word 07h,00h,00h,01h,07h
.word 0Eh,01h,01h,01h,0Eh     ; O 4F
.word 03h,04h,04h,04h,03h
.word 0Fh,09h,09h,09h,06h     ; P 50
.word 07h,00h,00h,00h,00h
.word 0Eh,01h,01h,01h,0Eh     ; Q 51
.word 03h,04h,05h,02h,05h
.word 0Fh,09h,09h,09h,06h     ; R 52
.word 07h,00h,01h,02h,04h
.word 00h,00h,00h,00h,00h     ; S 53
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; T 54
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; U 55
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; V 56
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; W 57
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; X 58
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; Y 59
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; Z 5A
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; [ 5B
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; \ 5C
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; ] 5D
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; ^ 5E
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; _ 5F
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; ' 60
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; a 61
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; b 62
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; c 63
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; d 64
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; e 65
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; f 66
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; g 67
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; h 68
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; i 69
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; j 6A
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; k 6B
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; l 6C
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; m 6D
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; n 6E
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; o 6F
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; p 70
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; q 71
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; r 72
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; s 73
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; t 74
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; u 75
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; v 76
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; w 77
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; x 78
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; y 79
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; z 7A
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; · 7B
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; ◦ 7C
.word 00h,00h,00h,00h,00h
.word 00h,00h,00h,00h,00h     ; ¥ 7D
.word 00h,00h,00h,00h,00h
```

S1C88 Familyの例）

0123456789

```
ChrFontSmlNum:
      DB    01Fh,011h,01Fh    ; 0 30
      DB    000h,01Fh,000h    ; 1 31
      DB    01Dh,015h,017h    ; 2 32
      DB    015h,015h,01Fh    ; 3 33
      DB    007h,004h,01Fh    ; 4 34
      DB    017h,015h,01Dh    ; 5 35
      DB    01Fh,015h,01Dh    ; 6 36
      DB    001h,001h,01Fh    ; 7 37
      DB    01Fh,015h,01Fh    ; 8 38
      DB    017h,015h,01Fh    ; 9 39
```

Width = 3ドット、Height = 6ドット

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGH
IJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnop
qrstuvwxyz{|}~¥

```
ChrFontAscii:
      DB    000h,000h,000h,000h,000h;   20
      DB    000h,000h,04Fh,000h,000h; ! 21
      DB    000h,007h,000h,007h,000h; " 22
      DB    014h,07Fh,014h,07Fh,014h; # 23
      DB    024h,02Ah,07Fh,02Ah,012h; $ 24
      DB    023h,013h,008h,064h,062h; % 25
      DB    036h,049h,055h,022h,050h; & 26
      DB    000h,005h,003h,000h,000h; ' 27
      DB    000h,01Ch,022h,041h,000h; ( 28
      DB    000h,041h,022h,01Ch,000h; ) 29
      DB    014h,008h,03Eh,008h,014h; * 2A
      DB    008h,008h,03Eh,008h,008h; + 2B
      DB    000h,050h,030h,000h,000h; , 2C
      DB    008h,008h,008h,008h,008h; - 2D
      DB    000h,060h,060h,000h,000h; . 2E
      DB    020h,010h,008h,004h,002h; / 2F
      DB    03Eh,051h,049h,045h,03Eh; 0 30
      DB    000h,042h,07Fh,040h,000h; 1 31
