

CMOS 8-BIT SINGLE CHIP MICROCOMPUTER **E0C88 Family**

***SIMULATOR FOR E0C88 FAMILY
OPERATION MANUAL***



NOTICE

No part of this material may be reproduced or duplicated in any form or by any means without the written permission of Seiko Epson. Seiko Epson reserves the right to make changes to this material without notice. Seiko Epson does not assume any liability of any kind arising out of any inaccuracies contained in this material or due to its application or use in any product or circuit and, further, there is no representation that this material is applicable to products requiring high level reliability, such as medical products. Moreover, no license to any intellectual property rights is granted by implication or otherwise, and there is no representation or warranty that anything made in accordance with this material will be free from any patent or copyright infringement of a third party. This material or portions thereof may contain technology or the subject relating to strategic products under the control of the Foreign Exchange and Foreign Trade Law of Japan and may require an export license from the Ministry of International Trade and Industry or other approval from another government agency.

Windows95, Windows98 and Windows NT are registered trademarks of Microsoft Corporation, U.S.A.
PC/AT and IBM are registered trademarks of International Business Machines Corporation, U.S.A.
All other product names mentioned herein are trademarks and/or registered trademarks of their respective owners.

CONTENTS

1	OVERVIEW OF THE E0C88 FAMILY SIMULATOR PACKAGE	1
1.1	Overview	1
1.2	Package Contents	1
1.3	Overview of Software Development Flow and Tools	2
2	INSTALLATION	4
2.1	Operating Environment	4
2.2	Installation Method	4
3	OVERVIEW OF SIMULATION FUNCTIONS AND OPERATIONS	5
4	LCD PANEL CUSTOMIZING UTILITY	7
4.1	Overview	7
4.2	Input/Output Files	7
4.3	Starting and Exiting	7
4.4	Window	8
4.5	Menus and Toolbar	9
4.5.1	Menus	9
4.5.2	Toolbar Buttons	11
4.6	Creating LCD Panel Data	12
4.6.1	Creating a New Panel and Setting the Panel Size	12
4.6.2	Creating Icons	13
4.6.3	Creating a Dot (pixel) Matrix	17
5	DEBUGGER	19
5.1	Features	19
5.2	Input/Output Files	19
5.3	Starting and Terminating the Debugger	20
5.4	Windows	21
5.4.1	Basic Structure of Window	21
5.4.2	[Command] Window	22
5.4.3	[LCD] Window	23
5.4.4	[Disassemble] Window	25
5.4.5	[Dump] Window	26
5.4.6	[Register] Window	27
5.4.7	[Symbol] Window	27
5.4.8	[Trace] Window	28
5.5	Menu	29
5.6	Tool Bar	32
5.7	Method for Executing Commands	33
5.7.1	Entering Commands from Keyboard	33
5.7.2	Executing from Menu or Tool Bar	35
5.7.3	Executing from a Command File	36
5.7.4	Log File	37
5.8	Debug Functions	38
5.8.1	Loading Files	38
5.8.2	Displaying and Modifying Program, Data, and Register	39
5.8.3	Executing Program	41
5.8.4	Break Functions	44
5.8.5	Trace Functions	47
5.8.6	Coverage	48

CONTENTS

5.9 *Command List* 49

5.10 *Component Mapping File (.cmp)* 50

5.11 *Port Setting File (.prt)* 52

5.12 *Simulator Project File (.spj)* 54

5.13 *Restrictions* 55

5.14 *Debugger Messages* 56

6 BITMAP UTILITY **57**

6.1 *Overview* 57

6.2 *Input/Output Files* 57

6.3 *Starting and Exiting* 57

6.4 *Window* 58

6.5 *Menus and Toolbar* 60

 6.5.1 *Menus* 60

 6.5.2 *Standard Toolbar Buttons* 62

 6.5.3 *Bitmap Edit Toolbar Buttons* 63

6.6 *Creating Bitmap Data* 64

 6.6.1 *New Data Wizard* 64

 6.6.2 *Creating Bitmap Images* 69

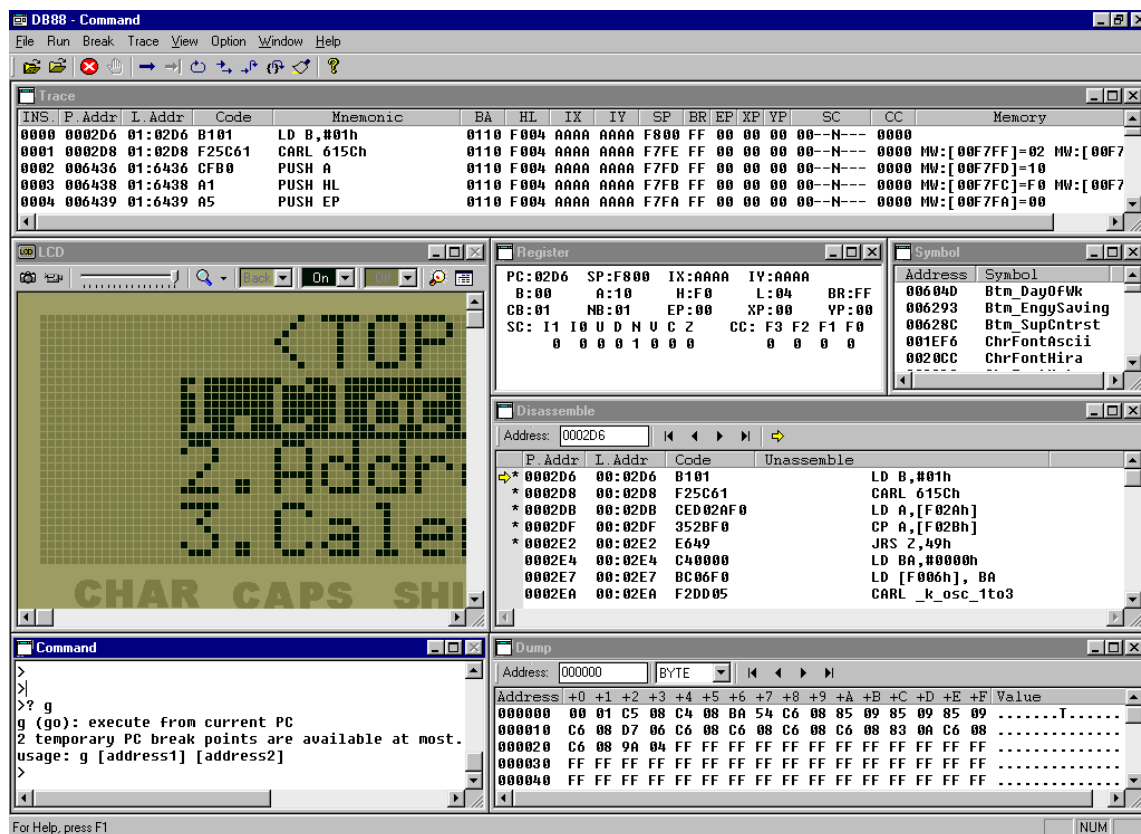
 6.6.3 *Editing Functions* 71

6.7 *Assembly Source File* 77

1 OVERVIEW OF THE E0C88 FAMILY SIMULATOR PACKAGE

1.1 Overview

The "Simulator for the E0C88" is a development tool for the E0C88 Family of 8-bit single-chip microcomputers. The debugger (simulator) included in this package allows you to debug software created with the E0C88 assembler using just a PC, without an in-circuit emulator (ICE) or other dedicated hardware. In addition to providing general debugger functions, the debugger simulates push-buttons or a key matrix that use I/O ports and LCD displays. The package includes utilities for creating bitmap and LCD panel data.



1.2 Package Contents

The E0C88 Family Simulator Package includes the following items. When unpacking, check that all items are present.

- (1) Tool disk (CD-ROM) 1

The official release version will also include a user's manual and warranty card.

Note: As a beta or prerelease version, this package may contain bugs. Users assume all responsibility for any problems arising from use of this version of the package.

1.3 Overview of Software Development Flow and Tools

Figure 1.3.1 shows the typical software development flow for the E0C88 Family. The items in bold indicate tools and related files provided in this package.

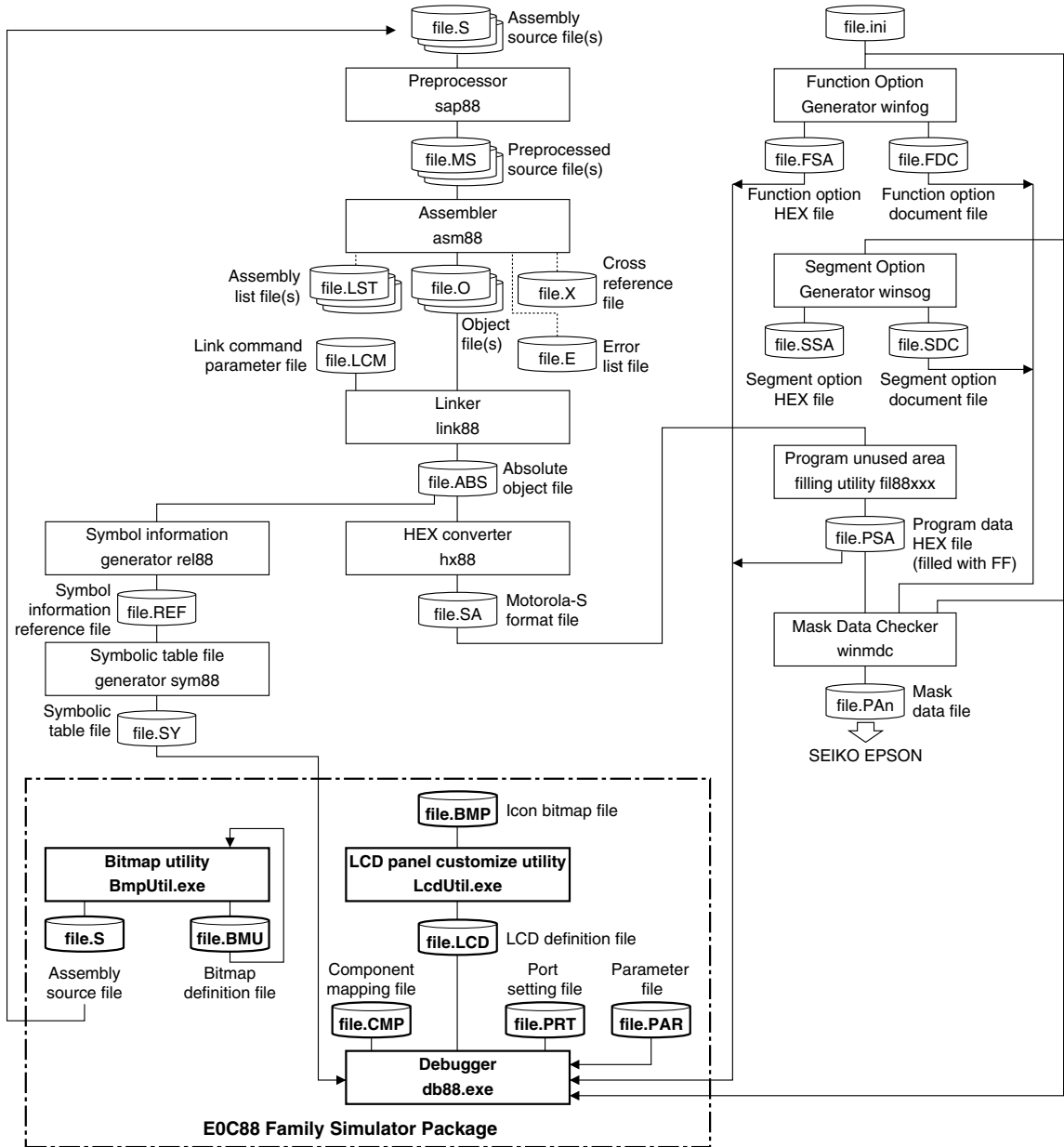


Fig. 1.3.1 Software development flow

Note: In addition to this package, software development for the E0C88 Family requires the E0C88 Family Structured Assembler package and the E0C88xxx Development Tool package available for each specific microcomputer model, as shown above.

The following provides an overview of the software tools included in the package.

LCD panel customize utility (LcdUtil.exe)

This utility creates a panel layout and COM/SEG port assignment data required by the debugger (db88.exe) to simulate a monochrome LCD panel display. Icons and other display objects are loaded from bitmap files (.bmp) and may be created with general-purpose paint software, enabling simulation of the actual product screen.

Debugger (db88.exe)

This software simulates device operations and debugging on a PC. Various functions are executed using commands entered from the keyboard or from files. Frequently-used break and single-step commands are registered to the toolbar to minimize repetitive keyboard tasks. Disassembled program code, register contents, and command execution results can be displayed in a multi-window screen to facilitate debugging. You can also simulate push-buttons or key matrix that use I/O ports and LCD displays.

Bitmap utility (BmpUtil.exe)

This utility creates bitmap image data (e.g., character data, sprite) for the dot matrix LCD display. The output data is created in the assembly source format with specified labels assigned to it to allow you to assemble data without modifications and link it to the program.

These tools run under Windows® 95/98 and Windows NT® 4.0.

2 *INSTALLATION*

This chapter describes the installation and operating environment for the tools included in the package.

2.1 *Operating Environment*

The following operating environment is required to use the E0C88 Family Simulator Package.

Personal computer

An IBM PC/AT or compatible, having a 200-MHz Pentium or faster CPU and 64 MB or more of RAM. Installation requires a CD-ROM drive.

- * For applications of about 1-MHz OSC3 clock to be run in real time, you must have a computer that contains a 400-MHz or faster Pentium CPU and more than 64 MB of RAM.

Display

Minimum 800 × 600 display resolution (SVGA)

Hard disk

Installation of the E0C88 Family Simulator Package requires a minimum of 15 MB of free hard disk space (more space is highly recommended).

System software

The package requires Microsoft® Windows® 95/Windows 98 or Windows NT® 4.0 (Service Pack 3 or higher), English or Japanese versions.

Other

In addition to this package, development of E0C88 Family applications requires the E0C88 Family Structured Assembler package and the E0C88xxx Development Tool package available for each specific microcomputer model.

2.2 *Installation Method*

Installing

Install the tools according to the following procedure.

- 1) Load the supplied CD-ROM into the drive.
- 2) Expand the 88sim.zip file into an appropriate folder.
- 3) If Internet Explorer is not set up, run 40Comupd.exe.

Notes:

- Expanding 88sim.zip copies the DLL files needed to run the tools into the same folder (88Sim) used by LcdUtil.exe and DB88.exe. If you are running Windows NT 4.0, delete MSVFM32.DLL from the DLL files.

- This procedure is used to install the beta release version. The formal release version will come with an installation utility.

Uninstalling

Delete all extracted files along with the folder.

3 OVERVIEW OF SIMULATION FUNCTIONS AND OPERATIONS

This chapter explains how the target application is simulated on a PC with the debugger (simulator) db88.

Supported MPUs

The current simulator version supports all applications using one of the following MPUs:

E0C88317, E0C88348, E0C88832, E0C88816, E0C88862, E0C88F360

Note: The formal release version will support additional MPUs. Please read the readme.txt file for confirmation of the most recent list of supported MPUs.

Target applications

This simulator is best suited to simulating applications that require key input and LCD display functions, such as watches, pocket calculators, electronic pocketbooks, and portable games.

OSC1 clock operation can be executed in real time. For OSC3 operation, real-time execution is possible in the range of 1 MHz to 2 MHz. (With PCs running on a 400-MHz Pentium or equivalent or faster and 64 MB RAM, real-time execution for approximately 1 MHz is possible.) However, for reasons involving instruction-level simulation accuracy, it is not possible to simulate high-accuracy timing tasks such as those for control systems.

Support for external devices is limited to ROM, RAM, LCD drivers (SED152A only), monochrome LCD panels, backlight, keys, and a key matrix.

Note: Simulation on a PC is subject to some other limitations. The formal release version will support a greater number of external devices. See Section 5.13, "Restrictions", and the readme.txt file.

Entering schematic information into the simulator

To set the information required to simulate the operation of external devices, load data from files.

ROM and RAM mapping

Use a parameter file (file.par) to set the addresses to which devices are mapped. For more information on creating this file, see the Development Tool Manual.

Mapping of external devices and the internally-generated clock frequency

Create a component mapping file (file.cmp) by writing addresses to which the LCD driver (SED152A), backlight, and a backlight control bit are mapped, then load the file into the simulator. This allows simulated control of these external devices. Use the component mapping file to set simulation conditions such as OSC1/OSC3 oscillation clock frequencies. For more information on creating this file, see Section 5.10, "Component Mapping File (.cmp)".

Entering key and key matrix specifications

Create a port setting file (.prt) containing a description of the relationship between key input ports and the I/O ports comprising the key matrix and target and PC keyboards, then load the file into the simulator. This allows simulated key input with the PC keyboard. For more information on creating this file, see Section 5.11, "Port Setting File (.prt)".

Entering LCD design

Create an LCD panel definition file (file.lcd) containing a record of LCD panel layout and SEG/COM port assignments, then load the file into the simulator. This allows simulated display on LCD panels. For more information on creating the LCD panel definition file, see Section 4, "LCD Panel Customizing Utility".

These files normally are loaded when the debugger (simulator) starts. For more information, see Section 5, "Debugger".

Loading and executing the target program

Use the debugger (simulator) if the lf command to load the target program and the g (or gr, s, n, se) command to run it. To simulate the LCD display or key input during program execution, use the debugger's [LCD] window.

LCD display: Displayed in the [LCD] window as on an actual LCD.

Key input: Activate the [LCD] window and type the appropriate key on the PC keyboard. You can also set and verify key input status in a dedicated window.

SVD evaluation: Manipulate the SVD slide bar on the [LCD] window to simulate SVD function.

4 LCD PANEL CUSTOMIZING UTILITY

4.1 Overview

The LCD Panel Customizing Utility (LcdUtil.exe, hereafter LcdUtil) creates the panel layout and COM/SEG port assignment data needed to simulate monochrome LCD panel displays with the debugger (db88.exe). Since they are loaded from bitmap files (.bmp), you can create icons or other display objects using general-purpose paint software to simulate the end-product screen. This utility is a device-independent tool; LCD driver settings that differ between microcomputer types are made using a device information definition file. Note that this file can be used not just for the internal LCD driver of the E0C88xxx chip, but for external LCD drivers from Seiko Epson.

4.2 Input/Output Files

Figure 4.2.1 shows the input/output files for LcdUtil.

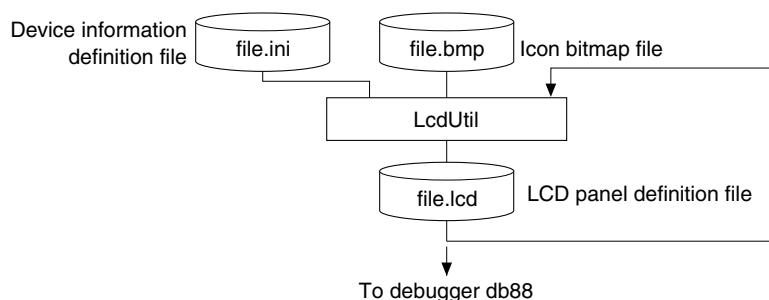


Fig. 4.2.1 LcdUtil input/output files

Device information definition file (file_name.ini)

This file contains information on LCD pins for an E0C88xxx or an external LCD driver. Use only files supplied by Seiko Epson. This file is compatible only with the model indicated by the file name. Do not attempt to modify the contents of the file or use the file for any other model.

Bitmap file (file_name.bmp)

Graphic objects representing icons are loaded from files in this format (monochrome bitmap). After loading multiple parts individually, edit the layout in LcdUtil.

LCD panel definition file (file_name.lcd)

This file contains information on the panel layout bitmap and COM/SEG pin assignments. Load this file into the debugger when simulating LCD display.