      DB    042h,061h,051h,049h,046h; 2 32
      DB    021h,041h,045h,04Bh,031h; 3 33
      DB    018h,014h,012h,07Fh,010h; 4 34
      DB    027h,045h,045h,045h,039h; 5 35
      DB    03Ch,04Ah,049h,049h,030h; 6 36
      DB    001h,071h,009h,005h,003h; 7 37
      DB    036h,049h,049h,049h,036h; 8 38
      DB    006h,049h,049h,029h,01Eh; 9 39
      DB    000h,036h,036h,000h,000h; : 3A
      DB    000h,056h,036h,000h,000h; ; 3B
      DB    008h,014h,022h,041h,000h; < 3C
      DB    014h,014h,014h,014h,014h; = 3D
      DB    000h,041h,022h,014h,008h; > 3E
      DB    002h,001h,051h,009h,006h; ? 3F
      DB    032h,049h,079h,041h,03Eh; @ 40
      DB    07Eh,011h,011h,011h,07Eh; A 41
      DB    07Fh,049h,049h,049h,036h; B 42
      DB    03Eh,041h,041h,041h,022h; C 43
      DB    07Fh,041h,041h,022h,01Ch; D 44
      DB    07Fh,049h,049h,049h,041h; E 45
      DB    07Fh,009h,009h,009h,001h; F 46
```

Width = 5ドット、Height = 8ドット

```
DB    03Eh,041h,049h,049h,07Ah; G 47
DB    07Fh,008h,008h,008h,07Fh; H 48
DB    000h,041h,07Fh,041h,000h; I 49
DB    020h,040h,041h,03Fh,001h; J 4A
DB    07Fh,008h,014h,022h,041h; K 4B
DB    07Fh,040h,040h,040h,040h; L 4C
DB    07Fh,002h,00Ch,002h,07Fh; M 4D
DB    07Fh,004h,008h,010h,07Fh; N 4E
DB    03Eh,041h,041h,041h,03Eh; O 4F
DB    07Fh,009h,009h,009h,006h; P 50
DB    03Eh,041h,051h,021h,05Eh; Q 51
DB    07Fh,009h,019h,029h,046h; R 52
DB    046h,049h,049h,049h,031h; S 53
DB    001h,001h,07Fh,001h,001h; T 54
DB    03Fh,040h,040h,040h,03Fh; U 55
DB    01Fh,020h,040h,020h,01Fh; V 56
DB    03Fh,040h,038h,040h,03Fh; W 57
DB    063h,014h,008h,014h,063h; X 58
DB    007h,008h,070h,008h,007h; Y 59
DB    061h,051h,049h,045h,043h; Z 5A
DB    000h,07Fh,041h,041h,000h; [ 5B
DB    002h,004h,008h,010h,020h; \ 5C
DB    000h,041h,041h,07Fh,000h; ] 5D
DB    004h,002h,001h,002h,004h; ^ 5E
DB    040h,040h,040h,040h,040h; _ 5F
DB    000h,001h,002h,004h,000h; ' 60
DB    020h,054h,054h,054h,078h; a 61
DB    07Fh,048h,044h,044h,038h; b 62
DB    038h,044h,044h,044h,020h; c 63
DB    038h,044h,044h,048h,07Fh; d 64
DB    038h,054h,054h,054h,018h; e 65
DB    008h,07Eh,009h,001h,002h; f 66
DB    00Ch,052h,052h,052h,03Eh; g 67
DB    07Fh,008h,004h,004h,078h; h 68
DB    000h,044h,07Dh,040h,000h; i 69
DB    020h,040h,044h,03Dh,000h; j 6A
DB    07Fh,010h,028h,044h,000h; k 6B
DB    000h,041h,07Fh,040h,000h; l 6C
DB    07Ch,004h,018h,004h,078h; m 6D
DB    07Ch,008h,004h,004h,078h; n 6E
DB    038h,044h,044h,044h,038h; o 6F
DB    07Ch,014h,014h,014h,008h; p 70
DB    008h,014h,014h,014h,07Ch; q 71
DB    07Ch,008h,004h,004h,008h; r 72
DB    048h,054h,054h,054h,020h; s 73
DB    004h,03Fh,044h,040h,020h; t 74
DB    03Ch,040h,040h,020h,07Ch; u 75
DB    01Ch,020h,040h,020h,01Ch; v 76
DB    03Ch,040h,030h,040h,03Ch; w 77
DB    044h,028h,010h,028h,044h; x 78
DB    00Ch,050h,050h,050h,03Ch; y 79
DB    044h,064h,054h,04Ch,044h; z 7A
DB    000h,018h,018h,000h,000h; · 7B
DB    000h,020h,050h,020h,000h; ◦ 7C
DB    015h,016h,07Ch,016h,015h; ¥ 7D
```

6.8 注意事項

最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_utility_J.txtを参照してください。

7　ポート設定ユーティリティ

7.1　概要

ポート設定ユーティリティ(PrtUtil.exe、以下PrtUtilと記述)は、シミュレータ(sim63.exe、sim88.exe)がターゲットのキー入力をシミュレートするために必要な、プッシュキーやキーマトリクスのポート割り付け、およびシミュレーションに使用するPCキーを設定するユーティリティです。設定したデータはテキスト形式でポート設定ファイル(.prt)に書き出され、それをシミュレータで読み込むことによりターゲットのキー入力をPCのキーボードを使用してシミュレートできます。
PrtUtilは機種に依存しない共通のツールです。

7.2　入出力ファイル

図7.2.1にPrtUtilの入出力ファイルを示します。

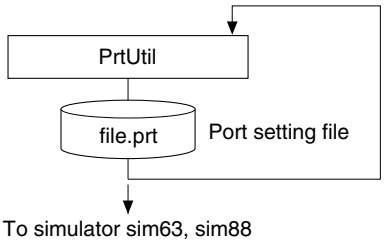


図7.2.1　PrtUtilの入出力ファイル

● ポート設定ファイル(file_name.prt)
プッシュキーやキーマトリクスとポートの対応を定義したテキストファイルです。このファイルをシミュレータにロードしてキー入力のシミュレーションを行います。作成したポート設定ファイルは、再度PrtUtilに読み込んで修正可能です。

7.3　起動と終了


PrtUtil.exe

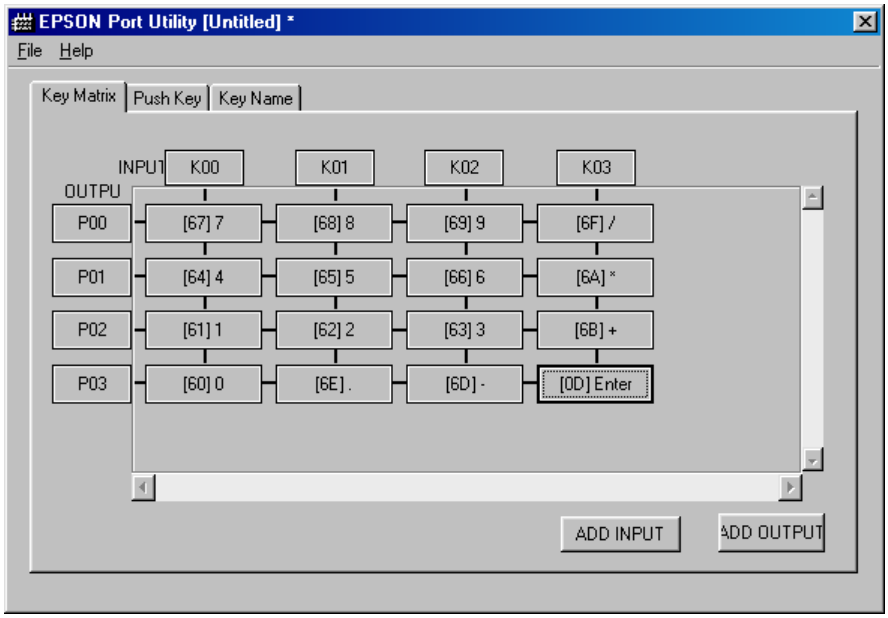
このアイコンをダブルクリックすると、PrtUtilが起動します。

終了するには、[File]メニューから[Exit]を選択します。

7.4　ウィンドウ

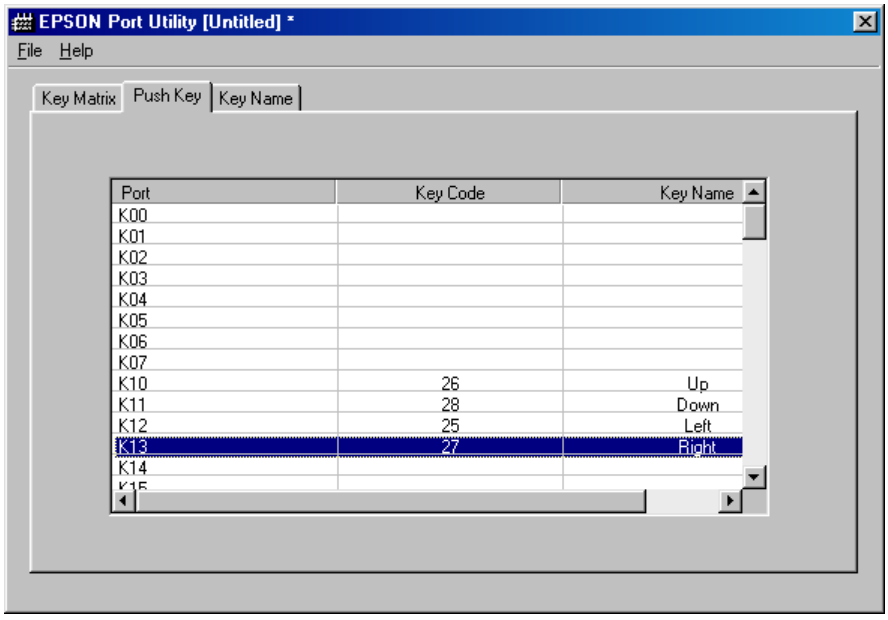
PrtUtilのウィンドウには[Key Matrix]、[Push Key]、[Key Name]の3つのタブがあり、設定する項目により選択します。

● [Key Matrix]



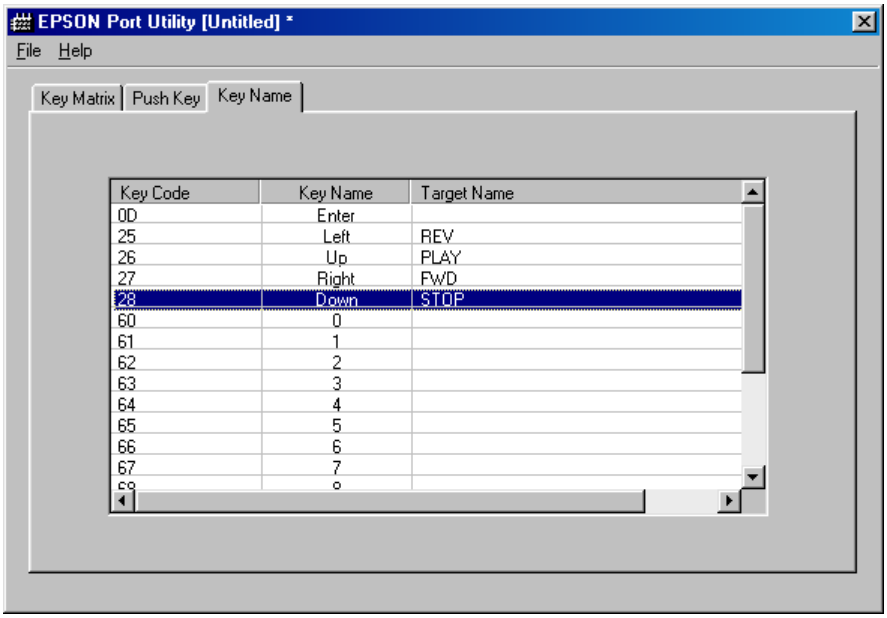
ここでは、キーマトリクスを構成する入出力ポートと、キー入力シミュレーションに使用するPCキーの指定を行います。

● [Push Key]



ここでは、プッシュキー入力に使用する入力ポートと、キー入力シミュレーションに使用するPCキーの指定を行います。

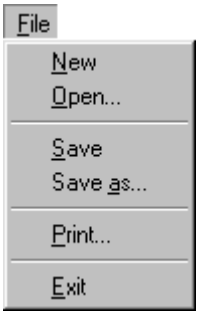
● [Key Name]



ここでは、キー入力シミュレーションに使用するPCキーにターゲット上のキー名称を定義します。

7.5　メニュー

[File]メニュー



[New]
新しいポート設定を開始します。

[Open...]
ポート設定ファイル(.prt)を開きます。

[Save]
現在編集中の内容をポート設定ファイル(.prt)に保存(上書き)します。

[Save as...]
現在編集中の内容を別の名称でポート設定ファイル(.prt)に保存します。

[Print...]
設定内容を印刷します。

[Exit]
PrtUtilを終了します。

[Help]メニュー

[About PrtUtil...]
PrtUtilのバージョン情報を表示します。

7.6　ポート設定データの作成

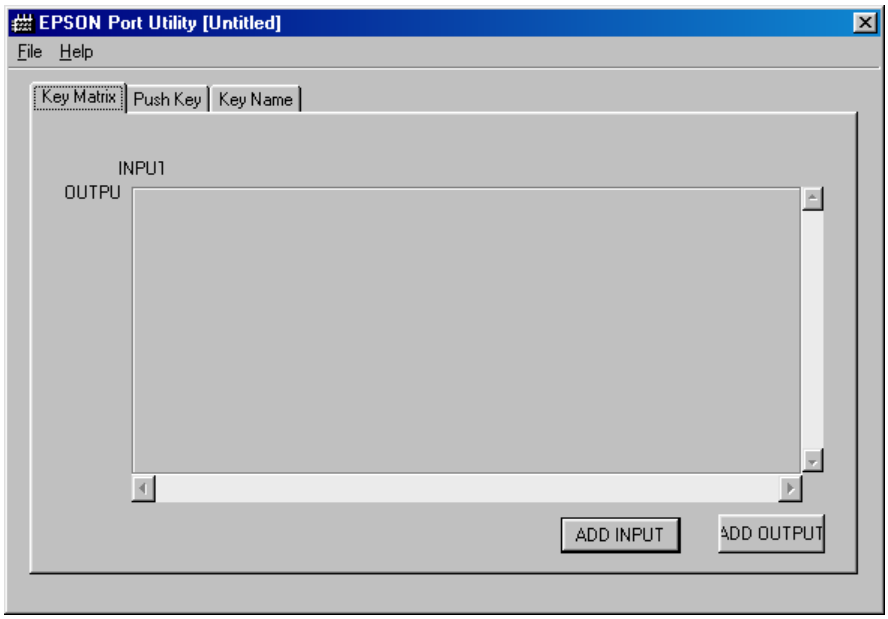
シミュレータは、S1C63xxx/S1C88xxxの入出力ポートを使用したキーマトリクスおよび入力ポートに接続するプッシュキーのキー入力シミュレーションに対応しています。本ユーティリティでは、以下の3種類の設定を個別に行うことができます。

1. キーマトリクスを構成するポートおよびシミュレーションに使用するPCキーの指定
2. プッシュキーを接続する入力ポートとシミュレーションに使用するPCキーの指定
3. シミュレータ上でPCキーをターゲットシステムのキー名称で扱うための定義

不要な項目を設定する必要はありません。
設定データは1つのポート設定ファイルにまとめられます。
以下、データの作成方法を説明します。

7.6.1 データの新規作成

PrtUtilを起動すると次のウィンドウが表示されます。



この状態で新規データの作成を開始することができます。
データの編集中/編集後に新しいデータを作成するには、[File]メニューから[New]を選択します。設定内容がすべてクリアされ、起動直後と同じ状態になります。編集中の最新データが保存されていない状態で[New]を選択すると、データを保存するか確認するメッセージが表示され、データを保存するか破棄するか、あるいは新規作成を中止するか選択できます。

7.6.2 既存データの編集

既存データを修正する場合などは、[File]メニューから[Open...]を選択し、編集するポート設定ファイルを読み込みます。編集中の最新データが保存されていない状態で[Open...]を選択すると、データを保存するか確認するメッセージが表示され、データを保存するか破棄するか、あるいはファイルの読み込みを中止するか選択できます。
ファイルが読み込まれるとウィンドウに設定内容が表示されます。必要な追加や修正を行ってください。
なお、作成、編集したデータは必ず[File]メニューの[Save]または[Save as...]で保存してください。

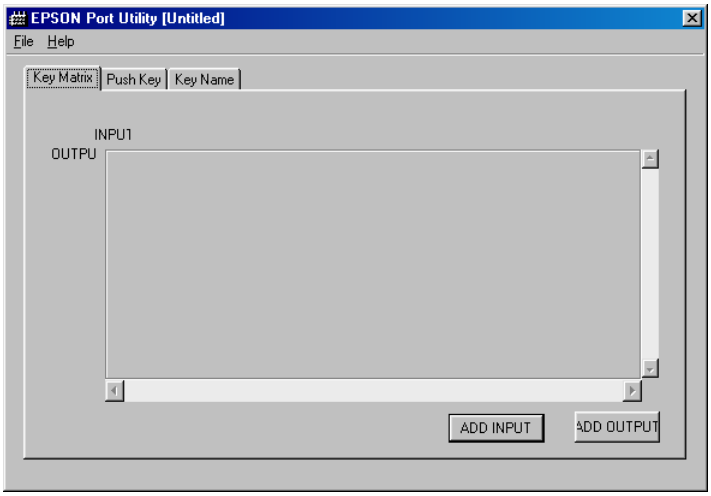
7.6.3 キーマトリクスデータの設定

ここでは、キーマトリクスデータの作成、修正方法を説明します。キーマトリクスを持たないターゲットシステムの場合は何も設定する必要はありません。

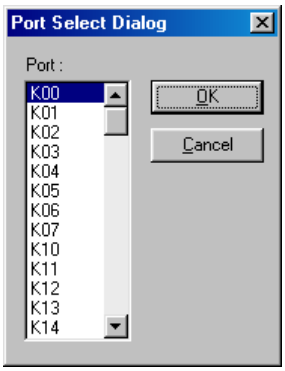
● キーマトリクスデータの新規作成

以下に新規作成の手順を示します。

1）[Key Matrix]タブをクリックし、キーマトリクス編集画面を表示させます。



2）[ADD INPUT]ボタンをクリックします。
[Port Select]ダイアログボックスが表示されます。



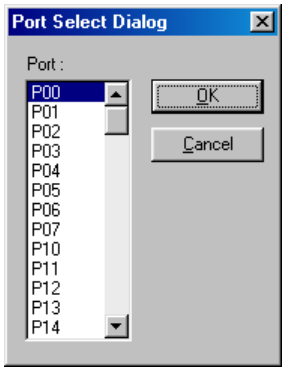
ここには、入力ポート（Kxx、Pxx）の一覧が表示されます。

注: 本ユーティリティはS1C63/S1C88 Familyの全機種に共通のため、個別機種には用意されていないポートの名称も表示されますので注意してください。

3）キーマトリクス仕様に従って使用する入力ポートを選択し、[OK]をクリックします。

選択したポート名が編集画面に表示されます。
同様に、キーマトリクスを構成するすべての入力ポートを選択します。

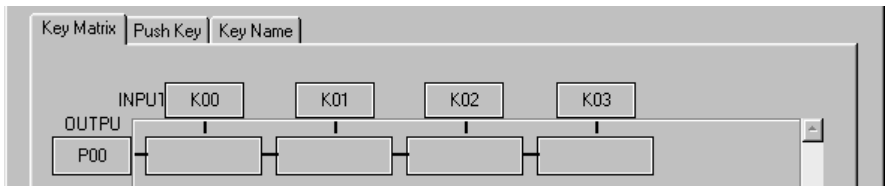
4）[ADD OUTPUT]ボタンをクリックします。
[Port Select]ダイアログボックスが表示されます。



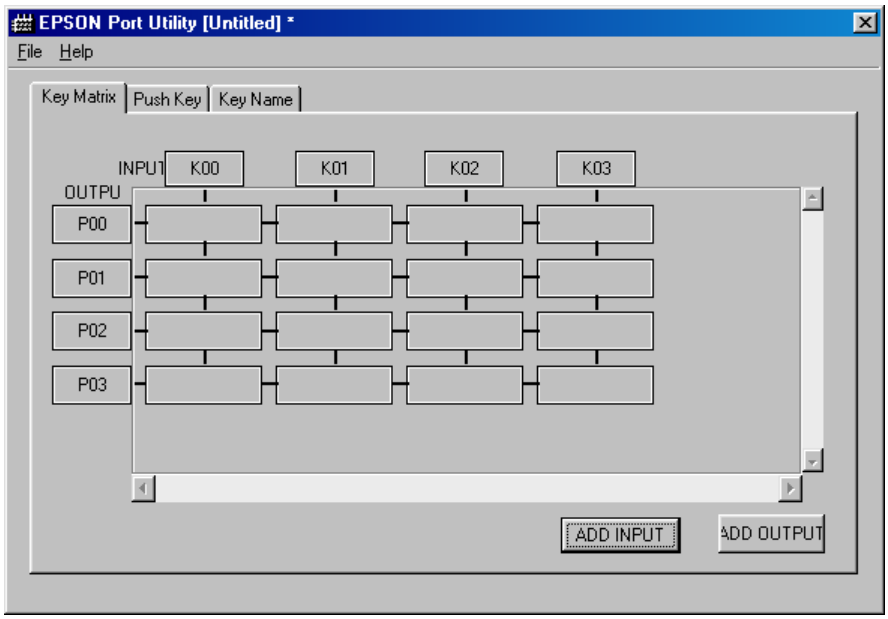
ここには、出力ポート（Pxx、Rxx）の一覧が表示されます。