4.3 Starting and Exiting

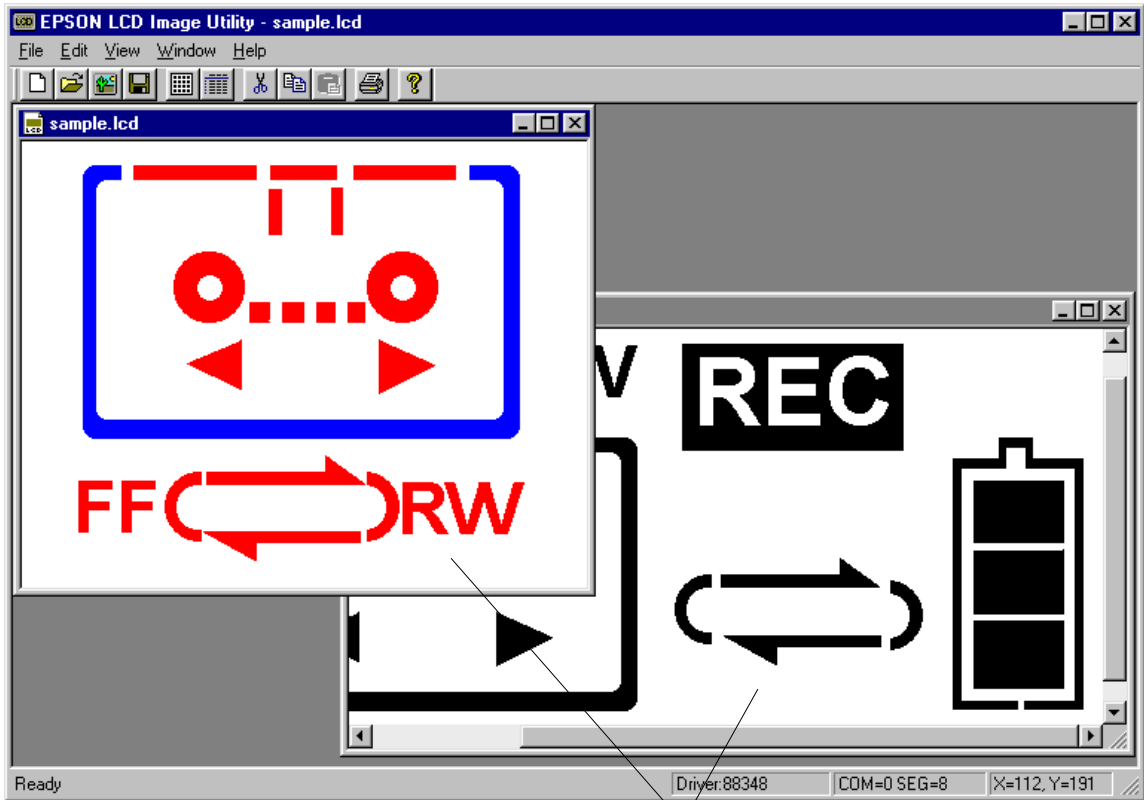


Double-click this icon to start LcdUtil.

LcdUtil.exe

To exit LcdUtil, select [Exit] from the [File] menu.

4.4 Window



Panel edit window

Panel edit window

When you open a bitmap file (.bmp) or LCD panel definition file (.lcd), the file is displayed in this window. In this window, you can lay out the panel or assign COM/SEG ports. You can open multiple instances of this window and move data between the two windows through Drag & Drop.

Basic window operations

Closing and activating the window

To close the window, click the [Close] button on the window. The windows being opened are listed on the [Window] menu. Select a window name from this menu to activate the window, or click anywhere in the desired window. You can also use the [Ctrl] + [Tab] keys to switch the active window from one window to the next.

Resizing and moving the window

Drag the window boundary to resize any window. The [Minimize] and [Maximize] buttons work in the same way as for general Windows applications. You can reposition any window within the display by dragging the title bar, but the panel edit window can only be resized and moved within the application window. If the image displayed in a window is extends beyond the window in any direction, a scroll bar appears.

Other

To align open windows, select [Cascade] or [Tile] from the [Window] menu.

4.5 Menus and Toolbar

4.5.1 Menus

[File] menu

File	
<u>N</u> ew	Ctrl+N
<u>O</u> pen...	Ctrl+O
Open Bitmap File...	
<u>C</u> lose	
<u>S</u> ave	Ctrl+S
Save <u>A</u> s...	
<hr/>	
<u>P</u> rint...	Ctrl+P
Print Preview	
Print Setup...	
<hr/>	
1 88348dmt.lcd	
2 C:\E0C88\...\sample.lcd	
3 C:\E0C88\...\icon.bmp	
<hr/>	
<u>E</u> xit	

[New] ([Ctrl] + [N])

Opens a new panel edit window.

[Open...] ([Ctrl] + [O])

Opens an LCD panel definition file (.lcd).

[Open Bitmap File...]

Opens a bitmap file (.lcd).

[Save] ([Ctrl] + [S])

Saves the contents of the active panel edit window to the LCD panel definition file (.lcd), overwriting the previous file.

[Save As...]

Saves the contents of the active panel edit window to the LCD panel definition file (.lcd) under another name.

[Print...] ([Ctrl] + [P])

Sends the bitmap of the active panel edit window to a printer.

[Print Preview]

Displays an on-screen print image of the active panel edit window.

[Print Setup...]

Brings up a dialog box to select paper size or printer.

File list

Listed here are recently opened files. You may open any of these files by selecting its filename.

[Exit]

Exits LcdUtil.

[Edit] menu

Edit	
<u>C</u> ut	Ctrl+X
<u>C</u> opy	Ctrl+C
<u>P</u> aste	Ctrl+V
<hr/>	
Insert dot <u>m</u> atrix	Ctrl+M
<u>I</u> con List	Ctrl+I
Resize LCD	
<hr/>	
Group Icon	
Release Group	

[Cut] ([Ctrl] + [X])

Cuts and copies a selected portion of the panel edit window to the clipboard.

[Copy] ([Ctrl] + [C])

Copies a selected portion of the panel edit window to the clipboard.

[Paste] ([Ctrl] + [V])

Pastes the portion copied to the clipboard into the upper left corner of the panel edit window.

[Insert dot matrix] ([Ctrl] + [M])

Inserts a dot matrix image into the panel edit window. Specify sizes and other parameters using the dialog box.

[Icon List] ([Ctrl] + [I])

Displays a list of icons currently in the active panel edit window. You can also perform COM/SEG assignments from the dialog box open here.

[Resize LCD]

Sets the size of the LCD panel. When you open a new panel edit window, the default size is 640 × 480.

[Group Icon]

Groups multiple icons into one icon.

[Release Group]

Ungroups the grouped icon into individual icons.

[View] menu



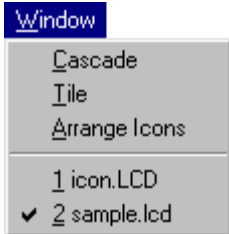
[Toolbar]

Alternately shows or hides the toolbar as you click this menu command.

[Status Bar]

Alternately shows or hides the status bar as you click this menu command.

[Window] menu



[Cascade]

Aligns the open panel edit windows, overlapping them diagonally.

[Tile]

Aligns the open panel edit windows by placing them side by side in rows.

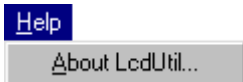
[Arrange Icons]

Lines up the minimized panel edit window icons at the bottom of the application window area.

Window list

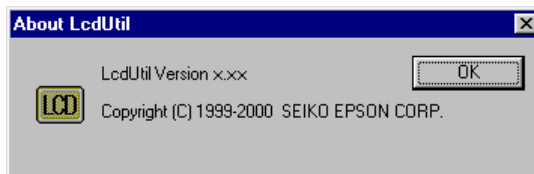
Currently open panel edit window names are listed here. Select a window name here to make the corresponding panel edit window active.

[Help] menu



[About LcdUtil...]

Displays version information for LcdUtil.



4.5.2 Toolbar Buttons



[New] button

Opens a new panel edit window.



[Open] button

Opens an LCD panel definition file (.lcd).



[Bitmap] button

Opens a bitmap file (.bmp).



[Save] button

Saves the contents of the active panel edit window to the LCD panel definition file (.lcd), overwriting the previous file.



[Dot Matrix] button

Inserts a dot matrix image into the panel edit window. Specify sizes and other parameters using the dialog box.



[Icon List] button

Displays a list of icons present in the active panel edit window. You can also perform COM/SEG assignments from the dialog box opened here.



[Cut] button

Cuts and copies a selected portion from the panel edit window to the clipboard.



[Copy] button

Copies a selected portion from the panel edit window to the clipboard.



[Paste] button

Pastes the portion copied to the clipboard into the upper left corner of the panel edit window.



[Print] button

Sends the bitmap of the active panel edit window to a printer.



[About] button

Displays version information for LcdUtil.

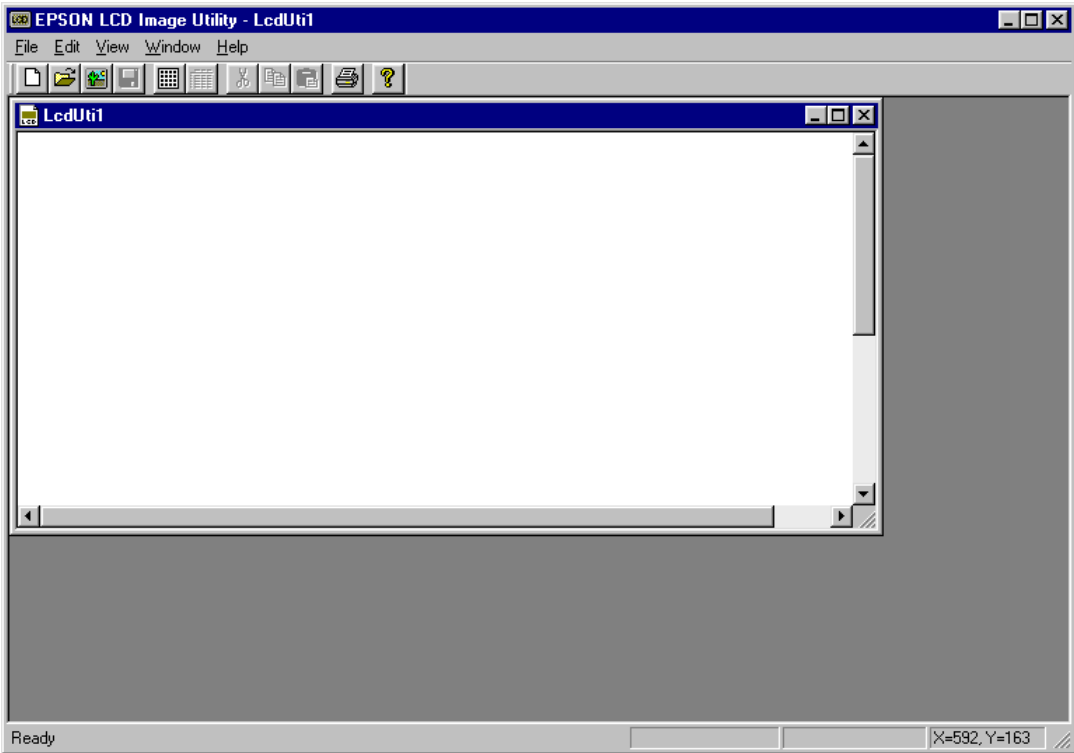
4.6 Creating LCD Panel Data

The panel edit window allows you to lay out icons and dot (pixel) matrix blocks just as they appear in the actual panel. COM/SEG ports can also be assigned from the panel edit window, as explained below.

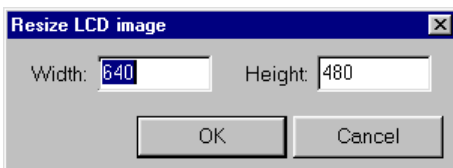
4.6.1 Creating a New Panel and Setting the Panel Size

To create an LCD panel definition file by placing a dot (pixel) matrix and icons on a blank panel edit window, you must first set up a new panel according to the following procedure.

- 1) Select [New] from the [File] menu (or click the [New] button). A blank panel edit window opens.



- 2) Select [Resize LCD] from the [Edit] menu. A [Resize LCD image] dialog box appears.



Enter a desired panel size and click [OK]. The value specified here represents the size of the LCD panel simulated with the debugger and must be specified by the number of pixels on the computer screen. The default size when creating a new panel is 640 × 480 pixels. You also need to set sizes for icon bitmaps and dot matrix to be placed.

You can change the panel size after finishing placement of icons or dot matrix.

A space for the panel has now been created. Here, you can place a dot matrix or the bitmaps of icons you've created separately. When creating an LCD panel definition file based on icons loaded from a bitmap file, the procedure given above for creating a new panel is unnecessary.

4.6.2 Creating Icons

This section explains how to assign icons to ports.

Creating bitmap data

Prepare bitmaps of icons using a scanner, or by creating with general paint software. Consider the following when preparing bitmaps.

Number of colors and file format

Create icons in black on white background and save the created icons in monochrome bitmap format (.bmp). Although LcdUtil can load bitmaps in up to 65,535 colors, this data is converted into black & white two-level data when loaded.

Size

LcdUtil allows you to specify the size of the LCD panel to be simulated in number of on-screen pixels. ([Resize LCD] on the [Edit] menu.) Determine the sizes of individual icons to suit the simulated panel size as you create them. Icon sizes cannot be adjusted in LcdUtil.

Pixel size can be specified in relation to pixels on the computer screen. When simulating a panel configured with a mixture of icons and dot matrix, consider the size of the matrix as you determine panel and icon size. The panel size can be enlarged or reduced in some steps between 25% and 200% in the debugger.

If you only need to verify the operation of your program, you are not required to create precise, faithful copies of icon bitmaps. This capability is merely provided to enable highly-accurate display simulation for design evaluation of the LCD panel.

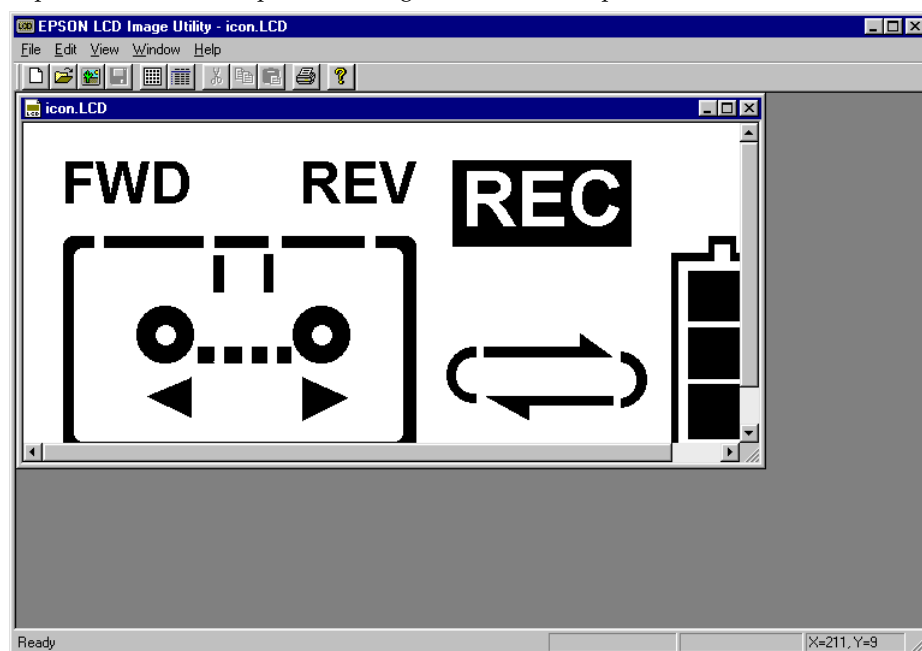
Number of files

All of the icons may be saved into a single file, or in multiple separate files. Due to the limited editing capabilities of LcdUtil, we recommend saving icons to a single file when creating bitmaps. We also recommend determining their layout before you load the file.

Loading bitmap data

Start LcdUtil and select [Open Bitmap File...] from the [File] menu (or click the [Bitmap] button). A standard [Open] dialog box appears. Select and load the created bitmap file.

A panel edit window opens, showing the loaded bitmap.



Editing icon layouts

From the loaded bitmap, areas comprised of consecutive black pixels are cut out as parts and recognized as icons. You can perform the following operations on these parts.

Selection

Click and select a part to change its highlight color to blue. The operations described below and COM/SEG port assignments are applied to the part in this selected state.

Move

Drag the part. You can move it into another panel edit window.

Copy

Copy and paste a part using the menu or toolbar buttons.

Delete

Use the [delete] key to delete a part.

Group

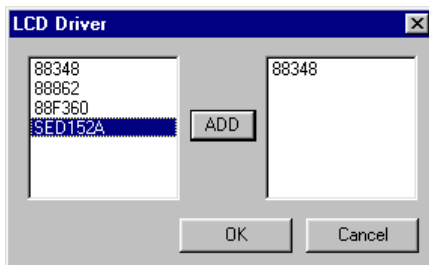
Hold down the [Ctrl] key while clicking one part and another. This allows selection of multiple parts. Then select [Group Icon] from the [Edit] menu. The parts are now grouped and may be treated as a single icon. To ungroup the grouped icon, select [Release Group] from the [Edit] menu.

Note: LcdUtil is provided with simple editing function. We recommend saving icons to a single file when creating bitmaps, as well as determining their layout before loading the file.

Port assignments

You can perform COM/SEG port assignments in the panel edit window. The procedure is given below.

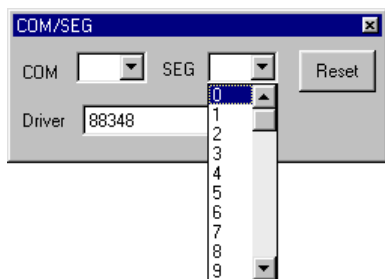
- 1) Double-click the icon to which you want to assign a port.
- 2) An [LCD Driver] dialog box appears.



From the list, select the model for which your application is being developed and click the [ADD] button. To use one of the external LCD drivers on the list, select and add it in the same way, then click the [OK] button. To cancel, click [Cancel].

* This dialog box appears when you assign ports for the first time. It is not displayed once you select a driver. Since the driver cannot be changed after ports are assigned, choose a driver carefully.

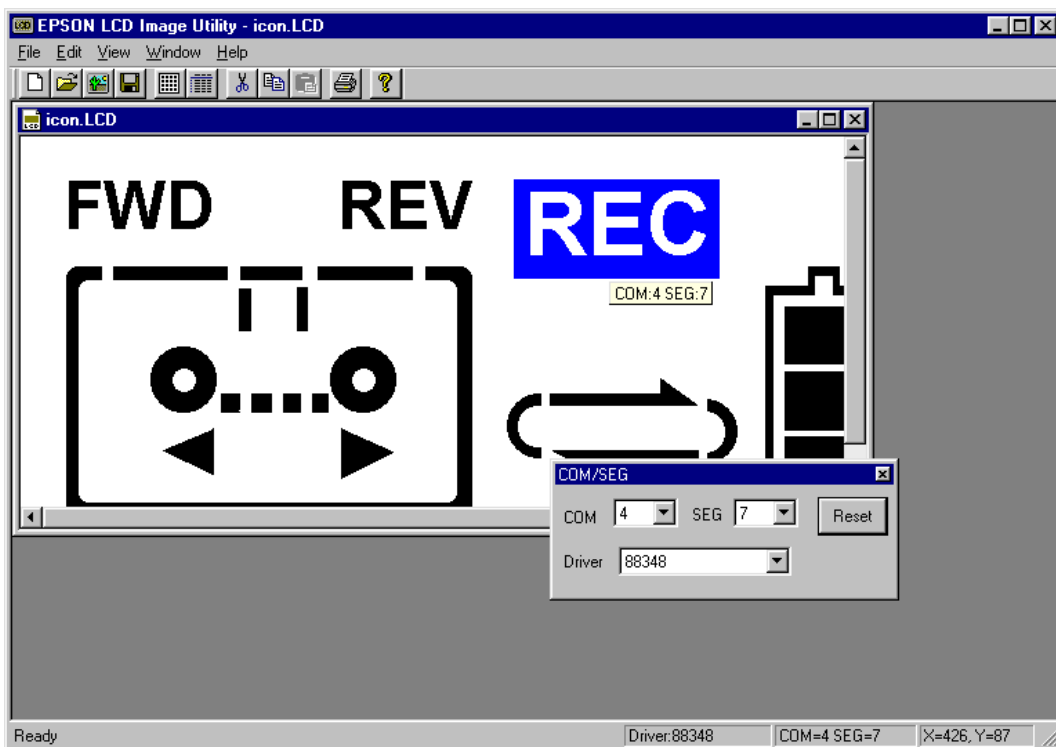
- 3) A [COM/SEG] dialog box appears.



Select a COM port and SEG port number from the pull-down list. When using an external LCD driver, you must also select [Driver]. Click the [Reset] button to clear the selected COM/SEG ports. The selected [Driver] is also restored to the default driver.

After selecting COM and SEG ports, click anywhere on the panel edit window or close this dialog box before assigning the next icon.

The icons with assigned ports are displayed in red. Moving the mouse pointer over any of those icons displays its assigned port and driver information in the status bar. The port information is also shown at the mouse cursor position.



For icons comprised of multiple parts, assign the same COM/SEG port to each part, or group them into a single icon before assigning COM/SEG ports. This allows selection of all parts with a single click.