注: 本ユーティリティはS1C63/S1C88 Familyの全機種に共通のため、個別機種には用意されていないポートの名称も表示されますので注意してください。

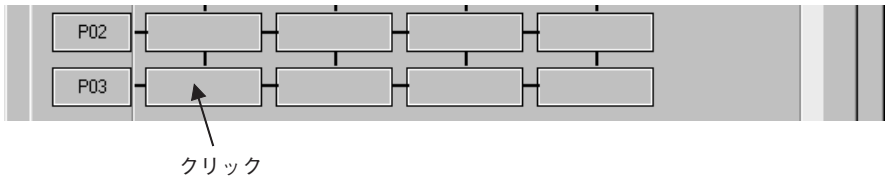
5）キーマトリクス仕様に従って使用する出力ポートを選択し、[OK]をクリックします。



選択したポート名が編集画面に表示されます。
同様に、キーマトリクスを構成するすべての出力ポートを選択します。

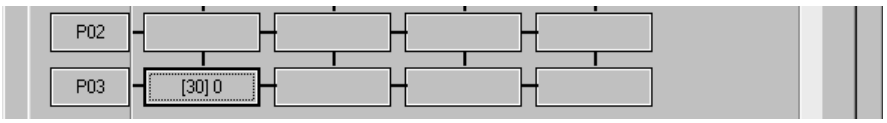


6）PCキーボードのキーを割り当てるボックスをクリックします。

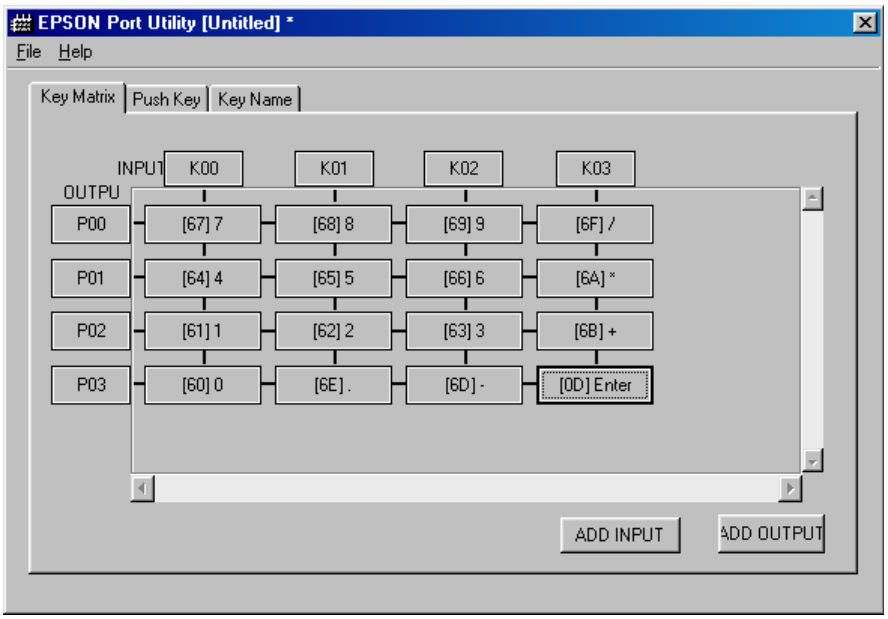


[Push Key]ダイアログボックスが表示されます。

7）シミュレーションに使用するキーを押します。
ダイアログボックスが閉じて、入力したキーのコードとキャラクタ/名称がボックス内に表示されます。



ほかのキーも同様に割り当てます。



注: 標準キーボードとテンキーパッドでは、同じ数字や記号でもコードが異なりますので注意してください。

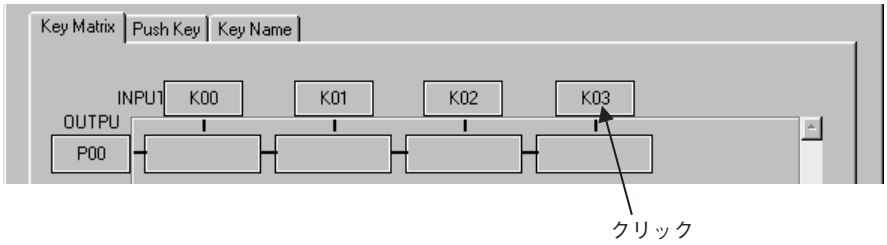
8）[File]メニューから[Save]または[Save as...]を選択してデータを保存します。

● キーマトリクスデータの修正

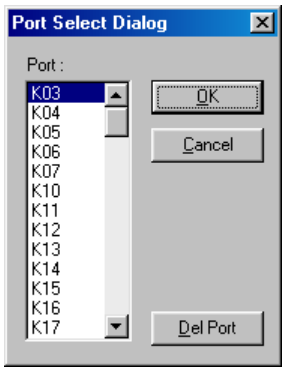
作成したキーマトリクスデータの修正方法を以下に示します。

ポートの変更

1）変更あるいは削除する入力ポート名または出力ポート名をクリックします。



[Port Select]ダイアログボックスが表示されます。



2）他のポートに置き換える場合は、一覧から新たなポート名を選択して[OK]をクリックします。

3）ポートを削除するには[Del Port]をクリックします。キーマトリクスが一列削除されます。

4）変更を中止する場合は[Cancel]をクリックします。

PCキーの変更

1）変更するPCキーのボックスをクリックします。
[Push Key]ダイアログボックスが表示されます。

2）ほかのキーを割り当てる場合は、そのキーを押します。

3）PCキーの割り当てを抹消するには[Clear]をクリックします。

4）変更を中止する場合は[Cancel]をクリックします。

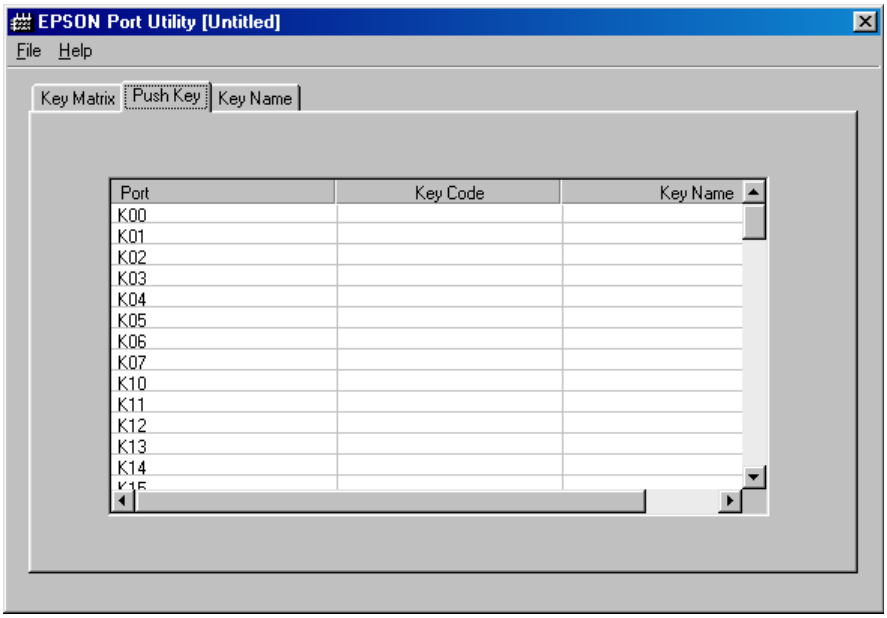
7.6.4 プッシュキーデータの設定

ここでは、プッシュキーデータの作成、修正方法を説明します。プッシュキーを持たないターゲットシステムの場合は何も設定する必要はありません。

● プッシュキーデータの新規作成

以下に新規作成の手順を示します。

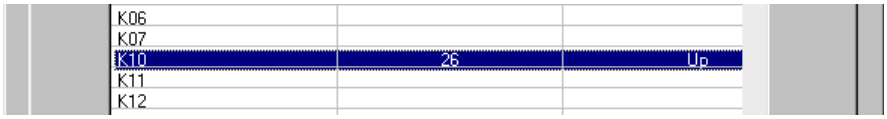
1）[Push Key]タブをクリックし、プッシュキー編集画面を表示させます。



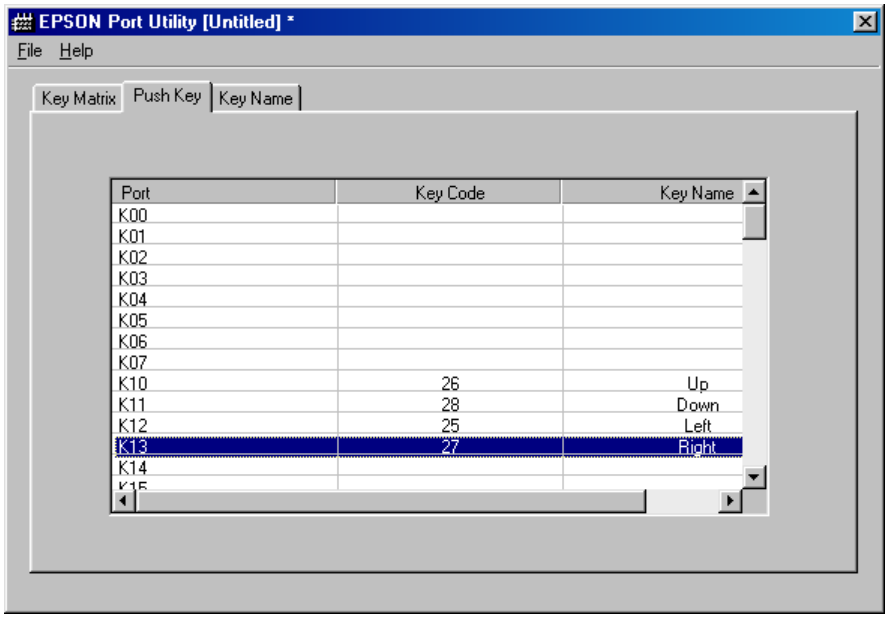
2）プッシュキーが接続される入力ポートの行をダブルクリックします。
[Push Key]ダイアログボックスが表示されます。

注: 本ユーティリティはS1C63/S1C88 Familyの全機種に共通のため、個別機種には用意されていないポートの名称も表示されますので注意してください。

3）シミュレーションに使用するキーを押します。
ダイアログボックスが閉じて、入力したキーのコードとキャラクタ/名称が選択したポートの行に表示されます。



同様に、プッシュキーを接続するすべての入力ポートを設定します。



注: 標準キーボードとテンキーパッドでは、同じ数字や記号でもコードが異なりますので注意してください。

4）[File]メニューから[Save]または[Save As...]を選択してデータを保存します。

● プッシュキーデータの修正

作成したプッシュキーデータの修正方法を以下に示します。

1）変更する入力ポートの行をダブルクリックします。
[Push Key]ダイアログボックスが表示されます。

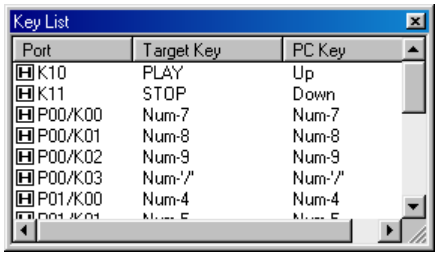
2）ほかのキーを割り当てる場合は、そのキーを押します。

3）PCキーの割り当てを抹消するには[Clear]をクリックします。

4）変更を中止する場合は[Cancel]をクリックします。

7.6.5 ターゲットキー名称の設定

ここでは、シミュレータ上でPCキーをターゲットシステムのキー名称で扱うための指定方法を説明します。
ここで設定したターゲットのキー名称は、シミュレータの[Key List]ウィンドウにPCキー名称とともに表示されます。
ここでの設定はデバッグ操作を支援するもので、シミュレーションの機能や動作に影響を与えるものではありません。



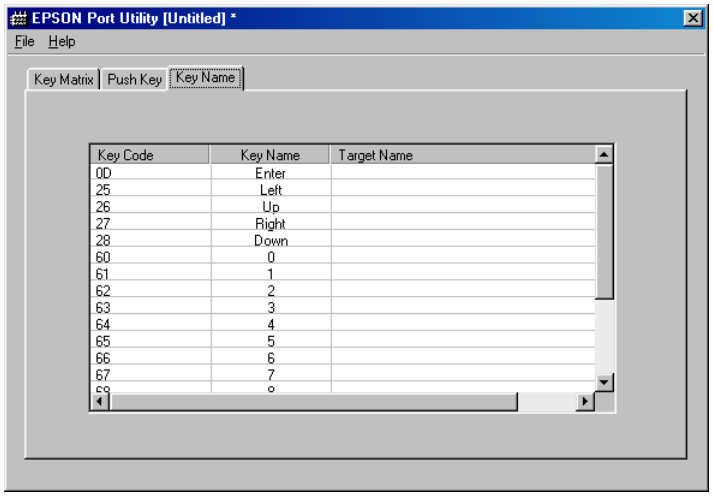
（シミュレータ [Key List]ウィンドウの表示例）

ターゲットのキー名称が定義されないものについては、PCキーと同じ名称が表示されます。

以下にターゲットキー名称を定義する手順を示します。

1）あらかじめ、キーマトリクスとプッシュキーの設定を済ませておきます。

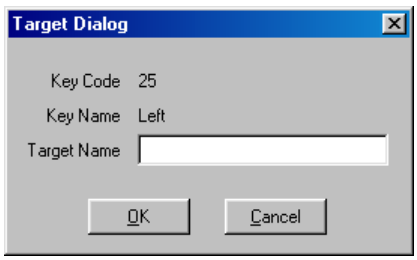
2）[Key Name]タブをクリックし、キー名称編集画面を表示させます。



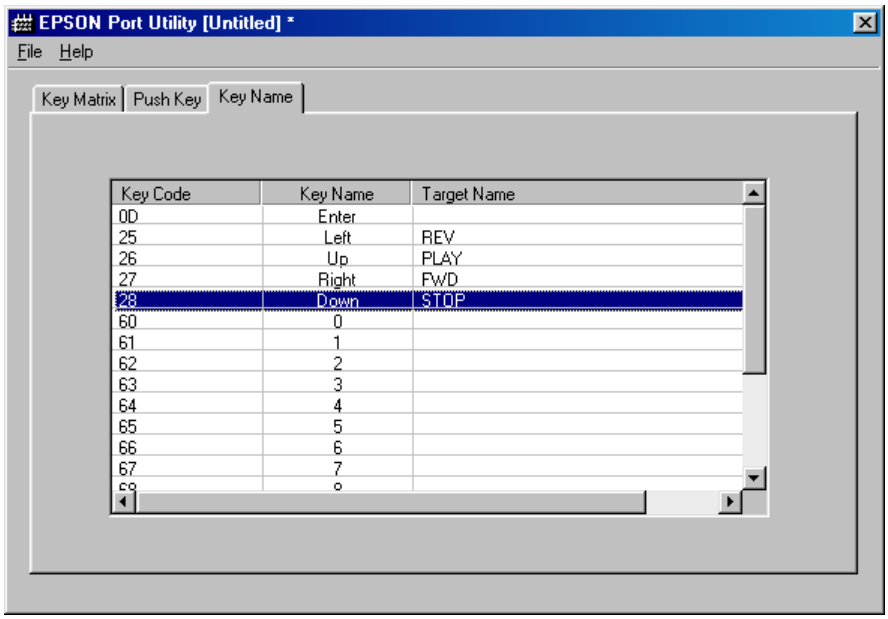
キーマトリクス、プッシュキーの設定で割り当てたPCキーのコードとキー名称が表示されます。

注: キーマトリクス、プッシュキーでPCキーの割り当てを行っていない場合、ここにキーコードとキー名称は表示されません。

3）ターゲットのキー名称を定義したいPCキーの行をダブルクリックします。
[Target]ダイアログボックスが表示されます。



4）[Target Name]ボックスにターゲットのキー名称を入力し、[OK]をクリックします。
ダイアログボックスが閉じて、入力したキー名称が選択した行に表示されます。



5）3と4の操作を必要なキーの分だけ繰り返します。

6）設定したキー名称を変更する場合も、3と4の操作を行ってください。定義したキー名称をクリアするには、[Target]ダイアログボックスの[Target Name]ボックス内の文字を削除して[OK]をクリックします。

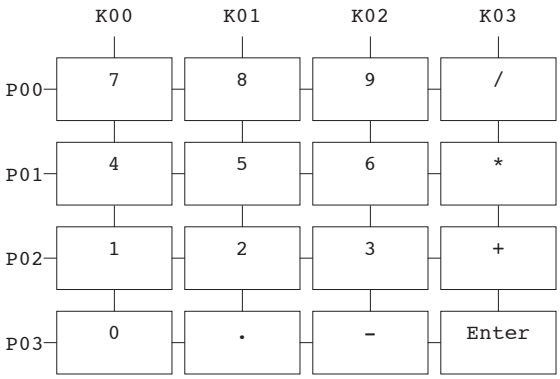
7.6.6 印刷

[File]メニューから[Print...]を選択することにより、作成したデータを印刷することができます。

印刷例:

```
[Push Key]
K10=Up [PLAY]
K11=Down [STOP]
K12=Left [REV]
K13=Right [FWD]


[Key Matrix]
```



7.7　注意事項

- PCの標準キーボードとテンキーパッドでは、同じ数字や記号でもコードが異なりますので注意してください。
  例: 標準キーボードの0～9 = 30h～39h、テンキーパッドの0～9 = 60h～69h

  特にデスクトップPCとノートPCで同じ定義ファイルを使用する場合は注意が必要です。
- PCキーボードの以下のキーはキー入力シミュレーション用に割り当てることはできません。
  ALT, Windows, Print, Screen, Num Lock, Scroll Lock, Pause (break), Caps, Fn (Function), アルファベット以外の言語用の特殊キー
- 本ユーティリティはS1C63/S1C88 Familyの全機種に共通のため、個別機種には用意されていないポートの名称も表示されます。それらを選択しないように注意してください。

最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_utility_J.txtを参照してください。

7.8 ポート設定ファイル(.prt)

ポート設定ファイルの内容を以下に示します。定義内容は[Push Key]、[Key Matrix]、[Key Name]の3セクションに分けられ、設定したセクションのみが記述されます。

プッシュキーの定義

セクションの宣言[Push Key]に続けて、ポートとキーコード(後述)の対応が記述されます。

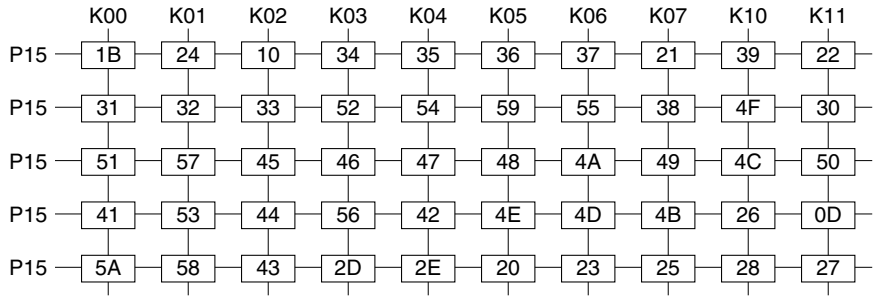
例:

```
[Push Key]
K00=30
K01=31
K02=32
K03=33
K04=34
K05=35
K06=36
K07=37
K10=38
K11=39
```

キーマトリックスの定義