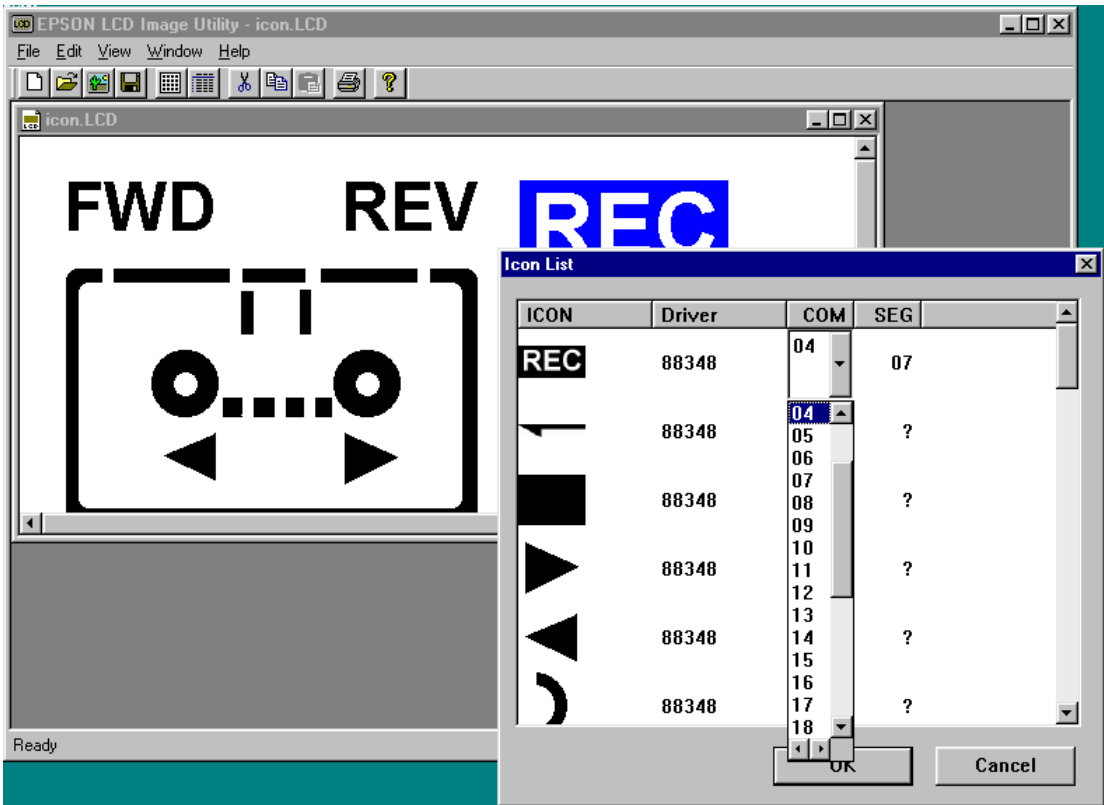
This dialog box is also displayed when you double-click an icon with assigned ports. This allows you to correct port assignments at a later time.

Note: When you copy and paste an icon with ports already assigned, or move such an icon into another panel edit window, its port assignment information is cleared. When editing layout of a panel on another panel edit window, move the icons for it to that window before assigning ports. If you group parts with different assigned ports into a single icon, the port assignment for the last part selected for grouping is applied to all parts of the grouped icon. The previous settings will not be restored when you ungroup the icon.

- 4) Select [Save] or [Save As...] from the [File] menu to save the contents set as an LCD panel definition file.

Displaying an icon list

Select [Icon List] from the [Edit] menu (or click the [Icon List] button) to display a list of icons within the currently active panel edit window.

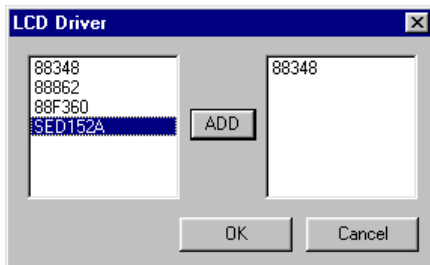


Click and select an icon on the list. The icon in the panel edit window turns blue to indicate the icon's position in the panel edit window of the selected icon. When you click a driver or COM/SEG number on this list, a pull-down list is displayed to enable you to change the set contents.

4.6.3 Creating a Dot (pixel) Matrix

This section explains how to create a dot matrix and how to assign ports to a dot matrix.

- 1) Create a new panel according to the procedure described in Section 4.6.1, or open a bitmap file containing icons or an LCD panel definition file. For the panel size, first calculate the size of the dot matrix, then determine a panel size greater than this value.
- 2) Select [Insert dot matrix] from the [Edit] menu (or click the [Dot Matrix] button).
- 3) The [LCD Driver] dialog box appears.



Select the model for which you are developing your application from the list and click the [ADD] button. To use one of external LCD drivers on the list, select and add it in the same way and click the [OK] button. To cancel, click [Cancel].

* This dialog box appears when no LCD drivers have been selected. It is not displayed when icons already have assigned ports.

- 4) A [Dot (pixel) Matrix] dialog box appears.

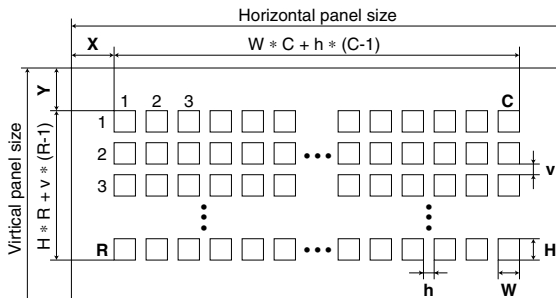
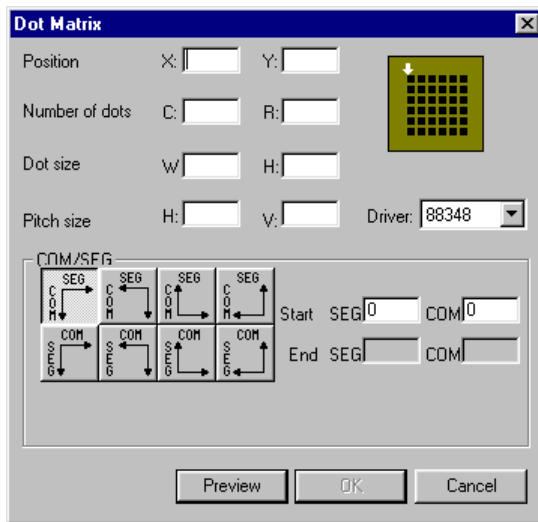


Fig. 4.6.3.1 Setting up a dot matrix

4 LCD PANEL CUSTOMIZING UTILITY

Set the following in this dialog box.

Position

Specify the distance from the upper left edge of the panel to the upper left edge of the dot matrix by the number of pixels on the computer screen. (X and Y in the diagram)

Number of dots

Specify the number of dots along the horizontal and the vertical directions of the dot matrix. (C and R in the diagram)

Dot size

Specify dot size for the dot matrix by number of pixels on the computer. (W and H in the diagram)

Pitch size

Specify the intervals at which individual dots are displayed by number of pixels on the computer. (h and v in the diagram)

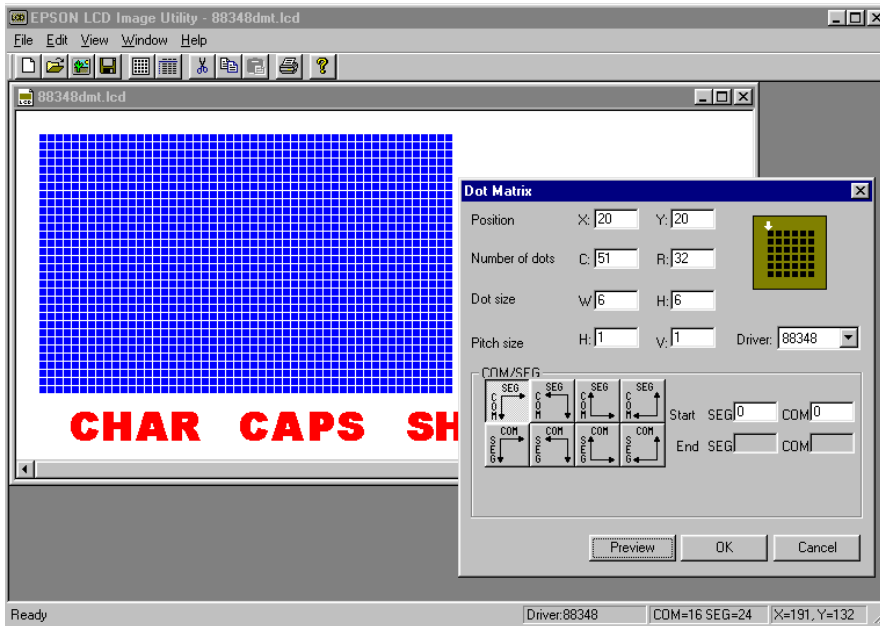
Driver

Select an LCD driver. If you have multiple LCD drivers driving a single dot matrix, create a dot matrix for each driver.

COM/SEG

Use one of the eight buttons to select a starting point for assignment. In the [Start] text box, set the COM/SEG port numbers assigned to that dot. Other dots are assigned automatically, according to the button you select.

Click the [Preview] button after setting each of the above items. A dot matrix of the specified size is displayed at the specified position in the panel edit window. This preview is shown to allow you to verify the contents you've set. The matrix is not actually created until you click the [OK] button.



The created dot matrix is shown in red. Move the mouse pointer over any of the dots to display information on assigned ports and driver in the status bar. The port information is also displayed at the mouse cursor position.

When you double-click the dot matrix, a [Dot Matrix] dialog box appears to allow you to change the settings at a later time. You can also change the position of the dot matrix by dragging it to a desired position.

Note: Copying and pasting a dot matrix deletes its port assignment information, although information on position and size is retained.

5 DEBUGGER

This chapter describes how to use the Debugger db88.

5.1 Features

The Debugger db88 is used to debug a program after reading an executable object file.

It has the following features and functions:

- Operations including LCD panel display can be simulated with a PC alone without any debugging hardware.
- Various data can be referenced at the same time using multiple windows.
- Frequently used commands can be executed from tool bars and menus using a mouse.
- Also available are disassembled code display and symbol display functions.
- Consecutive program execution and three types of single-stepping are possible.
- Three break functions are supported.
- Trace and coverage functions.
- An automatic command execution function using a command file.

5.2 Input/Output Files

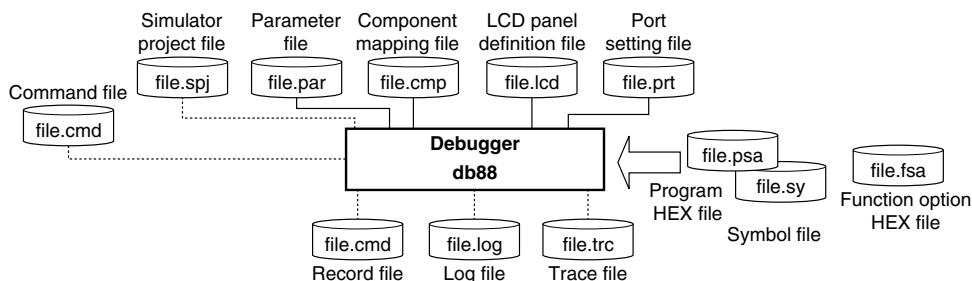


Fig. 5.2.1 Input/output files

Parameter file (file_name.par)

This text file contains memory information on each microcomputer model and is used to set the memory mapping information to the debugger. For the contents of this file, refer to the Development Tool Manual for each model.

Internal ROM data HEX file (file_name.psa)

This is the program file generated by the fil88xxx unused ROM area FF filling utility in Motorola S2 format file. The unused area of the built-in ROM has been filled with FFH and the system code is set to the system reserved area.

Function option HEX file (file_name.fsa)

This is the mask option setup file in Motorola S2 format that is generated by the function option generator.

Symbol information file (file_name.sy)

This is the symbol information file generated by the symbol table file generator. By preparing the file with the same name as the program file in the same directory as the program file, it will be automatically loaded at the same time the program is loaded. This file allows the debugger to display the symbols defined in the source.

Simulator project file (file_name.spj)

This file is used to specify a parameter file, LCD panel definition file, component mapping file and port setting file at the debugger start up. Enter file names using an editor to create this file. The debugger can be started up if this file does not exist by selecting the files from a dialog box.

LCD panel definition file (file_name.lcd)

This file includes an LCD panel layout bitmap and SEG/COM port allocation information. Create this file using the LCD panel customizing utility (LcdUtil).

Component mapping file (file_name.cmp)

This is the text file that sets the addresses where the external LCD driver and backlight are mapped.

Port setting file (file_name.prt)

This is the text file in which push keys, key-matrix configuration and the corresponding between the keys and ports are described.

Command file (file_name.cmd)

This text file contains a description of debug commands to be executed successively. By writing a series of frequently used commands in this file, the time and labor required for entering commands from the keyboard can be saved. The command described in the file are read and executed using the com command.

Log file (file_name.log)

This text file contains the executed commands and execution results. Output of this file can be controlled by the log command.

Record file (file_name.cmd)

This text file contains the executed commands. Output of this file can be controlled by the rec command. This command can be used as a command file.

Trace file (file_name.trc)

This text file contains the specified range of trace information. Output of this file can be controlled by the tf command.

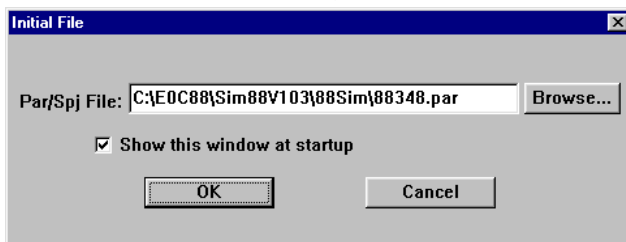
5.3 Starting and Terminating the Debugger



DB88.exe

Double-clicking this icon to start the debugger.

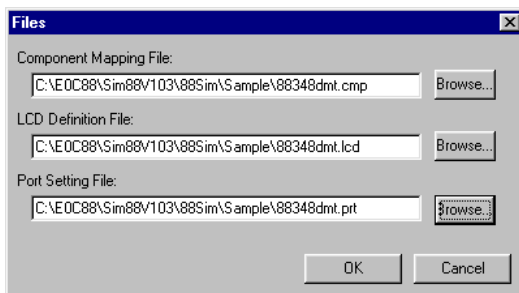
The dialog box shown below appears at the first time the debugger starts up. Enter the parameter file name or simulator project file name to the text box, or choose it using the [Browse...] button (see Section 5.12 for the simulator file).



If the [Show this window at startup] check box is deselected, this dialog box will not appear from the next startup of the debugger and the same file will be selected. After this, use the par command ([File | Load Parameter File...]) to redisplay this dialog box for changing the file.

When the check box is left on, the currently selected file name will appear in the text box at the next startup of the debugger allowing choose of the file by clicking the [OK] button only.

When a parameter file is selected, the dialog box shown below appears.



Enter the component mapping file name, LCD panel definition file name and port setting file name to the respective text boxes or choose them using the [Browse...] button. This dialog box will appear at the next start-up of the debugger even if the [Show this window at startup] is deselected. However, the selected file names will appear in the text boxes at the next startup of the debugger allowing choice of the files by clicking the [OK] button only.

This dialog box will not appear when a simulator project file is selected.

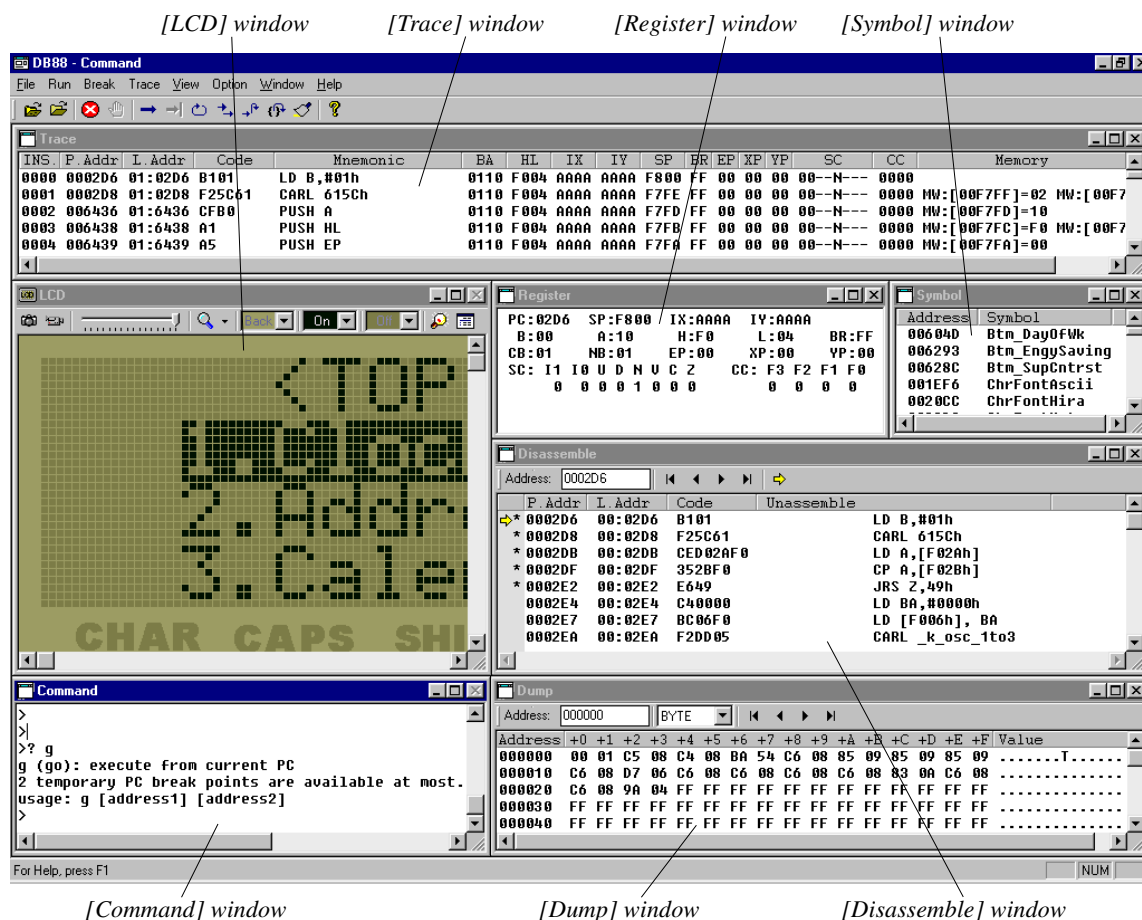
Select [Exit] from the [File] menu to terminate the debugger.

5.4 Windows

This section describes the types of windows used by the debugger.

5.4.1 Basic Structure of Window

The diagram below shows the window structure of the debugger.



Features common to all windows

(1) Open/close and activating a window

All windows except [Command] and [LCD] can be closed or opened.

To open a window, select the window name from the [View] menu. When a command is executed, the corresponding window opens if the command uses the window for displaying the executed results.

To close a window, click the [Close] box on the window.

The opened windows are listed in the [Window] menu. Selecting one from the list activates the selected window. It can also be done by simply clicking on an inactive window. Furthermore, pressing [Ctrl]+[Tab] switches the active window to the next open window.

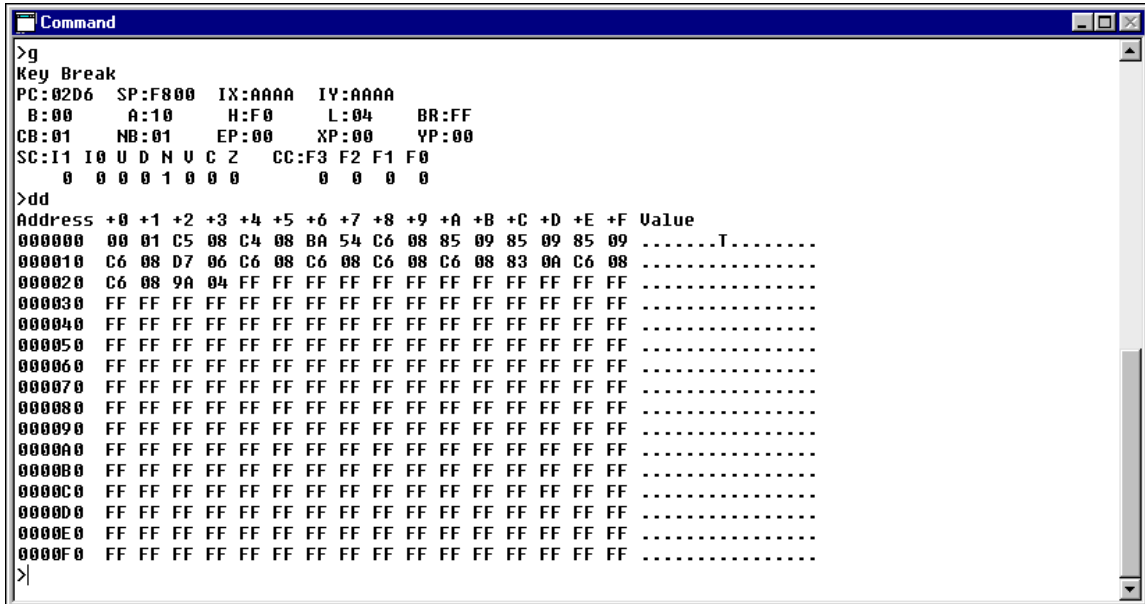
(2) Resizing and moving a window

Each window can be resized as needed by dragging the boundary of the window with the mouse. The [Minimize] and [Maximize] buttons work in the same way as in general Windows applications. Each window can be moved to the desired display position by dragging the window's title bar with the mouse. However, windows can only be resized and moved within the range of the application window.

(3) Other

The opened windows can be cascaded or tiled using the [Window] menu.

5.4.2 [Command] Window



The [Command] window is used to do the following:

(1) Entering debug commands

When the prompt ">" appears in the [Command] window, the system will accept a command entered from the keyboard.

(2) Displaying debug commands selected from menus or tool bar

When a command is executed by selecting the menu item or tool bar button, the executed command line is displayed in the [Command] window.

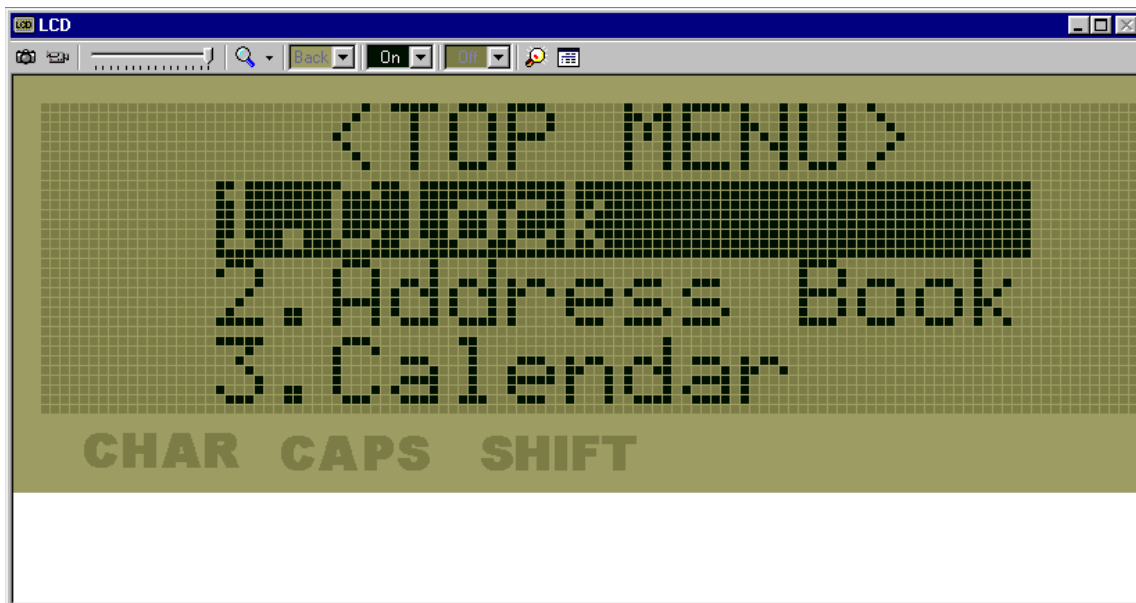
(3) Displaying command execution results

The [Command] window displays command execution results. However, some command execution results are displayed in the [Disassemble], [Dump], [Register], or [Trace] windows. The contents of these execution results are displayed when their corresponding windows are open. If the corresponding window is closed, the execution result is displayed in the [Command] window.

When writing to a log file, the content of the write data is displayed in the window. (Refer to the description for log command.)

Note: The [Command] window cannot be closed.

5.4.3 [LCD] Window

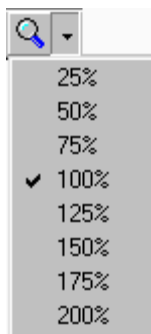


The [LCD] window has the following functions:

(1) LCD display simulation

Displays the LCD panel defined in the LCD panel definition file. The icons and dot matrix change their display status according to the program being executed.

Furthermore, the panel size and the color can be set using the following controls.



Scaling button & drop-down list

The panel size is magnified in 25% steps every time the button is clicked. If the size has reached 200%, the next click reduces the size to 25%. The drop-down list allows direct selection of the magnification rate.

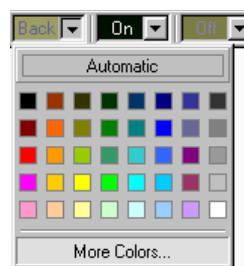
[Back], [On], [Off] drop-down lists

Set the LCD panel color.

[Back] Background color

[On] Dot color when it is on

[Off] Dot color when it is off



[Backlight] button

When this button is clicked, the dialog box shown below appears allowing registration of up to 4 backlight colors. The color can be adjusted using the RGB sliders and selecting the lower left check box turns the backlight on.



(2) Capturing the panel image**Camera button**

Clicking this button captures the LCD panel image at that point and then transfers it to the clipboard as a bitmap image. The captured image can be pasted to paint software to print and/or saved to a file.

Since this button is invalid during program execution, program execution must be broken at the desired image before it can be captured.

**Video button**

Clicking this button enables the panel image recording function (can be saved as an AVI file).

When starting the program execution in this status, two dialog boxes to specify a file name and a video compression format appear sequentially. Enter a file name and choose an available video compression format. The recording continues until the program execution is broken.

The generated file can be played back with the Windows standard medium player.

This button cannot be cancelled after it has been clicked once. To cancel the recording, click on the [Cancel] button in the file name input dialog box. The program will be executed, but the recording is cancelled.

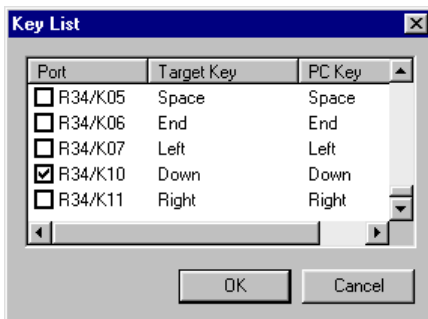
Note: The recording operation reduces the program execution speed.

(3) Key entry simulation

If the program being executed is waiting a key entry, key entry operation can be simulated using the PC keyboard after activating the [LCD] window.

The correspondence between PC keys and ports should be defined in the port setting file.

This definition list can be displayed using the [Key List] button.

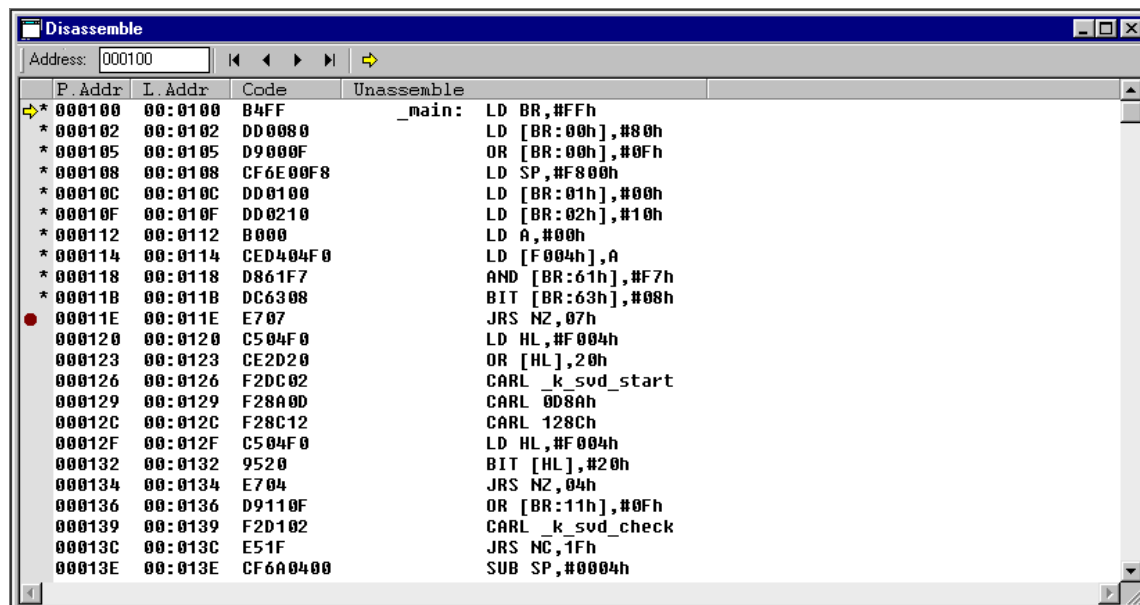


The displayed contents are port name, target key name and PC key name, respectively from the left. The correspondence between PC keys and target keys can be verified here. The check box on the left allows verification and setting the port status. The check mark indicates that the key connected to the port has been pressed (port is low level). The key entry status can be set even in break status. The key entry status is maintained during the next program execution until the user operates the key. This allows setting of the key entry status during single-step operation. The input port is fixed at low if the check-marked key is directly connected to the input port. In the case of a key matrix, the input port goes low only when the corresponding output port goes low.

(4) SVD simulation

This slider changes the detection level of the SVD to 16 steps.

5.4.4 [Disassemble] Window



The [Disassemble] window displays the contents of (1) to (3) listed below. This window also allows breakpoints to be set.

(1) Program code

The window displays the physical/logical addresses, codes, and disassembled contents.

Program display location can be changed by the following method as well as scrolling.

- Enter an address in the [Address] text box. Or specify an address using the u command.

The program is displayed from the selected address.



— Displays the beginning or end area of the memory.

— Displays one page before or after in the current window size.

— Displays the program from the current PC address.

* Updating of display

When a program is loaded and executed (g, gr, s, n, se, or rst command), or the memory contents are changed (de, df, or dm command), the display contents are updated. In this case the [Disassemble] window updates its display contents so that the current PC address can always be displayed.

(2) Current PC

The current PC (program counter) address is indicated by a yellow arrow at the beginning of the line.

(3) PC breakpoint

The address line where a breakpoint is set is indicated by a red ● mark at the beginning of the line.

(4) Coverage information

When the coverage function is turned on using the md command, the following program execution places an * at the beginning of the executed address line.

(5) Break setting at the cursor position

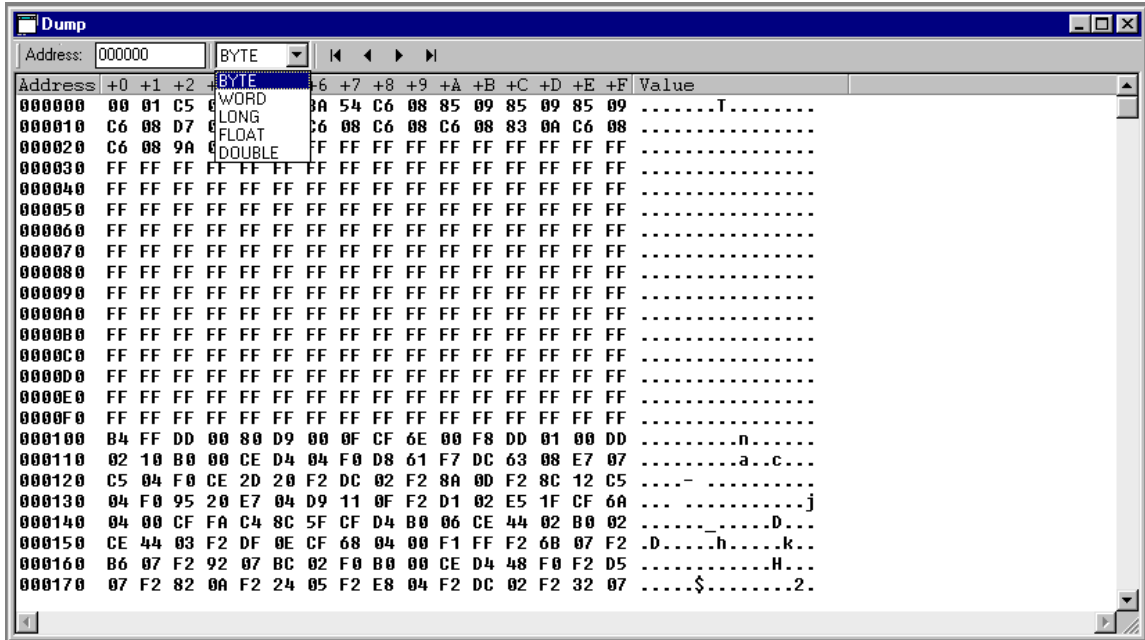


Place the cursor at an address line where a breakpoint is to be set. Then click on the [Break] button. A PC breakpoint will be set at that address. If the same is done at the address line where a PC breakpoint has been set, the breakpoint will be cleared. This function allows setting of two or more breakpoints.



If the [Go to Cursor] button is clicked, the program will execute beginning with the current PC position, and program execution breaks at the line where the cursor is located.

5.4.5 [Dump] Window



(1) Displaying data memory contents

The [Dump] window displays the memory dump results in hexadecimal numbers. Data is displayed in byte units by default. It can be changed to another size using the pull-down box. Memory display location can be changed by the following method as well as scrolling.

- Enter an address in the [Address] text box. Or specify an address using the dd command. Data is displayed from the selected address.



— Displays the beginning or end area of the memory.
 — Displays one page before or after in the current window size.

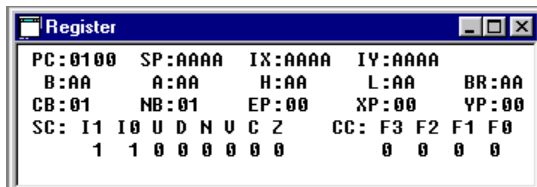
* Updating of display

The display contents of the [Dump] window are updated automatically when memory contents are modified with a command (de, df, or dm command), or by direct modification. After executing the program (g, gr, s, n, se, or rst command), the display contents are also updated. To refresh the [Dump] window manually, execute the dd command or click the vertical scroll bar.

(2) Direct modification of data memory contents

The [Dump] window allows direct modification of data memory contents. To modify data on the [Dump] window, place the cursor at the front of the data to be modified or double click the data, and then type a hexadecimal character (0–9, a–f). Data in the address will be modified with the entered number and the cursor will move to the next address. This allows successive modification of a series of addresses.

5.4.6 [Register] Window



(1) Displaying register contents

The [Register] window displays the contents of the E0C88 CPU registers and condition flags.

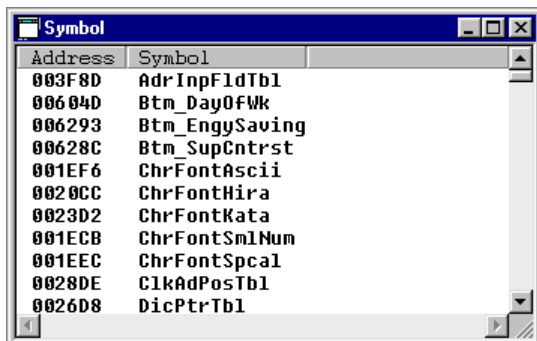
* Updating the display

The display is updated when registers are dumped (rd command), when register data is modified (rs command), when the CPU is reset (rst command), or after program execution (g, gr, s, se, or n command) is completed.

(2) Direct modification of register contents

The [Register] window allows direct modification of register contents. To modify data on the [Register] window, select (highlight) the data to be modified and type a hexadecimal number (0–9, a–f), then press [Enter]. The register data will be modified with the entered number.

5.4.7 [Symbol] Window



The [Symbol] window can display the symbol list, if a symbol file (.sy) is loaded.

Symbols are listed in alphabetical order by default. It can be changed to address order using the "sy /a" command.

- * The symbol file is automatically loaded simultaneously with the target program. However, it must be the same name (extension is .sy) and be located in the same directory as the target program file.

5.4.8 [Trace] Window

Trace																
INS	P. Addr	L. Addr	Code	Mnemonic	BA	HL	IX	IY	SP	BR	EP	XP	YP	SC	CC	Memory
8173	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8174	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8175	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8176	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8177	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8178	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8179	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8180	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8181	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8182	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8183	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8184	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8185	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8186	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8187	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8188	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8189	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	
8190	000928	01:0928	DC2404	BIT [BR:24h],#04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	MR:[00FF24]=08
8191	00092B	01:092B	E6FC	JRS Z,FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10-----	Z 0000	

After the trace function is turned on by the md command, the debugger samples trace information while the target program is running. The trace data buffer has a capacity for 8192 instructions (overwritten from the beginning if the capacity is exceeded), and its data can be displayed in the [Trace] window.

The following lists the trace contents:

- Instruction number
- Fetched code and disassembled contents
- Register and condition flag contents
- Memory access status (R/W, address, data)

This window also displays the trace data search results by the ts command.

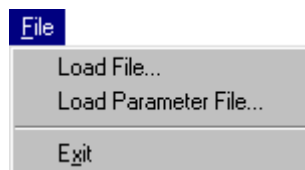
* Updating of display:

The contents of the [Trace] window are cleared when the target program is executed. After the execution has finished, the [Trace] window displays the contents of the trace buffer.

5.5 Menu

This section outlines the menu bar available with the debugger. The menu bar has eight menus, each including frequently-used commands.

[File] Menu



[Load File...]

This button reads an internal ROM HEX file in Motorola S2 format into the debugger. It performs the same function when the lf command is executed.

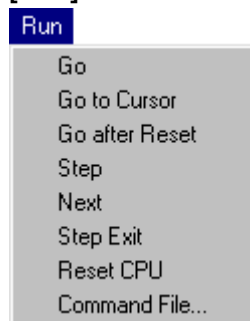
[Load Parameter File...]

This button reads a parameter file into the debugger. It performs the same function when the par command is executed.

[Exit]

This menu item quits the debugger. It performs the same function when the q command is executed.

[Run] Menu



[Go]

This menu item executes the target program from the address indicated by the current PC. It performs the same function when the g command is executed.

[Go to Cursor]

This menu item executes the target program from the address indicated by the current PC to the cursor position in the [Disassemble] window (the address of that line). It performs the same function when the g <address> command is executed. Before this menu item can be selected, the [Disassemble] window must be open and the address line where the program is to break must be clicked.

[Go after Reset]

This button resets the CPU and then executes the target program after fetching the reset vector. It performs the same function when the gr command is executed.

[Step]

This menu item executes one instruction step at the address indicated by the current PC. It performs the same function when the s command is executed.

[Next]

This button executes one instruction step at the address indicated by the current PC. If the instruction to be executed is cars, carl, call, or int, it is assumed that a program section until control returns to the next address constitutes one step and all steps of their subroutines are executed. This button performs the same function when the n command is executed.

[Step Exit]

This button executes the target program from the address indicated by the current PC. If the program starts from inside a subroutine, the program execution will stop when the sequence returns to the parent routine. This button performs the same function when the se command is executed.

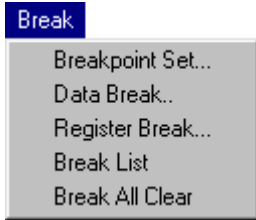
[Reset CPU]

This menu item resets the CPU. It performs the same function when the rst command is executed.

[Command File...]

This menu item reads a command file and executes the debug commands written in that file. It performs the same function when the com or cmw command is executed.

[Break] Menu



[Breakpoint Set...]

This menu item displays, sets or clears PC breakpoints using a dialog box. It performs the same function as executing the bp command.

[Data Break...]

This menu item displays, sets or clears data break conditions using a dialog box. It performs the same function as executing the bd command.

[Register Break...]

This menu item displays, sets or clears register break conditions using a dialog box. It performs the same function as executing the br command.

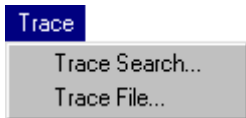
[Break List]

This menu item displays the all break conditions that have been set. It performs the same function as executing the bl command.

[Break All Clear]

This menu item clears all break conditions. It performs the same function as executing the bac command.

[Trace] Menu



[Trace Search...]

This menu item searches trace information from the trace data buffer under the condition specified using a dialog box. It performs the same function as executing the ts command.

[Trace File...]

This menu item saves the specified range of the trace information displayed in the [Trace] window to a file. It performs the same function as executing the tf command.

[View] Menu



[Command]

This menu item activates the [Command] window.

[Disassemble]

This menu item opens or activates the [Disassemble] window and displays the program from the current PC address.

[Dump]

This menu item opens or activates the [Dump] window and displays the memory contents from the memory start address.

[Register]

This menu item opens or activates the [Register] window and displays the current values of the registers.

[Trace]

This menu item opens or activates the [Trace] window and displays the trace data sampled in the trace data buffer.

[Symbol]

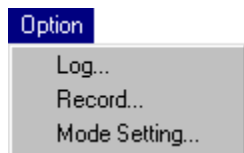
This menu item opens or activates the [Symbol] window and displays the contents of the symbol file if it has been loaded.

[Toolbar]

This menu item shows or hides the toolbar.

[Status Bar]

This menu item shows or hides the status bar.

[Option] Menu**[Log...]**

This menu item starts or stops logging using a dialog box. It performs the same function as executing the log command.

[Record...]

This menu item starts or stops recording of a command execution using a dialog box. It performs the same function as executing the rec command.

[Mode Setting...]

This menu item sets the trace function, coverage function, step-display mode, and execution interval time for the cmw command. It performs the same functions as executing the md command.

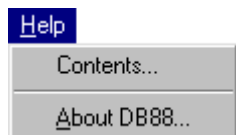
[Windows] Menu**[Cascade]**

This menu item cascades the opened windows.

[Tile]

This menu item tiles the opened windows.

This menu shows the currently opened window names. Selecting one activates the window.

[Help] Menu**[Contents...]**

This menu item displays the contents of help topics.

[About DB88...]

This menu item displays an About dialog box for the debugger.

5.6 Tool Bar

This section outlines the tool bar available with the debugger.

The tool bar has 12 buttons, each one assigned to a frequently used command.



The specified function is executed when you click on the corresponding button.



[Load File] button

This button reads an internal ROM HEX file in Motorola S2 format into the debugger. It performs the same function when the lf command is executed.



[Load Parameter] button

This button reads a parameter file into the debugger. It performs the same function when the par command is executed.



[Key Break] button

This button forcibly breaks execution of the target program. This function can be used to cause the program to break when the program has fallen into an endless loop.



[Break] button

Use this button to set and clear a breakpoint at the address where the cursor is located in the [Disassemble] window. This function is valid only when the [Disassemble] window is open.



[Go] button

This button executes the target program from the address indicated by the current PC. It performs the same function when the g command is executed.



[Go to Cursor] button

This button executes the target program from the address indicated by the current PC to the cursor position in the [Disassemble] window (the address of that line). It performs the same function when the g <address> command is executed.