セクションの宣言[Key Matrix]に続けて、使用する入力ポートと出力ポートが記述されます。次に、入力ポートごとに、割り当てられたPCキーのコードが記述されます。キーがない場合は、空白になります。

例:




```
[Key Matrix]
OUTPUT=P15,P16,P17,R27,R34
INPUT =K00,K01,K02,K03,K04,K05,K06,K07,K10,K11

P15   = 1B, 24, 10, 34, 35, 36, 37, 21, 39, 22
P16   = 31, 32, 33, 52, 54, 59, 55, 38, 4F, 30
P17   = 51, 57, 45, 46, 47, 48, 4A, 49, 4C, 50
R27   = 41, 53, 44, 56, 42, 4E, 4D, 4B, 26, 0D
R34   = 5A, 58, 43, 2D, 2E, 20, 23, 25, 28, 27
```

キー名称

セクションの宣言[Key Name]に続けて、プッシュキーまたはキーマトリクスに割り当てられているPCキーに定義するターゲットの名称が記述されます。

例:

```
[Key Name]
24=Caps         ←Homeキー（キーコード24）=Caps
22=On/Off       ←PageDownキー（キーコード22）=On/Off
21=Mode         ←PageUpキー（キーコード21）=Mode
23=Char         ←Endキー（キーコード23）=Char
2D=Data         ←Insertキー（キーコード2D）=Data
```

キーコード一覧（16進）

PCのキーボードでキー入力したときにシミュレータが受け取るキーコードです。テンキー（NumPad）は独立したキーコードを持っています。

キー	コード
0～9	30～39
A～Z	41～5A
BackSpace	08
Tab	09
Enter	0D
Shift	10
Ctrl	11
Pause	13
Esc	1B
Space	20
PageUp	21
PageDown	22
End	23
Home	24
Left	25
Up	26
Right	27
Down	28
Insert	2D
Delete	2E
（テンキー）	
0～9	60～69
*	6A
+	6B
-	6D
.	6E
/	6F
（ファンクションキー）	
F1～F24	70～87

8　ロジックアナライザ

8.1　概要

ロジックアナライザ(LogAna.exe、以下LogAnaと記述)は、シミュレータ(Sim63/Sim88)の[I/O Terminal]ウィンドウで作成した入出力ログファイルを入力し、そこに記録されている入出力ポートの状態(シリアル入出力とA/D変換ログには未対応)を波形表示します。LogAnaは機種に依存しない共通のツールです。
シミュレータの[I/O Terminal]ウィンドウによるポート入出力シミュレーションについては、3.6.4項(S1C63 Family)または4.6.4項(S1C88 Family)の"[I/O Terminal]ウィンドウ"を参照してください。

8.2　入力ファイル

LogAnaは[I/O Terminal]の入出力ログファイルを入力します。

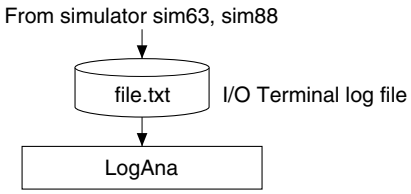


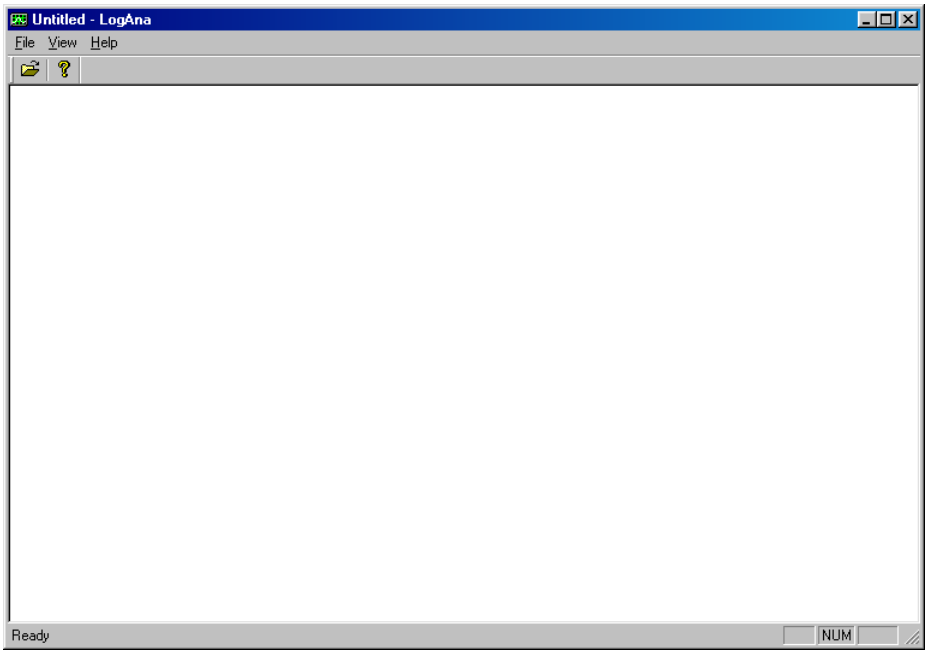
図8.2.1　LogAnaの入力ファイル

● 入出力ログファイル(file_name.txt)
シミュレータの[I/O Terminal]ウィンドウに表示されたログを記録したテキストファイルです。[I/O Terminal]ウィンドウの[File]メニューから[Save]を選択して作成します。

8.3　起動と終了

LogAna.exe

このアイコンをダブルクリックするとLogAnaが起動し、次のウィンドウを表示します。

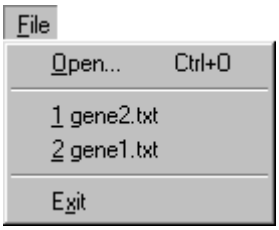


終了するには、[File]メニューから[Exit]を選択します。

8.4　メニューとツールバー

8.4.1 メニュー

[File]メニュー



[Open...] ([Ctrl]+[O])
入出力ログファイルを開きます。

ファイルリスト
ここにリストされているファイル名は最近開いたファイルです。ここからの選択でも、そのファイルを開くことができます。

[Exit]
LogAnaを終了します。

[View]メニュー

[Toolbar]
ツールバーの表示/非表示を切り替えます。

[Status Bar]
ステータスバーの表示/非表示を切り替えます。

[Help]メニュー

[About LogAna...]
LogAnaのバージョン情報を表示します。

8.4.2 ツールバーボタン

[Open]ボタン
入出力ログファイルを開きます。

[About]ボタン
LogAnaのバージョン情報を表示します。

8.5 操作方法

入出力ログファイルの作成からLogAnaによる波形表示までの手順を以下に示します。

1) シミュレータの入出力シミュレーションに使用するIOTファイルをテキストエディタで作成します。
作成方法については、3.13項(S1C63 Family)または4.13項(S1C88 Family)の"IOTファイル(.iot)"を参照してください。
なお、入出力波形を表示させたいポートは"watch"文で記述しておく必要があります。
例:
```
[General I/O]
  watch  P10,P11,P12,P13,P14,P15,P16,P17
                          :
```
ここに記載されていないポートの入出力は[I/O Terminal]ウィンドウのログに記録されませんので、LogAnaで表示することはできません。

2) シミュレータを起動し、ターゲットプログラムも含め、必要なファイルをロードします。

3) シミュレータの[LCD]ウィンドウにある[I/O]ボタンをクリックし、[I/O Terminal]ウィンドウを表示させます。

4) [I/O Terminal]ウィンドウの[File]メニューから[Open]を選択し、IOTファイルを読み込みます。

5) 必要に応じてブレークポイント/条件を設定しておきます。

6) リセット後、ターゲットプログラムを実行します。
シミュレータは指定されたポートの状態を監視し、入出力の変化をログにして[I/O Terminal]ウィンドウに出力します。

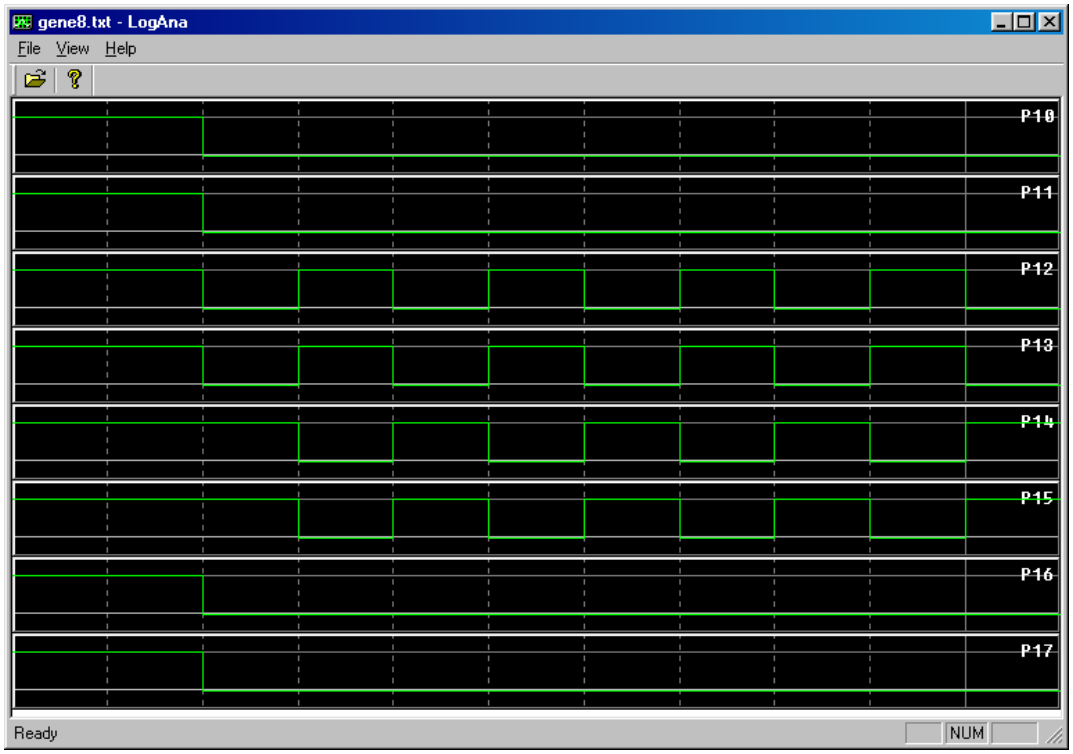
7) ブレーク条件によりプログラムが実行を中断するまで待ちます。または、[Key Break]ボタンで実行を中断させます。

8) [I/O Terminal]ウィンドウの[File]メニューから[Save]を選択し、ログをファイルに保存します。ファイル名には任意の拡張子を使用できます(通常は.txt)。
ログファイル例:
```
P10=1 P11=1 P12=1 P13=1 P14=1 P15=1 P16=1 P17=1

00:00:01.964 P10 >> 0  P11 >> 0  P12 >> 0  P13 >> 0  P16 >> 0  P17 >> 0
00:00:02.964 P12 >> 1  P13 >> 1  P14 >> 0  P15 >> 0
00:00:03.964 P12 >> 0  P13 >> 0  P14 >> 1  P15 >> 1
00:00:04.964 P12 >> 1  P13 >> 1  P14 >> 0  P15 >> 0
00:00:05.964 P12 >> 0  P13 >> 0  P14 >> 1  P15 >> 1
00:00:06.964 P12 >> 1  P13 >> 1  P14 >> 0  P15 >> 0
00:00:07.964 P12 >> 0  P13 >> 0  P14 >> 1  P15 >> 1
00:00:08.964 P12 >> 1  P13 >> 1  P14 >> 0  P15 >> 0
00:00:09.964 P12 >> 0  P13 >> 0  P14 >> 1  P15 >> 1
00:00:10.964 P12 >> 1  P13 >> 1  P14 >> 0  P15 >> 0
```

9) LogAnaを起動します。

10) [File]メニューから[Open...]を選択するか、[Open]ボタンでログファイルを読み込みます。
ログの内容に従って、波形が表示されます。



(前ページのログファイルを読み込んだ表示例)