Before this button can be selected, the [Disassemble] window must be open and the address line where the program is to break must be clicked.



[Go after Reset] button

This button resets the CPU and then executes the target program after fetching the reset vector. It performs the same function when the gr command is executed.



[Step] button

This button executes one instruction step at the address indicated by the current PC. It performs the same function when the s command is executed.



[Next] button

This button executes one instruction step at the address indicated by the current PC. If the instruction to be executed is cars, carl, call, or int, it is assumed that a program section until control returns to the next address constitutes one step and all steps of their subroutines are executed. This button performs the same function when the n command is executed.



[Step Exit] button

This button executes the target program from the address indicated by the current PC. If the program starts from inside a subroutine, the program execution will stop when the sequence returns to the parent routine. This button performs the same function when the se command is executed.



[Reset CPU] button

This button resets the CPU. It performs the same function when the rst command is executed.



[Help Topics] button

By clicking on this button, a help window appears on the screen, displaying the contents of help topics.

5.7 Method for Executing Commands

All debug functions can be performed by executing debug commands. This section describes how to execute these commands.

5.7.1 Entering Commands from Keyboard

Select the [Command] window (by clicking somewhere on the [Command] window). When the prompt ">" appears on the last line in this window and a cursor is blinking behind it, the system is ready to accept a command from the keyboard.

Input a debug command at the prompt position. The commands are not case-sensitive; they can be input in either uppercase or lowercase.

General command input format

```
>command [ parameter [ parameter ... parameter ] ] ↵
```

- A space is required between a command and parameter.
- Space is required between parameters.

Use the arrow keys, [Back Space] key, or [Delete] key to correct erroneous input.

When you press the [Enter] key after entering a command, the system executes that command. (If the command entered is accompanied by guidance, the command is executed when the necessary data is input according to the displayed guidance.)

Input example:

```
>g↵ (Only a command is input.)
>com test.cmd↵ (A command and parameter are input.)
```

Command input accompanied by guidance

For commands that cannot be executed unless a parameter or the commands that modify the existing data are specified, a guidance mode is entered when only a command is input. In this mode, the system brings up a guidance field, so input a parameter there.

Input example:

```
>lf↵
File name ? :test.abs↵ ← Input data according to the guidance (underlined part).
>
```

• Commands requiring parameter input as a precondition

The lf command shown in the above example reads a program file into the debugger. Commands like this that require an entered parameter as a precondition are not executed until the parameter is input and the [Enter] key pressed. If a command has multiple parameters to be input, the system brings up the next guidance, so be sure to input all necessary parameters sequentially. If the [Enter] key is pressed without entering a parameter in some guidance session of a command, the system assumes the command is canceled and does not execute it.

• Commands that replace existing data after confirmation

The commands that rewrite memory or register contents one by one provide the option of skipping guidance (do not modify the contents), returning to the immediately preceding guidance, or terminating during the input session.

[Enter] key	Skips input.
[^] key	Returns to the immediately preceding guidance.
[q] key	Terminates the input session.

5 DEBUGGER

Input example:

```
>de↵ ← Command to modify data memory.
Data enter address ? :001000↵ ← Inputs the start address.
001000 A:1↵ ← Modifies address 001000H to 1.
001001 A:^↵ ← Returns to the immediately preceding address.
001000 1:0↵ ← Inputs address 001000H back again.
001001 A:↵
001002 A:↵
001001 A:q↵ ← Terminates the input session.
>
```

Numeric data format of parameter

For numeric values to be accepted as a parameter, they must be input in hexadecimal numbers for almost all commands. However, some parameters accept decimal or binary numbers.

The following characters are valid for specifying numeric data:

Hexadecimal: 0-9, a-f, A-F, *

Decimal: 0-9

Binary: 0, 1, *

("*" is used to mask bits when specifying a data pattern.)

Specification with a symbol

For address specifications, the symbols registered in the symbol file (.sy) can also be used.

Input example:

```
>u Main↵ ← Displays the program from the label Main
```

- * The symbol file is automatically loaded simultaneously with the target program. However, it must be the same name (extension is .sy) and be located in the same directory as the target program file.

Successive execution using the [Enter] key

The commands listed below can be executed successively by using only the [Enter] key after executing once. Successive execution here means repeating the previous operation or continuous display of the previous contents.

Execution commands: g, s, n, se, com











Display commands: u, dd, td

The successive execution function is terminated when some other command is executed.

5.7.2 Executing from Menu or Tool Bar

The menu and tool bar are assigned frequently-used commands as described in Sections 5.5 and 5.6. A command can be executed simply by selecting desired menu command or clicking on the tool bar button. Table 5.7.2.1 lists the commands assigned to the menu and tool bar.

Table 5.7.2.1 Commands that can be specified from menu or tool bar

Command	Function	Menu	Button
lf	Load program file	[File Load File...]	
par	Load parameter file	[File Load Parameter File...]	
g	Execute program successively	[Run Go]	
g <address>	Execute program to <address> successively	[Run Go to Cursor]	
gr	Reset CPU and execute program successively	[Run Go after Reset]	
s	Single step execution	[Run Step]	
n	Step execution with skip subroutine	[Run Next]	
se	Exit from subroutine	[Run Step Exit]	
com	Load and execute command file	[Run Command File...]	-
cmw	Load and execute command file with wait	[Run Command File...]	-
rst	Reset CPU	[Run Reset CPU]	
bp, bc (bpc)	Set/clear PC breakpoint	[Break Breakpoint Set...]	
bd, bdc	Set/clear data break	[Break Data Break...]	-
br, brc	Set/clear register break	[Break Register Break...]	-
bl	Break list	[Break Break List]	-
bac	Clear all break conditions	[Break Break All Clear]	-
ts	Search trace information	[Trace Trace Search...]	-
tf	Save trace information to file	[Trace Trace File...]	-
u	Disassemble display	[View Disassemble]	-
dd	Dump memory	[View Dump]	-
rd	Display register values	[View Register]	-
td	Display trace information	[View Trace]	-
sy	Display symbols	[View Symbol]	-
log	Turn log output on or off	[Option Log...]	-
rec	Record commands to a command file	[Option Record...]	-
md	Set modes	[Option Mode Setting...]	-

5.7.3 Executing from a Command File

Another method for executing commands is to use a command file that contains descriptions of a series of debug commands. By reading a command file into the debugger the commands written in it can be executed.

Creating a command file

Create a command file as a text file using an editor.

Although there are no specific restrictions on the extension of a file name, Seiko Epson recommends using ".cmd".

Command files can also be created using the rec command. The rec command creates a command file and saves the executed commands to the file.

Example of a command file

The example below shows a command group that loads a program file, sets a breakpoint and then executes the program.

Example: File name = start.cmd

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

A command file to write the commands that come with a guidance mode can be executed. In this case, be sure to break the line for each guidance input item as a command is written.

Reading in and executing a command file

The debugger has the com and cmw commands available that can be used to execute a command file. The com command reads in a specified file and executes the commands in that file sequentially in the order they are written.

The cmw command performs the same function as the com command except that each command is executed at intervals specified by the md command (1 to 256 seconds).

Example: com start.cmd

```
cmw test.cmd
```

The commands written in the command file are displayed in the [Command] window.

Restrictions

Another command file can be read from within a command file. However, nesting of these command files is limited to a maximum of five levels. An error is assumed and the subsequent execution is halted when the com or cmw command at the sixth level is encountered.

5.7.4 Log File

The executed commands and the execution results can be saved to a file in text format that is called a "log file". This file allows verification of the debug procedures and contents.

The contents displayed in the [Command] window are saved to this file.

Command example

```
>log tst.log
```

After the debugger is set to the log mode by the log command (after it starts outputting to a log file), the log command toggles (output turned on in log mode ↔ output turned off in normal mode).

Therefore, you can output only the portions needed can be output to the log file.

Display of [Command] window in log mode

The contents displayed in the [Command] window during log mode differ from those appearing in normal mode.

- (1) When executing a command when each window is open
(When the window that displays the command execution result is opened)
Normal mode: The contents of the relevant display window are updated. The execution results are not displayed in the [Command] window.
Log mode: The same contents as those displayed in the relevant window are also displayed in the [Command] window. However, changes made to the relevant window by scrolling or opening it are not reflected in the [Command] window.
- (2) When executing a command while each window is closed
When the relevant display window is closed, the execution results are always displayed in the [Command] window regardless of whether operation is in log mode or normal mode.

5.8 Debug Functions

This section outlines the debug features of the debugger, classified by function.

5.8.1 Loading Files

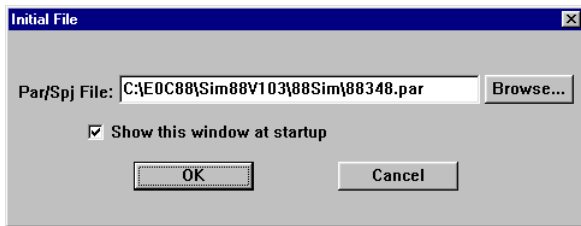
Table 5.8.1.1 lists the files read by the debugger and the load commands.

Table 5.8.1.1 Files and load commands

File	Type	Generation tool	Command	Menu	Button
1. Parameter file	.par	–	par	[File Load Parameter File...]	
2. LCD panel definition file	.lcd	LcdUtil	–	–	–
3. Component mapping file	.cmp	–	–	–	–
4. Port setting file	.prt	–	–	–	–
5. Simulator project file	.spj	–	par	[File Load Parameter File...]	
6. Program file	.psa	fil88xxx	lf	[File Load File...]	
7. Function option file	.fsa	fog88xxx or winfog	lf	[File Load File...]	
8. Symbol file	.sy	sym88	–	–	–
9. Command file	.cmd	–	com/cmw	[Run Command File...]	–

Files 1 to 4 are required for starting the debugger.

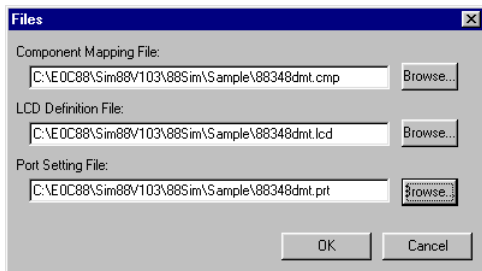
Either a parameter file (1) or a simulator project file (5) must be selected in the dialog box shown below at the first time the debugger starts up.



Enter the parameter file name or simulator project file name to the text box, or choose it using the [Browse...] button. If the [Show this window at startup] check box is deselected, this dialog box will not appear from the next startup of the debugger and the same file will be selected.

Loading a parameter file resets the debugger. The memory mapping information set by the parameter file can be displayed using the ma command. Refer to the Development Tool Manual for more information on the parameter file.

When a parameter file is selected, the dialog box shown below appears to select files from 2 to 4. When a simulator project file is selected, this dialog box does not appear because the file names (2 to 4) are specified in the file.



Enter a component mapping file name, LCD panel definition file name and a port setting file name to the respective text boxes, or chose using the [Browse...] button. These files are selected once, the file names will appear in the text boxes at the next startup of the debugger allowing choice of the files by clicking the [OK] button only.

The lf command loads a program file (.psa) or a function option HEX file (.fsa). The debugger distinguishes these two files with the specified extension.

The symbol file is required to specify addresses using the symbols defined in the source. Debugging can be done even if this file is not loaded. The symbol file is loaded simultaneously with the program file by the lf command. However, it must be the same name (extension is .sy) and be located in the same directory as the program file.

Refer to Section 5.7.3, for the command file.

5.8.2 Displaying and Modifying Program, Data, and Register

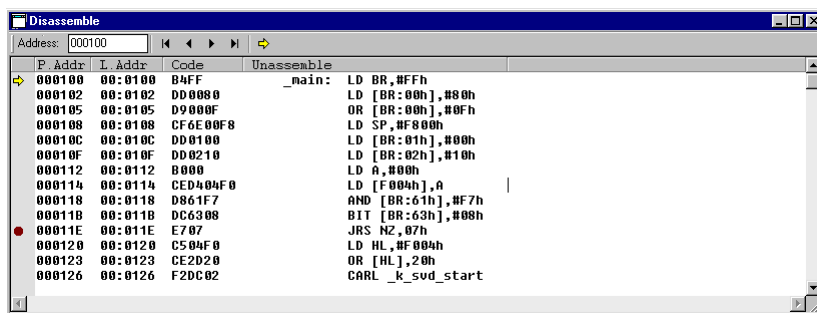
The debugger has functions to operate on the memory and registers. Available memory area is set to the debugger according to the map information that is given in a parameter file.

Program code display

Table 5.8.2.1 Program display command/menu item

Function	Command	Menu	Button
Disassemble display	u	[View Disassemble]	–

The program codes are displayed in the [Disassemble] window or [Command] window with the addresses and disassembled contents. Normally they are displayed from the current PC address. The display start position can be specified using the control on the [Disassemble] window or the u command. See Section 5.4.4, "[Disassemble] Window", for the display contents and operation on the [Disassemble] window.



Memory operation

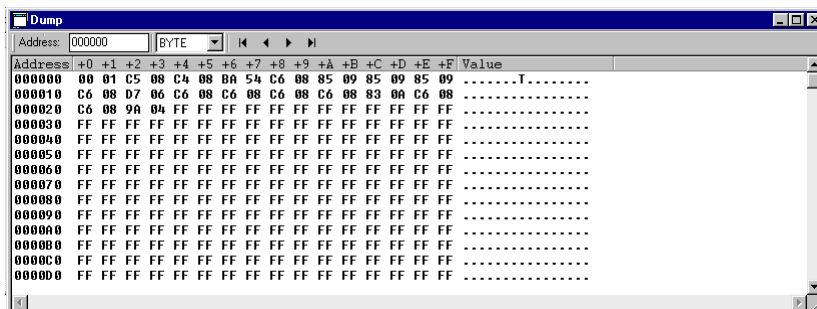
The following operations can be performed on the memory areas (ROM, RAM, display memory, I/O memory):

Table 5.8.2.2 Memory operation commands/menu item

Function	Command	Menu	Button
Dumping memory data	dd	[View Dump]	–
Entering/modifying memory data	de	–	–
Rewriting specified area	df	–	–
Coping specified area	dm	–	–

(1) Dumping memory

The memory contents are displayed in hexadecimal dump format. If the [Dump] window is opened, the contents of the [Dump] window are updated; if not, the contents of the data memory are displayed in the [Command] window.



(2) Entering/modifying data

Data at a specified address is rewritten by entering hexadecimal data. Data can be directly modified on the [Dump] window.

(3) Rewriting specified area

An entire specified area is rewritten with specified data.

(4) Copying specified area

The content of a specified area is copied to another area.

See Section 5.4.5, "[Dump] Window", for the display contents and operation on the [Dump] window.

Operating registers

The following operations can be performed on registers:

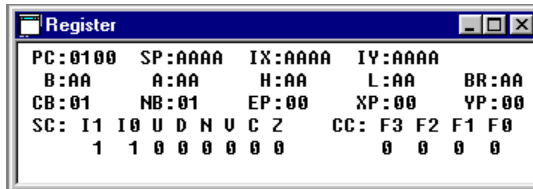
Table 5.8.2.3 Register operation commands/menu item

Function	Command	Menu	Button
Displaying register values	rd	[View Register]	-
Modifying register value	rs	-	-

(1) Displaying registers

Register contents can be displayed in the [Register] or [Command] window.

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



(2) Modifying register values

The contents of the above registers can be set to any desired value.

The register values can be directly modified on the [Register] window.

5.8.3 Executing Program

The debugger can execute the target program successively or execute instructions one step at a time (single-stepping).



Successive execution

(1) Types of successive execution

There are two types of successive execution available:

- Successive execution from the current PC
- Successive execution after resetting the CPU

Table 5.8.3.1 Commands/menu items/tool bar buttons for successive execution

Function	Command	Menu	Button
Successive execution from current PC	g	[Run Go]	
Successive execution after resetting CPU	gr	[Run Go after Reset]	

(2) Stopping successive execution

Using the successive execution command (g <address1> <address2>), can specify up to two temporary break addresses that are only effective during program execution.

The temporary break address can also be specified from the [Disassemble] window.

If the cursor is placed on an address line in the [Disassemble] window and the [Go to Cursor] button clicked, the program starts executing from the current PC address and breaks after executing the instruction at the address the cursor is placed.

Except being stopped by this temporary break, the program continues execution until it is stopped by one of the following causes:

- Break conditions set by a break set up command are met.
- The [Key Break] button is clicked.
- An undefined area is accessed.



[Key Break] button * When the program does not stop, use this button to forcibly stop it.

(3) Simulation for LCD panel display and key inputs

The [LCD] window shows the LCD panel images according to the program sequence during the program execution. It also allows simulation of key inputs using the keyboard of the computer. These functions are configured with the component mapping file, LCD panel definition file and port setting file loaded at the startup of the debugger. See Section 5.4.3, "[LCD] Window", for more information.

Single-stepping




(1) Types of single-stepping

There are three types of single-stepping available:

- Stepping through all instructions (STEP)
All instructions are executed one step at a time according to the PC, regardless of the type of instruction.
- Stepping through instructions except subroutines (NEXT)
The cars, carl, call, and int instructions are executed under the assumption that one step constitutes the range of statements until control is returned to the next step by a return instruction. Other instructions are executed in the same way as in ordinary single-stepping.
- Exit from a subroutine (STEP EXIT)
The current subroutine is executed from the current PC until the return instruction is executed and then the execution is terminated after returning to the parent routine. This function will be the same operation as the g command if it is executed at the top-level routine. The program will not be terminated when a lower level subroutine is called and the return instruction in the routine is executed.

In either case, the program starts executing from the current PC.

Table 5.8.3.2 Commands/menu items/tool bar buttons for single-stepping

Function	Command	Menu	Button
Stepping through all instructions	s	[Run Step]	
Stepping through all instructions except subroutines	n	[Run Next]	
Exit from subroutine	se	[Run Step Exit]	

When executing s or n by command input, the number of steps to be executed can be specified, up to 65,535 steps. When using menu commands or tool bar buttons, the program is executed one step at a time.

In the following cases, single-stepping is terminated before a specified number of steps is executed:

- The [Key Break] button is clicked.
- An undefined area is accessed.

Single-stepping is not suspended by breaks set by the user such as a PC break or data break.



[Key Break] button * When the program does not stop, use this button to forcibly stop it.

(2) Display during single-stepping

In the initial debugger settings, the display is updated as follows:

When the [Disassemble], [Register], or [Dump] window is open, the display contents are also updated after the last step has been executed. If the [Register] window is closed, its contents are displayed in the [Command] window.



The display mode can be changed so that display contents will be updated every step using the md command.

(3) Key entry simulation during single-stepping

Key entry status can be set on the [Key List] window displayed by clicking on the [Key List] button on the [LCD] window and is maintained during single-stepping.

Resetting the CPU

Table 5.8.3.3 Commands/menu items/tool bar buttons for resetting CPU

Function	Command	Menu	Button
Reset CPU	rst	[Run Reset CPU]	
Successive execution after resetting CPU	gr	[Run Go from Reset]	

The CPU is reset when the gr command is executed, or by executing the rst command.

The following shows the initial settings when the CPU is reset.

(1) Internal registers of the CPU and memory

The CPU internal registers are initialized as follows during initial reset.

Table 5.8.3.4 Initial settings

Register name	Code	Bit length	Initial value
Data register A	A	8	Undefined
Data register B	B	8	Undefined
Index (data) register L	L	8	Undefined
Index (data) register H	H	8	Undefined
Index register IX	IX	16	Undefined
Index register IY	IY	16	Undefined
Program counter	PC	16	Undefined*
Stack pointer	SP	16	Undefined
Base register	BR	8	Undefined
Zero flag	Z	1	0
Carry flag	C	1	0
Overflow flag	V	1	0
Negative flag	N	1	0
Decimal flag	D	1	0
Unpack flag	U	1	0
Interrupt flag 0	I0	1	1
Interrupt flag 1	I1	1	1
New code bank register	NB	8	01H
Code bank register	CB	8	Undefined*
Expand page register	EP	8	00H
Expand page register for IX	XP	8	00H
Expand page register for IY	YP	8	00H

* Reset exception processing loads the reset vector stored in bank 0, 000000H–000001H into the PC. At the same time, 01H of the NB initial value is loaded into CB.

The internal RAM and external RAM are not initialized at initial reset.

The respectively stipulated initializations are done for internal peripheral circuits.

(2) Redisplaying the [Disassemble] and [Register] windows

Because the PC is reset, the [Disassemble] window is redisplayed beginning with that address.

The [Register] window is redisplayed with the settings above.

5.8.4 Break Functions

The target program is made to stop executing by one of the following causes:

- Break command conditions are satisfied.
- The [Key Break] button is clicked.
- An undefined area is accessed.

Break by command

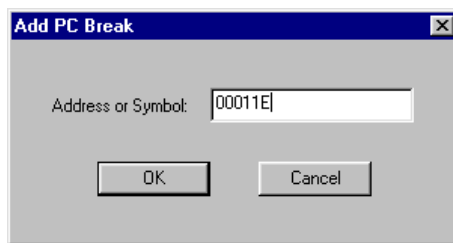
The debugger has three types of break functions that allow the break conditions to be set by a command. When the set conditions in one of these break functions are met, the program under execution is made to break.