ログには、ポートの状態変化が1ms単位で記録されています。これは、ターゲットが動作クロック(OSC1、OSC3)を切り換えても変わりません。
したがって、表示される波形の解像度も1msです。

なお、LogAnaは複数のログの読み込みには対応していません。同時に参照したい場合は、LogAnaを複数起動させる必要があります。

8.6　注意事項

最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_utility_J.txtを参照してください。

9　ROMデータ設定ユーティリティ

9.1　概要

ROMデータ設定ユーティリティ(Rom88.exe、以下Rom88と記述)は、バイナリ形式のROMイメージデータをS1C88 Familyシミュレータ(Sim88)にロードします。また、Sim88の[Dump]ウィンドウでパッチをあてたデータをバイナリ形式のROMイメージで保存することもできます。

注: Rom88はS1C88 Family専用のユーティリティです。S1C63 Familyには使用できません。
　　動作OS環境については、S5U1C88000QのReadme_J.txtを参照してください。

9.2　入出力ファイル

図9.2.1にRom88の入出力ファイルを示します。

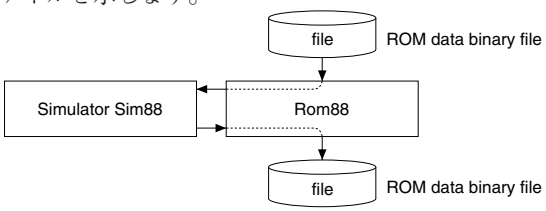


図9.2.1　Rom88の入出力ファイル

● ROMデータバイナリファイル
　S1C88xxx用のROMイメージデータです。

9.3　使用方法

Rom88の使用方法を以下に示します。

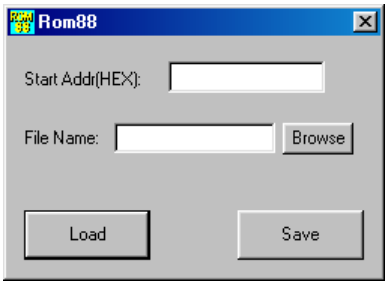
● ROMイメージデータのロード

1）!!重要!! 必ずシミュレータSim88を先に起動し、パラメータファイル等の必要なファイルを読み込んでおきます。

2）次のアイコンをダブルクリックしてRom88を起動します。

Rom88.exe

[Rom88]ダイアログボックスが表示されます。



3）[Start Addr(HEX)]にロードするメモリの先頭アドレスを16進数で入力します。

4）[Browse]ボタンでファイル選択ダイアログボックスを表示させ、ロードするファイルを選択します。あるいは[File Name]ボックスに直接入力します。

5）[Load]ボタンをクリックします。

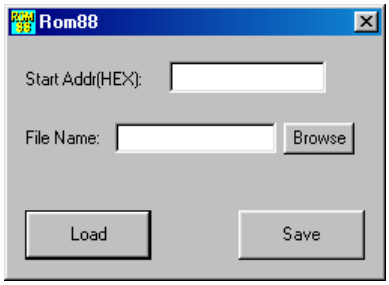
以上で、指定ファイルのデータがシミュレータ内のメモリエリアに読み込まれ、Rom88は終了します。

● ROMイメージデータのセーブ

1）上記の手順でROMイメージデータをロードします。

2）シミュレータ上でROM領域のデータに対して必要な編集を行います。

3）Rom88を起動します。
[Rom88]ダイアログボックスが表示されます。



4）[Start Addr(HEX)]に保存するメモリの先頭アドレスを16進数で入力します。

5）[Browse]ボタンでファイル選択ダイアログボックスを表示させ、保存先ファイルを選択します。あるいは[File Name]ボックスに直接入力します。
保存先には既存のファイルを選択してください。

6）[Save]ボタンをクリックします。

以上で、指定アドレスからデータがファイルに書き出され、Rom88は終了します。

9.4　注意事項

Rom88は、指定されたバイナリファイルの内容を指定アドレスからすべて書き込みます。ロードするROMイメージデータのサイズがパラメータファイルで指定された領域を超えないように注意してください。
最新バージョンの制限事項やサポート機能、バグ情報についてはS5U1C88000Qのrel_utility_J.txtを参照してください。

セイコーエプソン株式会社

半導体事業部　IC 営業部

<IC 国内営業グループ>

東京　〒191-8501 東京都日野市日野421-8
TEL（042）587-5313（直通）　FAX（042）587-5116

大阪　〒541-0059 大阪市中央区博労町3-5-1 エプソン大阪ビル15F
TEL（06）6120-6000（代表）　FAX（06）6120-6100

ドキュメントコード：411392102
2001 年 10 月 作成Ⓛ B
2009 年 8 月 改訂Ⓗ