(1) Break by PC

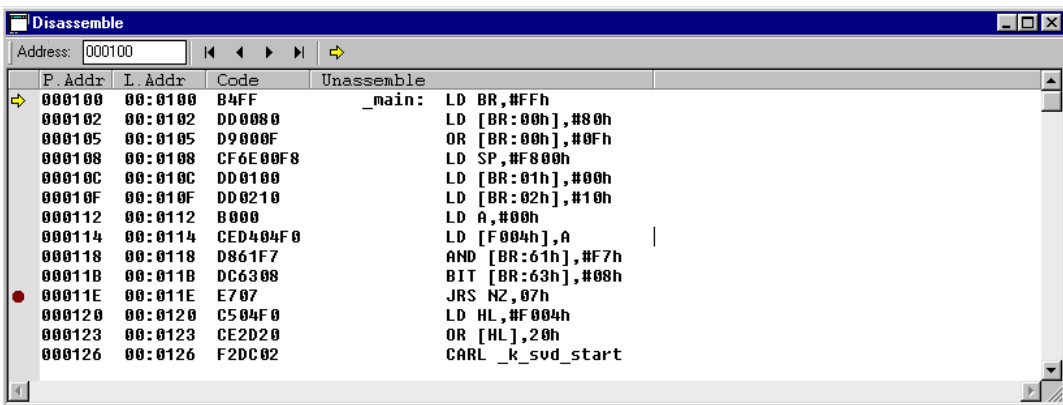
This function causes the program to break when the PC matches the set address. The program is made to break after executing the instruction at that address. The PC breakpoints can be set up to 64 addresses.

Table 5.8.4.1 Commands/menu items/tool bar button to set breakpoints

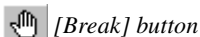
Function	Command	Menu	Button
Set breakpoints	bp	[Break Breakpoint Set...]	
Clear break points	bc (bpc)	[Break Breakpoint Set...]	



The addresses that are set as PC breakpoints are marked with a ● as they are displayed in the [Disassemble] window.



Using the [Break] button easily allows the setting and canceling of breakpoints.



[Break] button

Click on the address line in the [Disassemble] window at where the program break is desired (after moving the cursor to that position) and then click on the [Break] button. A ● mark will be placed at the beginning of the line indicating that a breakpoint has been set there, and the address is registered in the breakpoint list. Clicking on the line that begins with a ● and then the [Break] button cancels the breakpoint you have set, in which case the address is deleted from the breakpoint list.

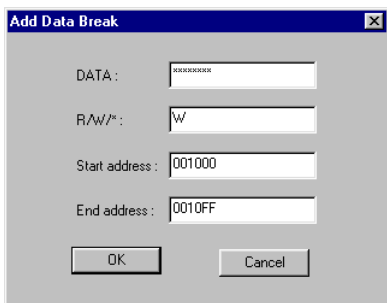
* The temporary break addresses that can be specified by the successive execution commands (g) do not affect the set addresses in the breakpoint list.

(2) Data break

This break function allows a break to be executed when a location in the specified memory area is accessed. In addition to specifying a memory area in which to watch accesses, specification as to whether the break is to be caused by a read or write, as well as specification of the content of the data read or written. The read/write condition can be masked, so that a break will be generated for whichever operation, read or write, is attempted. Similarly, the data condition can also be masked in bit units. A break occurs after completing the cycle in which an operation to satisfy the above specified condition is performed.

Table 5.8.4.2 Commands/menu item to set data break

Function	Command	Menu	Button
Set data break condition	bd	[Break Data Break...]	–
Clear data break condition	bdc	[Break Data Break...]	–



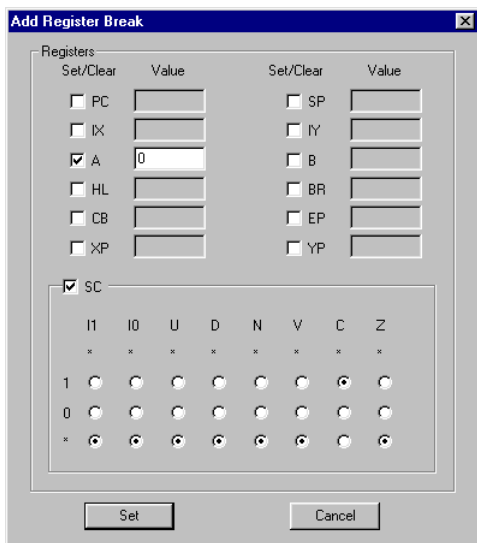
For example, if the program is executed after setting the data break condition as Address = 001000H–0010FFH, Data pattern = * (mask) and R/W = W, the program breaks after writing any data to the area from address 001000H to address 0010FFH.

(3) Register break

This break function causes a break when the PC, SP, IX, IY, A, B, HL, BR, CB, NB, EP, XP and YP registers and flags (I1, I0, U, D, N, V, C, Z) reach a specified value. Each register can be masked (so they are not included in break conditions). The flag register can be masked in bit units. A break occurs when the above registers are modified to satisfy all set conditions.

Table 5.8.4.3 Commands/menu item to set register break

Function	Command	Menu	Button
Set register break condition	br	[Break Register Break...]	–
Clear register break condition	brc	[Break Register Break...]	–



For example, if the program is executed after setting 0 for the data of register A and 1 for the data of flag C and masking all others, the program execution breaks when the A register is cleared to 0 and the C flag is set to 1.

Forced break by the [Key Break] button

The [Key Break] button can be used to forcibly terminate the program under execution when the program has fallen into an endless loop or cannot exit a standby (HALT or SLEEP) state.



[Key Break] button

Break due to an illegal access

The program execution will be terminated when an error listed below occurs.

Accessing to an undefined program area

A break occurs when the target program jumps to a program area that has not been defined in the parameter file.

In this case the following message is displayed in the [Command] window.

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

```
>
```

Accessing to outside the stack area

A break occurs when a stack operation is executed to an area outside the stack defined in the parameter file.

In this case the following message is displayed in the [Command] window.

```
Out of stack area
```

```
>
```

5.8.5 Trace Functions

The debugger has a function to trace program execution.

The trace function is initially disabled and can be enabled using the md command ([Option | Mode setting...]).

Trace data buffer and trace information

The debugger has a trace data buffer. When the debugger executes the program, the trace information on each executed instruction is taken into this buffer. The trace data buffer has the capacity to store information for 8,192 instructions. When the trace information exceeds this capacity, the data is overwritten, the oldest data first. Consequently, the trace information stored in the trace data buffer is always within 8,192 instructions. The trace data buffer is cleared when a program is executed, starting to trace the new execution data.

INS	P Addr	L Addr	Code	Mnemonic	BA	HL	IX	IY	SP	BR	BP	XP	YP	SC	CC	Memory
8173	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8174	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8175	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8176	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8177	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8178	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8179	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8180	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8181	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8182	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8183	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8184	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8185	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8186	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8187	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8188	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8189	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	
8190	000928	01:0928	DC2404	BIT [BR:24h], #04h	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	MR: [00FF24]=00
8191	00092B	01:092B	E6FC	JRS Z, FCh	AA00	F004	AAAA	AAAA	F7FD	FF	00	00	00	10----Z	0000	

The following lists the trace information that is taken into the trace data buffer in every instruction execution cycle. This list is corresponded to display in the [Trace] window.

- INS: Executed instruction number (0 to 8191, decimal)
0000 means oldest trace data.
- P Addr: PC address (hexadecimal physical address)
- L Addr: PC address (hexadecimal logical address)
- Code: Instruction code (hexadecimal)
- Mnemonic: Disassembled instruction code
- BA to CC: Values of the CPU registers (hexadecimal)
- IDZC: Values of I, D, Z and C flags (binary) after cycle execution
- Memory: Memory read/write operation (MR: or MW:), accessed memory address (hexadecimal), and data (hexadecimal)

Displaying and searching trace information

The sampled trace information is displayed in the [Trace] window after a program execution has finished. In the [Trace] window, the entire trace data buffer can be seen by scrolling the window. The trace information can be displayed beginning from a specified cycle using a command. The display contents are as described above.

If the [Trace] window is closed, the information can be displayed in the [Command] window using a command.

Table 5.8.5.1 Command/menu item to display trace information

Function	Command	Menu	Button
Display trace information	td	[View Trace]	-

It is possible to specify a search condition and display the trace information that matches a specified condition.

The search condition can be selected from the following three:

1. Program's execution address
2. Address from which data is read
3. Address to which data is written

When the above condition and one address are specified, the system starts searching. When the trace information that matches the specified condition is found, the system displays the found data in the [Trace] window (or in the [Command] window if the [Trace] window is closed).

Table 5.8.5.2 Command/menu item to search trace information

Function	Command	Menu	Button
Search trace information	ts	[Trace Trace Search...]	–

Saving trace information

The trace information within the specified range can be saved to a file.

Table 5.8.5.3 Command/menu item to save trace information

Function	Command	Menu	Button
Save trace information	tf	[Trace Trace File...]	–

5.8.6 Coverage

The debugger can sample coverage information (i.e., information on addresses at which a program is executed) and the information can be displayed in the [Command] window.

This function is initially disabled and can be enabled using the md command ([Option | Mode setting...]). Because the executed address range is displayed as shown below, it is possible to know which areas have not been executed.

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

Table 5.8.6.1 Coverage commands

Function	Command	Menu	Button
Display coverage information	cv	–	–
Clear coverage information	cvc	–	–

When the coverage function is enabled, the [Disassemble] window shows the executed address with an * placed at the beginning of the line.

5.9 Command List

Table 5.9.1 Command list

Classification	Command	Function
Memory operation	dd [<code><addr1></code> [<code><addr2></code>] [{-B -W -L -F -D}]]	Dump memory data
	de [<code><addr. <data1></code> [<code>..<code><data16></code>]]</code>	Enter memory data
	df [<code><addr1></code> <code><addr2></code> <code><data></code>]	Fill memory area
	dm [<code><addr1></code> <code><addr2></code> <code><addr3></code>]	Copy memory area
Register operation	rd	Display register values
	rs [<code><reg></code> <code><value></code> [<code>..<code><reg></code> <code><value></code>]]</code>	Modify register value reg={PC SP IX IY A B HL BR CB EP XP YP SC I1 I0 U D N V Z C}
Program execution	g [<code><addr1></code> [<code><addr2></code>]]	Execute program successively from current PC
	gr [<code><addr1></code> [<code><addr2></code>]]	Execute program successively after resetting CPU
	s [<code><step></code>]	Single stepping from current PC
	n [<code><step></code>]	Single stepping with skip subroutines
	se	Exit from subroutine
CPU reset	rst	Reset CPU
Break	bp [<code><addr1></code> [<code>..<code><addr16></code>]]</code>	Set breakpoints
	bc [<code><addr1></code> [<code>..<code><addr16></code>]]</code>	Clear breakpoints
	bpc [<code><addr1></code> [<code>..<code><addr16></code>]]</code>	
	bd [<code><data></code> {r w *} <code><addr1></code> <code><addr2></code>]	Set data break condition
	bdc	Clear data break condition
	br [<code><reg></code> <code><value></code> [<code>..<code><reg></code> <code><value></code>]]</code>	Set register break condition reg={PC SP IX IY A B HL BR CB EP XP YP SC}
	brc	Clear register break condition
	bl	Display all break conditions
	bac	Clear all break conditions
	Program display	u [<code><addr></code>]
Symbol display	sy [/a]	Display symbol list
Load file	lf [<code><file name></code>]	Load program/option HEX file
	par [<code><file name></code>]	Load parameter file
Trace	td [<code><index></code>]	Display trace information
	ts [{pc dr dw} <code><addr></code>]	Search trace information
	tf [[<code><index1></code> [<code><index2></code>]] <code><file name></code>]	Save trace information
Others	cv [<code><addr1></code> [<code><addr2></code>]]	Display coverage information
	cvc	Clear coverage information
	com [<code><file name></code> [<code><interval></code>]]	Load and execute command file
	cmw [<code><file name></code>]	Load and execute command file with execution interval
	rec [<code><file name></code>]	Record executed commands to file
	log [<code><file name></code>]	Logging
	ma	Display map information
	md [<code><option></code> <code><num></code> [<code>..<code><option></code> <code><num></code>]]</code>	Set modes option={-tm -cv -s -cm}
	q	Quit debugger
	?	Display command usage

5.10 Component Mapping File (.cmp)

The component mapping file (peripheral setting file) is the text file to record the information required to simulate the CPU, key entry and LCD display by the debugger. The files that contain the internal peripheral circuit data for available models are provided. Use these files after adding the necessary information using a text editor. Since the debugger (simulator) saves the LCD panel information set in the debugger, such as LCD panel color, to this file, do not set the file attribute to Read Only.

The example below is 88348dmt.cmp in the sample directory.

Example: 88348dmt.cmp

```
[Internal]
CPU=CPU.bmc

LCD=LcdDrv88.bmc
K/P/R port=KPRport.bmc
SVD=SVD88.bmc

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

[External]
Ext0=SED152A.bmc,080000,080002
```

[Internal] Internal peripheral circuit parameters

The file does not allow the user to modify these parameters.
Edit the parameter file (.par) to set the ROM/RAM size.

[Settings] CPU parameters

All of these parameters must be specified as follows:

```
[Settings]           ← Start of Settings section
CpuType=88348       ← Family name
ChipMode=MCU       ← Chip type (MCU or MPU)
CpuModel=3         ← CPU model (1 or 3)
OSC1=32.768KHz    ← OSC1 clock frequency
OSC3=4.9152MHz    ← OSC3 clock frequency
```

[External] External device parameters

Use the parameter file (.par) to set the ROM/RAM size.

When using external devices other than ROM and RAM, describe the information of the devices according to the address decoder settings. The currently supported devices are LCD driver (SED152A) and backlights (BkLight).

```
[External]           ← Start of External section
Ext0=SED152A.bmc,080000,080001
Ext1=BkLight.bmc,180004,180004,4L
```

Definition: Ext<N>=<device>.bmc, <Start_addr>, <End_addr>, <Option>

Ext<N>: N is a sequential device number beginning from 0.

<device>.bmc: Peripheral device definition file name

<Start_addr>: I/O map start address

<End_addr>: I/O map end address

<Option>: Peripheral specific option

Setting the SED152A

The SED152A is mapped to a 2-byte space by connecting it to A0 the most significant bit of the address bus. The example shown above sets addresses 080000H (A0=0) and 080001H (A0=1) assuming that the connected CE signal becomes active from address 080000H. When data is written to this address, it is output to D0–D7 of the SED152A. On the other hand, the D0–D7 data output from the SED152A can be read from this address.

Setting the BkLight

Specify the I/O address connected to the backlight. The start and end addresses have the same value. The option allows definition of the bit number and active level (H or L) to turn the backlight on. The example shown above defines so that the backlight goes on when bit 4 is set to low.

5.11 Port Setting File (.prt)

The port setting file is the text file that contains the definitions required to simulate key entry operation. Use a text editor to create it.

Definition of push-keys

Declare the push-key section using [Push Key], and then define the correspondence between ports and key codes (described later).

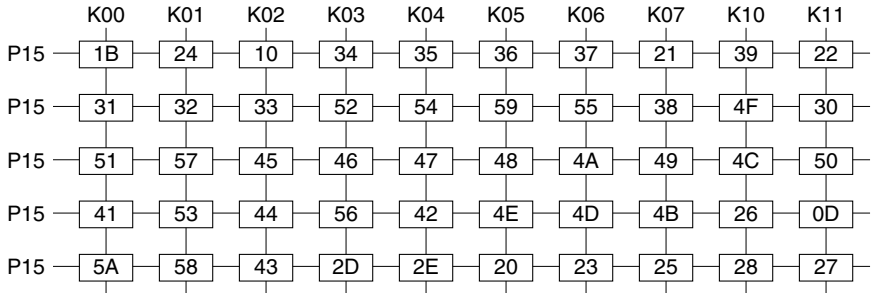
Example:

```
[Push Key]
K00=30
K01=31
K02=32
K03=33
K04=34
K05=35
K06=36
K07=37
K10=38
K11=39
```

Definition of the key matrix

Declare the key-matrix section using [Key Matrix], and define the input and output ports that configure the key matrix. Then define the key code corresponding to the input port that will be scanned when the output port goes low. Leave blank if no key is connected to the input port.

Example:



[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
```


Key names

The key name of the target system can be assigned to a PC key (code) used in the push-key or key-matrix definition. Declare the key name section using [Key Name] and then define the correspondence between PC key codes and target key name.

Example:

```
[Key Name]
24=Caps      ← Home key (key code 24) = Caps
22=On/Off    ← PageDown key (key code 22) = On/Off
21=Mode      ← PageUp key (key code 21) = Mode
23=Char      ← End key (key code 23) = Char
2D=Data      ← Insert key (key code 2D) = Data
```

Key code list (hexadecimal)

The following lists the key codes that are sent to the simulator when PC keys are entered. Note that the numeric keypad (NumPad) has an independent key code.

Key	Code
0 to 9	30 to 39
A to Z	41 to 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
(Numeric keypad)	
0 to 9	60 to 69
*	6A
+	6B
-	6D
.	6E
/	6F
(Function keys)	
F1 to F24	70 to 87

5.12 Simulator Project File (.spj)

The simulator project file is the text file that contains the names of the parameter file, LCD panel definition file, component mapping file and port setting file to be used. Use a text editor to create it. This file allows simultaneous selection of the files described above.

When the debugger starts up or the par command ([File | Load Parameter File]) is executed, a dialog box appears to select a parameter file or a simulator project file. When selecting a parameter file, a simulator project file is not necessary.

The following is an example of the file contents.

Example: 88348dmt.spj

```
[Setting]
PAR=88348.par      ← Parameter file
LCD=88348dmt.lcd  ← LCD panel definition file
CMP=88348dmt.cmp  ← Component mapping file
PRT=88348dmt.prt  ← Port setting file
```

5.13 Restrictions

- This simulator supports the following models:
E0C88317, E0C88348, E0C88832, E0C88816, E0C88862, E0C88F360
- The supported external devices are ROMs, RAMs, LCD driver (SED152A only), monochrome LCD panels, backlight, keys and key matrix.
- The simulator supports external ROM, RAM and LCD drivers that are interfaced through the external bus. Serial connection of the SED152A and accessing to external ROM/RAM via general-purpose ports are not supported.
- This simulator performs instruction-level simulation. Therefore, execution cycles shorter than the instruction cycle cannot be simulated.
- Since timers are simulated based on the instruction cycle, the timings are different from those of the actual hardware.
- Wait-cycles for accessing the external peripheral devices cannot be simulated properly.
- The following functions are not supported:
 1. Serial I/O
 2. Buzzer output
 3. TOUT/FOUT output
 4. A/D converter, D/A converter and analog comparator
 5. Event counter mode of programmable timer
- Program files not in Motorola S2 format cannot be debugged (simulated). C and assembler source level debugging are not supported.
- To simulate the sound generator, a sound card that supports playback of PCM sound source must be installed in the PC.
- Two or more simulators cannot be executed simultaneously for multiple simulations.
- When using the multiple-key entry reset function of the K ports, the simulator cannot reset the CPU even if the four assigned keys are pressed simultaneously. To reset using four keys, press three keys immediately after pressing one other key or press two keys immediately after pressing two other keys.

Some restrictions described above may be lifted or modified at product release. Refer to readme.txt for the latest information such as restrictions, supported models/functions and known bugs.

5.14 Debugger Messages

Status message list

Message	Description	Commands involved
Break by accessing no map program area	Break caused by accessing undefined program-memory area.	
Break by data break	Break caused by data break condition.	
Break by PC break	Break caused by PC breakpoint.	
Break by register break	Break caused by register break condition.	
Key Break	Break caused by [Key break] button.	
No coverage data	No coverage information.	cv
No matched trace data	No trace information matches the specified condition.	ts
No trace data	No trace information.	td, tf, ts
Out of stack area	Break caused by accessing outside stack area.	
Set to log off mode	Set log function off.	log
Set to log on mode	Set log function on.	log
Set to record off mode	Set record function off.	rec
Set to record on mode	Set record function on.	rec
Simulator initialization error!	Failed to initialize the simulator.	
Symbol file does not exist	The corresponding symbol file does not exist in the same directory as the program file.	lf
Symbol file is loaded	The corresponding symbol file is loaded with the program file.	lf

Error message list

Message	Description	Commands involved
Address out of range, use 0 - 0x7FFFFFFF	The address is outside program memory area.	g, bp, bc(bpc), u, ts, cv
Address out of range, use 0 - 0xFFFFFFFF	The address is outside data memory area.	dd, de, df, dm, bd, ts
Cannot open file	The file cannot be opened.	lf, par, com, cmw
Data out of range, use 0 - 0xFF	Data value is invalid.	de, df
End address < start address	The end address is less than the start address.	dd, df, dm, cv
End index < start index	The end index is less than the start index.	tf
Error file type (extension should be CMD)	The file extension is invalid as command file.	com, cmw
Error file type (extension should be PAR)	The file extension is invalid as parameter file.	par
Incorrect number of parameters	Number of parameters is invalid.	
Incorrect r/w option, use r/w*	The read/write option is invalid.	bd
Incorrect register name, use PC/SP/SP/IX/IY/A/B/HL/BR/CB/EP/XP/YP/SC	The register name is invalid for setting register break condition.	br
Incorrect register name, use PC/SP/SP/IX/IY/A/B/HL/BR/CB/EP/XP/YP/SC/11/10/U/D/N/V/Z/C	The register name is invalid.	rs
Index out of range, use 0 - 8191	The index is over the valid range.	td
Input address does not exist	Attempt is made to clear a breakpoint that has not been set.	bp, bc (bpc)
Invalid command	The command is invalid.	
Invalid data pattern	The data pattern is invalid.	bd, br
Invalid display unit, use -B/-W/-L/-F/-D	The option is invalid for assigning display unit.	dd
Invalid file name	The file extension is invalid as program or function option file.	lf
Invalid option, use -tm/-cv/-s/-cm	The option is invalid for setting mode.	md
Invalid value	The input value is invalid.	
Maximum nesting level(5) is exceeded, cannot open file	The nesting level of command file is over 5.	com, cmw
No symbol information	No symbol information (symbol file is not loaded).	sy
Number of steps out of range, use 0 - 65535	Number of steps in single-stepping is invalid.	s, n
This command is not supported in current mode	Trace/coverage commands are not supported in trace/coverage off mode.	td, ts, tf, cv, cvc

Warning message list

Message	Description	Commands involved
64 break addresses are already set	Attempt is made to set more than 64 breakpoints.	bp
Break address already exists	The address is already set as a breakpoint.	bp
Identical break address input	The same address is input more than once in the command line.	bp

6 BITMAP UTILITY

6.1 Overview

The Bitmap Utility (BmpUtil.exe, hereafter BmpUtil) creates bitmap image data (e.g., character data, sprite) for the dot matrix LCD display. The output data is created in the assembly source format with specified labels assigned to it to allow you to assemble data without modifications and link it to the program. This utility is compatible with bit layouts for E0C883xx and E0C888xx series LCD controllers and display memory. (See Section 5.12.5, "Display memory", of the E0C883xx/E0C888xx Technical Manual.)

6.2 Input/Output Files

Figure 6.2.1 shows input/output files for BmpUtil.

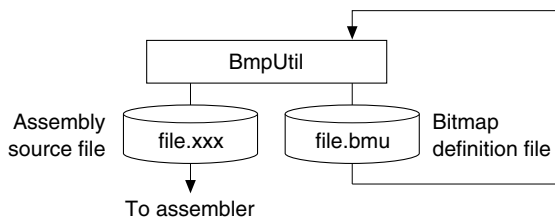


Fig. 6.2.1 BmpUtil input/output files

Bitmap definition file (file_name.bmu)

This file contains the defined bitmap data and the COM/SEG output pattern. This file always has ".bmu" as its extension.

Assembly source file (file_name.xxx)

This text file can be read by the E0C88 Family assembler. Bitmap data is written with DB pseudo-instructions. You can assign an extension to this file.

6.3 Starting and Exiting



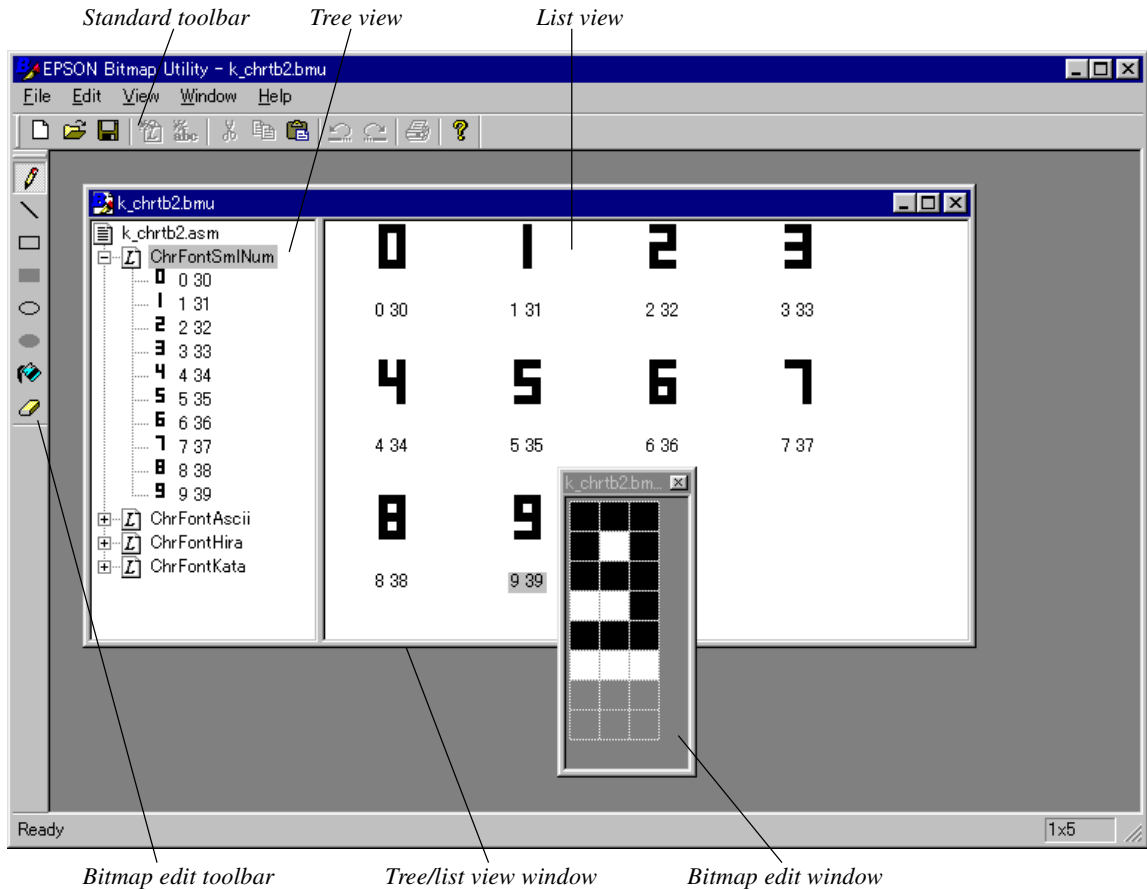
BmpUtil.exe

Double-click this icon to start BmpUtil.

To exit BmpUtil, select [Exit] from the [File] menu.

6.4 Window

The configuration of the BmpUtil windows is shown below:



Tree/list view window

This window is provided for each assembly list file. This window appears when you create a new file or read a bitmap definition file. You can open multiple files for editing. This window is divided into two parts: the tree view pane and list view pane. You can resize these areas by dragging their boundaries.

Tree view

This area displays the tree structure for three hierarchical levels, namely, the name of the assembly source file to be output, its labels, and the data defined in each label. You can use this area to select desired items, to rename a file or label, and to add new or delete current labels or data.

List view

This area displays the list of data defined in the label currently selected in tree view. The name of data is used as the comment added to each piece of data when it is output to the assembly source file. Use this area to select desired data, to rename the data, and to add new or delete current data.

Bitmap edit window

This window is displayed when you double-click a bitmap icon within list view. Use this window to create a new or edit the current bitmap.

Toolbars

Two toolbars, the standard and the bitmap edit toolbars, are provided. The standard toolbar contains commands on the [File] and [Edit] menus, while the bitmap edit toolbar is used to edit bitmap images. The bitmap edit toolbar becomes active when you open a bitmap edit window.

Basic window operations

Closing and activating the window

To close the window, click the [Close] button on the window. The windows being opened are listed on the [Window] menu. Select a window name from this menu to activate the corresponding window, or click anywhere in the desired window. You can also use the [Ctrl] + [Tab] keys to switch the active window from one window to the next. (This window switching technique works only in tree/list view windows.)

Resizing and moving the window

Drag the window boundaries to resize any window. The [Minimize] and [Maximize] buttons work in the same way as in other Windows applications. You can reposition any window within the display by dragging the title bar, but each window can be resized and moved only within the application window. A scroll bar is displayed if the image displayed in a window would otherwise extend beyond the window in any direction.

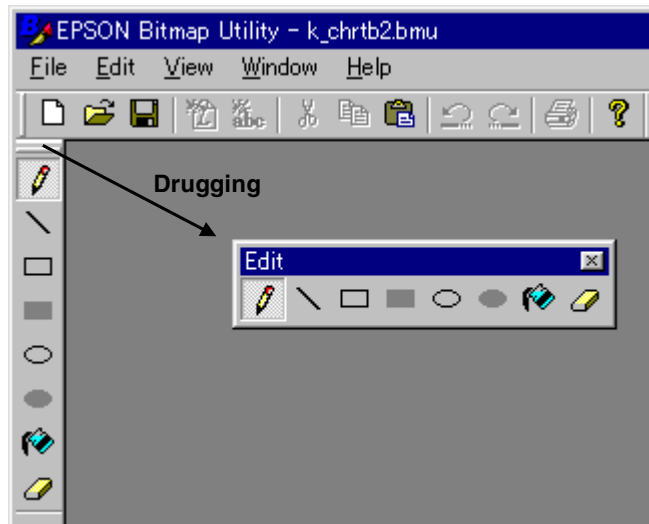
Aligning windows

To align open windows, select [Cascade] or [Tile] from the [Window] menu. (This method of aligning windows works only in tree/list view windows.)

Moving toolbars

To move a toolbar, drag the its heading. Drag the toolbar to the top of the window to arrange toolbar icons horizontally. Drag the toolbar to the extreme left or right of the window to align the toolbar icons vertically. If you drag the toolbar anywhere else in the window (or if you double-click on the heading, the buttons form a floating toolbar.

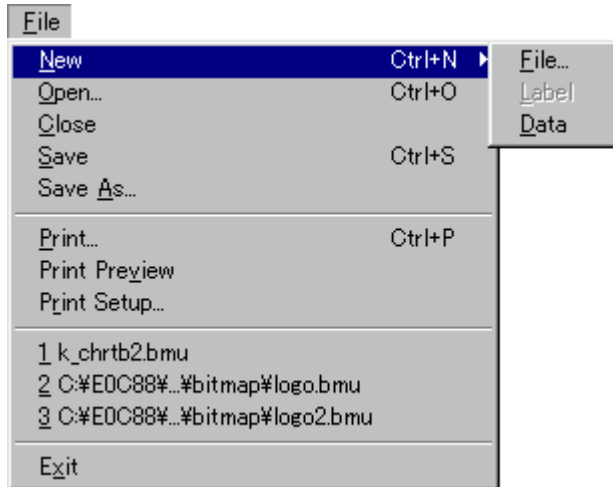
You can hide the toolbars by selecting a command from the [View] menu.



6.5 Menus and Toolbar

6.5.1 Menus

[File] menu



[New] ([Ctrl] + [N])

When the tree/list view window is open

[File...]

Creates a new bitmap definition file and displays a wizard to set new data type and size. A tree/list view window is open when the setting is done.

[Label]

Creates a new label. This option becomes active when the assembly source file name is selected in the tree view.

[Data]

Creates a new bitmap pattern. This option becomes active when one of the labels is selected in the tree view.

[New...] ([Ctrl] + [N]) When the tree/list view window is not open

Creates a new bitmap definition file and displays a wizard to set new data type and size. A tree/list view window is open when the setting is done.

[Open...] ([Ctrl] + [O])

Opens a new bitmap definition file (.bmu).

[Close]

Closes the currently active tree/list view window. This command works in the same way as the [Close] button in the window. If you made changes to the data, this command displays a dialog box that prompts you to save your changes.

[Save] (Ctrl) + [S]

Saves the contents of the active tree/list view window to the bitmap definition file (.bmu) and assembly source file, overwriting the previous files.

[Save As...]

Saves the contents of the active tree/list view window to the bitmap definition file (.bmu) under another name.

[Print...] ([Ctrl] + [P])

Prints all bitmap images, defined in the active tree/list view window, in a list.

[Print Preview]

Displays a print image.

[Print Setup...]

Displays a dialog box to select paper size and printer.

File list

Recently opened files are listed here. You may open a file by selecting from this list.

[Exit]

Exits BmpUtil.

[Edit] menu**[Undo] ([Ctrl] + [Z])**

Cancels up to four of the most-recent editing actions performed within the bitmap edit window.

[Redo] ([Ctrl] + [Y])

Executes up to four of most-recent editing actions canceled with [Undo].

[Cut] ([Ctrl] + [X])

Cuts and copies a selected portion to the clipboard.

[Copy] ([Ctrl] + [C])

Copies a selected portion to the clipboard.

[Paste] ([Ctrl] + [V])

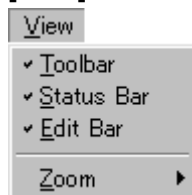
Pastes the portion copied to the clipboard.

[Delete] ([Ctrl] + [D])

Deletes a selected item.

[Rename]

Renames an item selected in the tree/list view window.

[View] menu**[Toolbar]**

Alternately shows or hides the toolbar as you select this menu command.

[Status bar]

Alternately shows or hides the status bar as you select this menu command.

[Edit Bar]

Alternately shows or hides the bitmap edit toolbar as you select this menu command.

[Zoom]

Zooms the dot display size of the bitmap edit window.

[Window] menu**[Cascade]**

Aligns the open tree/list view windows, overlapping them diagonally.

[Tile Horizontally]

Aligns the open tree/list view windows by placing them horizontally.

[Tile Vertically]

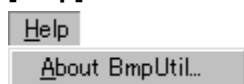
Aligns the open tree/list view windows by placing them vertically.

[Arrange Icons]

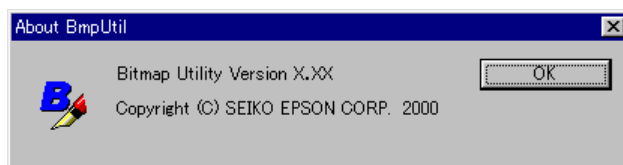
Arranges minimized tree/list view window icons at the lower space of the application window.

Window list

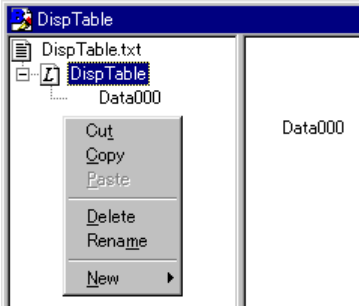
Lists the currently open tree/list view windows. Select a window to activate the corresponding tree/list view window.

[Help] menu**[About BmpUtil...]**

Displays the version information for BmpUtil.



Pop-up menu



To display a pop-up menu, right-click with the mouse pointer positioned within the tree view or list view pane. Depending on the item selected, different commands in the menu become active. Commands in the menu work the same way as the others already mentioned.

6.5.2 Standard Toolbar Buttons



[New File] button

Creates a new bitmap definition file and displays a wizard to set new data type and size. A tree/list view window is open when the setting is done.



[Open] button

Opens a bitmap definition file (.bmu).



[Save] button

Saves the contents of the active tree/list view window to the bitmap definition file (.bmu) and assembly source file, overwriting the previous files.



[New Label] button

Creates a new label. This button becomes active when the assembly source file name is selected in the tree view.



[New Data] button

Creates a new bitmap pattern. This button becomes active when one of the labels is selected in the tree view.



[Cut] button

Cuts and copies a selected portion to the clipboard.



[Copy] button

Copies a selected portion to the clipboard.



[Paste] button

Pastes the portion copied to the clipboard.



[Undo] button

Cancels up to four of the most recent editing actions performed within the bitmap edit window.



[Redo] button

Executes up to four of the most recent editing actions canceled with [Undo].



[Print] button

Prints all bitmap images defined in the active tree/list view window in a list.



[About] button

Displays version information for BmpUtil.

6.5.3 *Bitmap Edit Toolbar Buttons*

Use these buttons to create new bitmap images or to correct current bitmap images. They become active when the bitmap edit window is open.

**[Pen] button**

Sets or resets dots to white or black.

**[Line] button**

Traces a straight line.

**[Rectangle] button**

Draws a rectangular frame.

**[Filled Rectangle] button**

Draws a filled rectangle.

**[Ellipse] button**

Draws an elliptical frame.

**[Filled Ellipse] button**

Draws a filled ellipse.

**[Fill] button**

Fills an area in which all dots are the same color with the selected color.

**[Erase] button**

Erases a selected area by filling it with white.

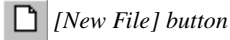
6.6 Creating Bitmap Data

This section explains how to create a bitmap.

6.6.1 New Data Wizard

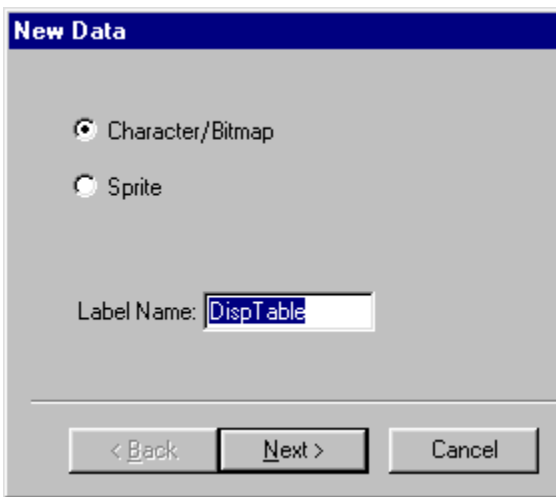
Starting the new data wizard

To create new bitmap data, select [New...] (or [New-File]*) from the [File] menu or click the [New File] button. A [New Data] dialog box appears.



- * The [File] menu shows [New...] when a tree/list view window is not open. The same menu shows [New], indicating that there is a subordinate menu, when tree/list view window is open.

[New Data] dialog box



Use this dialog box to select the type of data you want to create and to specify a label.

Data type

For data type, select either Character/Bitmap or Sprite.

Character/Bitmap

Select this option to create a bitmap image for the dot matrix LCD or font data for the character generator.

Sprite

Select this option to create a sprite data of 16 × 16 dots pixels consisting of bitmap and clear planes.

The data thus created can only be used with models equipped with sprite-capable display controllers.

Note: No current models are sprite-capable.

Label

Type a label in the [Label Name] text box. This is the label that will be used by the application program to access data. Since the label will be written in the assembly source file to be output, be sure to use characters and symbols that the assembler can identify. Avoid exceeding the maximum number of characters provided for the assembler.

Note that the entered name will also be used as the assembly source file name.

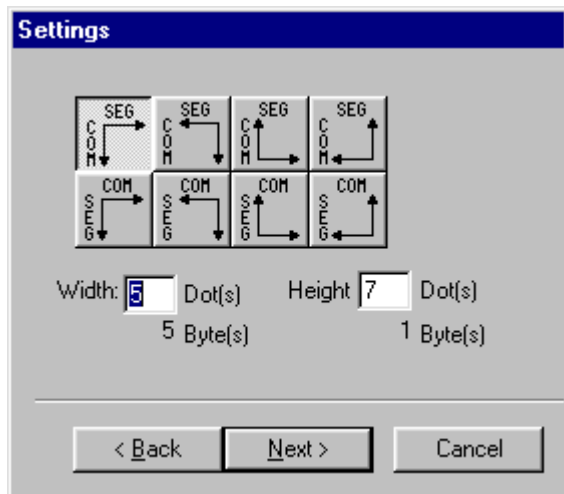
Each label or assembly source file name specified in this dialog box can be changed later in tree view. You can add labels at a later point. In other words, you can define multiple labels for a single assembly source file.

After selecting the data type and specifying the label, click the [Next>] button (when [Character/Bitmap] is selected).

If you select [Sprite], you do not need to specify any other information, such as bitmap size. Click the [Finish] button — which appears in place of the [Next>] button — to exit the New Data wizard. The tree/list view pane is displayed.

[Settings] dialog box (when [Character/Bitmap] is selected)

Select [Character/Bitmap] and click the [Next>] button in the [New Data] dialog box to display the [Settings] dialog box. If you change your mind and wish to select [Sprite], click the [<Back] button to return to the [New Data] dialog box.



Use this dialog box to specify the bitmap image size and COM/SEG layout.

COM/SEG layout

Use one of the eight buttons to select a COM/SEG layout according to the COM/SEG port assignment for the dot-matrix LCD so that the created bitmap is properly oriented when displayed. Keep in mind that you cannot change this option once the data is created.


Bitmap image size

Type a horizontal and vertical number of dots for the bitmap image respectively in the [Width] and [Height] text boxes.

Under each text box, you can find the number of bytes required in each direction. Multiply these numbers to calculate the number of bytes required for each piece of data.

Example of settings

Figure 6.6.1.1 shows an example of creating character generator data of 5×7 dots. (default settings)

COM/SEG layout: 
Width: 5 dots, Height: 7 dots

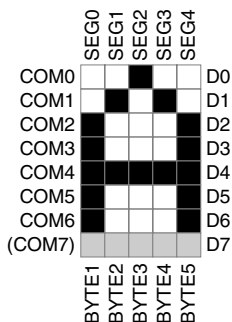


Fig. 6.6.1.1 Example of bitmap image size settings

The COM and SEG numbers that appear in the example merely indicate that an arrangement in ascending order in each of the directions indicated by arrows on the selected button. These numbers can vary, depending on the display memory addresses to which data is actually written.

The data created (bytes and bits) is assigned to data bits by ascending order of COM numbers.

Since data is assigned in the COM direction in units of one byte, some data bits will become invalid if you specify a number of dots that is not a multiple of eight for this direction. In the example, the data is assigned to COM0 through 6 while D7 is masked. You can change this assignment and mask the D0 in place of the D7 using the [Offset] dialog box, shown next.

See "[Output] dialog box" in a later section for information on specifying data output sequence when you have more than eight dots for the COM direction.

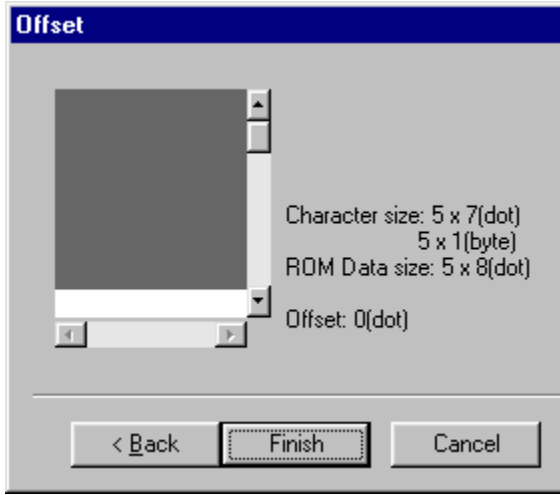
Click the [Next>] button after making the required selections.

Depending on the size you selected, no more dialog boxes may be displayed. In this case, the [Finish] button appears in place of the [Next>] button. Click the [Finish] button to exit the New Data wizard. The tree/list view window is displayed.

[Offset] dialog box (when [Character/Bitmap] is selected)

The [Offset] dialog box appears if you specify a number of dots in the [Settings] dialog box that is not a multiple of eight for the COM direction. Click the [<Back] button to return to the [Settings] dialog box.

Note: The [Offset] dialog box does not appear when you specify a number of dots that is a multiple of eight for the COM direction.



Use this dialog box to specify the data offset value in the COM direction.

Because display data is assigned in the COM direction in units of eight dots (one byte), some COM output data is not used if you specify a number of dots not a multiple of eight for this direction. By default, bitmap assignment starts with COM0. For this reason, the data to be assigned to the largest COM number will be invalid.

Data will be assigned to the gray area within the box enclosed by the scroll bars, while the white area remains unused. Use the scroll bars to move the unused area to COM0. The [Offset] value changes when you move the unused area.

Suppose, for example, that you create data of 5 x 7 dots as described in the previous example. The data arrangement then changes as follows:

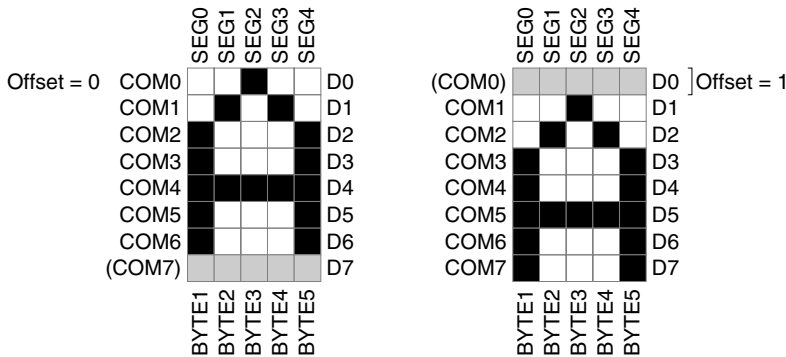


Fig. 6.6.1.2 Example of offset setting

This setting creates final data with 0 selected as the unused bit.

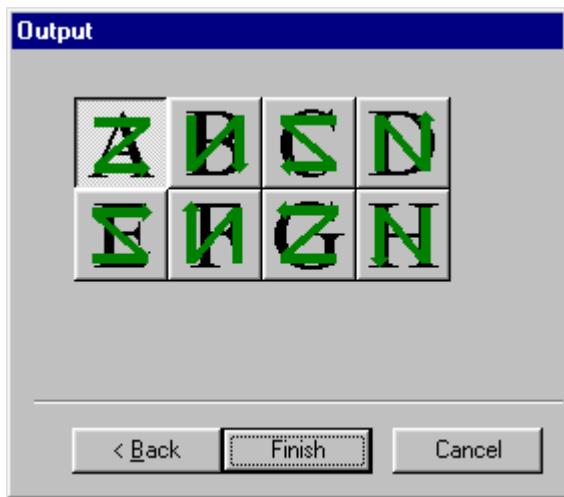
Click the [Next>] button after specifying an offset value.

Depending on the size selected in the [Settings] button (when you specify eight or fewer dots for the COM direction), no more dialog boxes may be displayed. In this case, the [Finish] button is displayed in place of the [Next>] button. Click the [Finish] button to exit the New Data wizard. The tree/list view window is displayed.

[Output] dialog box (when [Character/Bitmap] is selected)

The [Output] dialog box appears following the [Setting] or [Offset] dialog box if you specify nine dots or more for the COM direction in the [Settings] dialog box. Click the [<Back] button to return to the previous dialog box.

Note: The [Output] dialog box does not appear if you specify eight or fewer dots for the COM direction.



Use this dialog box to select one data output sequence from the eight options (patterns A to H). Data is output to the assembly source file in the sequence shown on the selected button.

Figure 6.6.1.3 shows an example in which bitmap data of 16 × 16 dots is output.

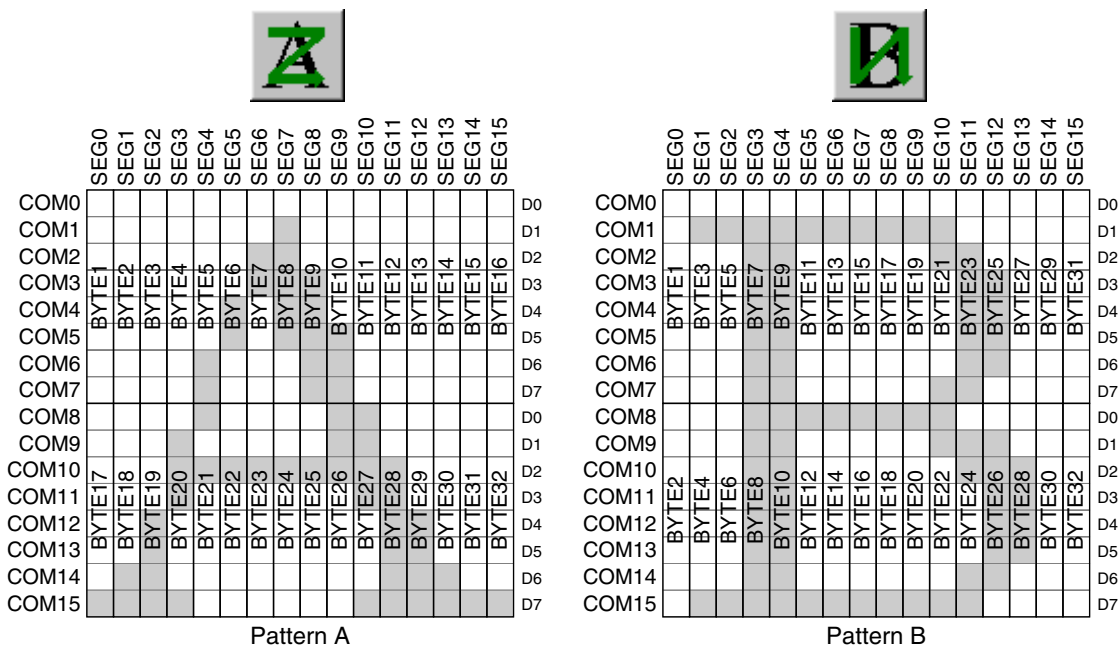


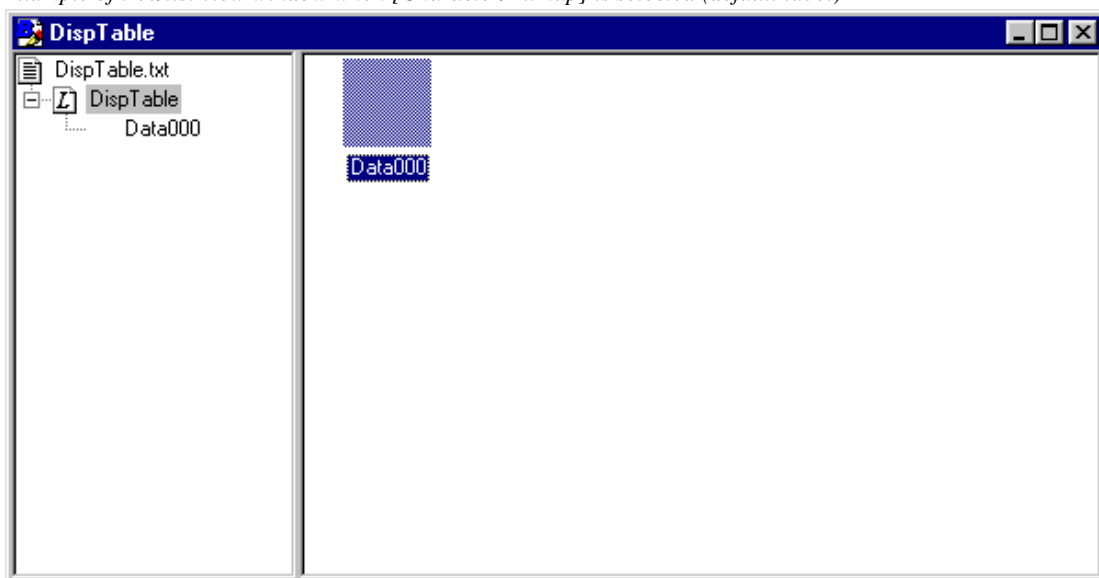
Fig. 6.6.1.3 Data output pattern

After selecting a data output pattern, click the [Finish] button to exit the New Data wizard. The tree/list view window is displayed.

Tree/list view window displayed when new data is created

Exiting the New Data wizard displays the tree/list view window shown below.

Example of tree/list view window when [Character/Bitmap] is selected (default label)

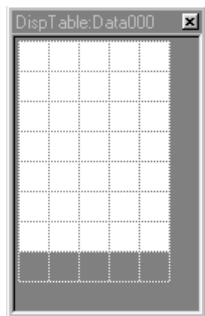


The assembly source file and label appear in the list view with the names specified in the [New Data] dialog box. Under the label, you can see that new data (blank data) has been created with the name of Data000 (SP000 when [Sprite] is selected). Display the data by double-clicking the label. See Section 6.6.3, "Editing Functions", for information on changing these names or adding and deleting labels and data.

6.6.2 Creating Bitmap Images

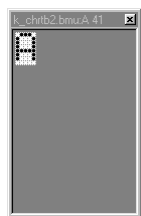
This section explains how to create bitmap images.

Bitmap edit window

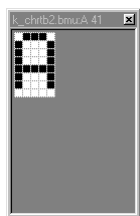


A bitmap edit window is displayed when you double-click a bitmap icon in list view. You can use this window to create and edit bitmap images. Creating a new bitmap image displays a blank bitmap icon. In this case, double-click a desired position above the data name.

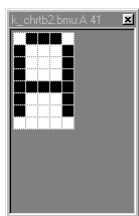
The gray area within the window cannot be edited. When you create a new bitmap image, the area that can be edited appears white (dots OFF). By default, the bitmap initially appears with its dots enlarged at the maximum ratio (500%). You can change sizes using [Zoom] in the [View] menu.



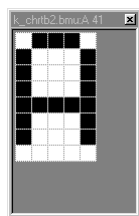
100%



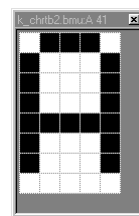
200%



300%

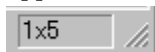


400%



500% (default)

When you place the mouse pointer (which turns into the button selected from the bitmap edit toolbar) anywhere in the bitmap edit window, the pointer position (coordinates with the 0 position set at the upper left corner of the bitmap) is displayed in the status bar in X × Y format.



After creating a bitmap image using the editing tools described in a later section, click the window [Close] button to complete your bitmap edit task.

Bitmap editing tools

Use the following buttons in the bitmap edit toolbar to edit bitmap images:



[Pen] button

If you click the left mouse button a dot is black. And if you click the right mouse button a dot is white. By dragging the mouse the trace can be erased or comes alive.



[Line] button

Traces a line connecting the start and end points when you drag the mouse. Black is selected as the line color when you use the left mouse button. White is selected as the line color when you use the right mouse button.



[Rectangle] button

Draws a rectangular frame with the start and end points set as two corners of a frame, diagonally opposite each other, when you drag the mouse. Black is selected as the frame color when you use the left mouse button. White is selected as the frame color when you use the right mouse button.



[Filled Rectangle] button

Draws a filled rectangle with the start and end points set as two corners of a rectangle, diagonally opposite each other, when you drag the mouse. Black is selected as the border and fill color when you use the left mouse button. White is selected as the border and fill color when you use the right mouse button.



[Ellipse] button

Draws an elliptical frame passing through the start and end points when you drag the mouse. Black is selected as the frame color when you use the left mouse button. White is selected as the frame color when you use the right mouse button.



[Filled Ellipse] button

Draws a filled ellipse whose border passes through the start and end points when you drag the mouse. Black is selected as the border and fill color when you use the left mouse button. White is selected as the border and fill color when you use the right mouse button.



[Fill] button

Fills an area in which all dots are the same color with a selected color when you click one of the dots in that area. Black is selected as the fill color when you use the left mouse button. White is selected as the fill color when you use the right mouse button.



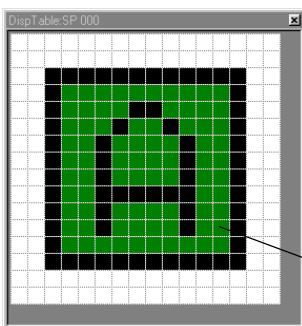
[Erase] button

Fills a rectangular area (with the start and end points set as two corners of the rectangle that are diagonally opposite each other), drawn by dragging the mouse, with white.

All dragging operations other than that for the [Fill] button can be canceled by pressing the [ESC] key before you release the mouse button (as you drag the mouse).

You can also cancel up to four of your most recent actions by selecting [Undo] or [Redo] from the [Edit] menu or the [Undo] or [Redo] button from the standard toolbar.

Editing sprite data



Use the above-mentioned editing tools to edit bitmap images for sprite data. When editing sprite data, you can also set transparent dots. To set transparent dots, first select the desired tool from the above-mentioned editing tools, then enable dots (using the left mouse button) while holding down the [Ctrl] key. Dots specified as transparent dots appear green. To cancel transparent dots, disable dots (using the right mouse button) while holding down the [Ctrl] key.

(Green) clear dots

6.6.3 Editing Functions

This section describes editing functions other than the ones for bitmap image creation described in the previous section.

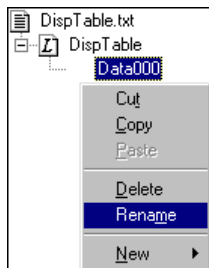
Saving and read files

After creating data, save the data to a file by selecting [Save] or [Save As...] from the [File] menu (or by clicking the [Save] button), as in other Windows applications. BmpUtil outputs a bitmap definition file (.bmu) and assembly source file (name displayed in the tree view). To add data or to make changes to existing data, open the corresponding bitmap definition file by selecting [Open...] from the [File] menu (or by clicking the [Open] button). You cannot open an assembly source file.

Renaming a file

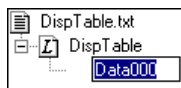
To rename a bitmap definition file (.bmu), select [Save As...] from the [File] menu and save the file under another name.

To rename the assembly source file, a label, or data, follow the steps given below.



1. Click a desired item.
2. Re-click the same item (as opposed to double-clicking the item — which does not have the desired effect.) You can also rename an item by selecting [Rename] from the [Edit] menu or from the pop-up menu which appears by right-clicking.

You can now rename the desired item. To change the entire name, simply begin typing. To change a part of the current name, select that part and type the new characters.



Notes:

- If you rename the assembly source file, a new assembly source file is created under a new name after you save the first piece of data after the renaming of the assembly source file.

- Use assembler-readable characters for the assembly source file and label names. Avoid exceeding the maximum number of characters provided for the assembler.
- Data names are used merely as comments to identify data within the assembly source file to be output. They are not used to perform control operations within the program.

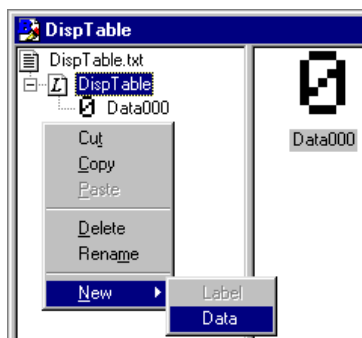
Adding, deleting, and copying data

Adding data

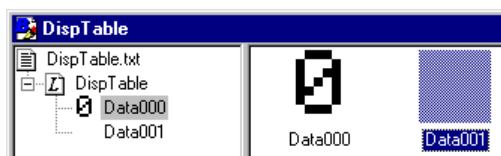
To add data, follow the steps given below.



1. Select a label to which to add data in the tree view. The list of data defined in the selected label is displayed in the list view.



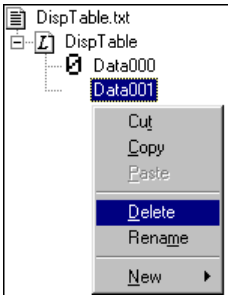
2. Select [New - Data] from the [File] menu, or right-click and select [New - Data] from the pop-up menu. New data is added to the tree view and the list view. Rename the data as necessary.



- * When creating new data based on existing data, you can add this new data by copying and pasting existing data.

Deleting data

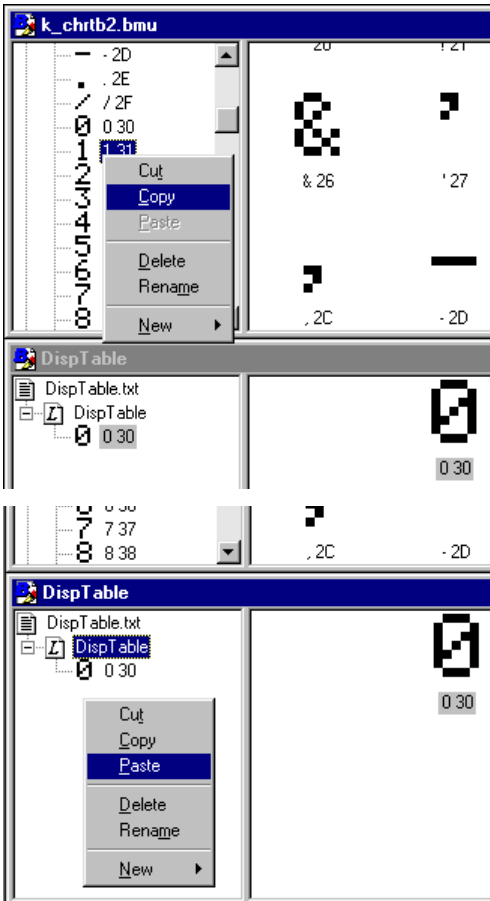
To delete data, follow the steps given below.



1. Select data you wish to delete from the list or the tree views. Use the [Shift] or [Ctrl] key to select multiple pieces of data.
2. Select [Delete] from the [Edit] menu, or right-click and select [Delete] from the pop-up menu. The selected data is deleted from the tree view and the list view.

Cutting, copying, and pasting data

To cut (or copy) and paste data, follow the steps given below.



1. Select the data you wish to cut or copy from the list or the tree view. Use the [Shift] or [Ctrl] key to select multiple pieces of data.
2. Select [Cut] (or [Copy]) from the [Edit] menu, or right-click and select [Cut] (or [Copy]) from the pop-up menu. (You can also use the keyboard shortcut [Ctrl] + [X] or [Ctrl] + [C].)

3. Select a label to which to paste the data from the tree view. (You can also paste the data to a label in other window.) Then select [Paste] from the [Edit] menu, or right-click and select [Paste] from the pop-up menu. (You can also use the keyboard shortcut [Ctrl] + [V].)

The data is pasted to the tree and the list views.



If the data size in the destination label differs from the source data size, a confirmation dialog box appears to ask you whether you wish to continue. You can choose to cancel pasting. If you choose to continue and paste the data, blank dots are added to the right-hand and bottom sections of the pasted data if the data size in the destination label exceeds the source data size. If the data size in the destination label is less than the source data size, the pasted data is clipped.

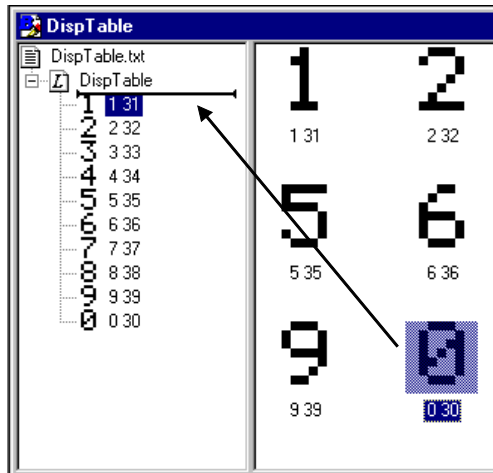
Since the data you cut or copy is saved in the clipboard, you can paste it repeatedly.

When you select a label and paste the data, the data is copied as the last piece of data in that label. To paste the data somewhere between the current pieces of data, select a desired piece of data. The data is pasted before the selected data.

Dragging and dropping data

You can move or copy data by dragging and dropping it, as follows.

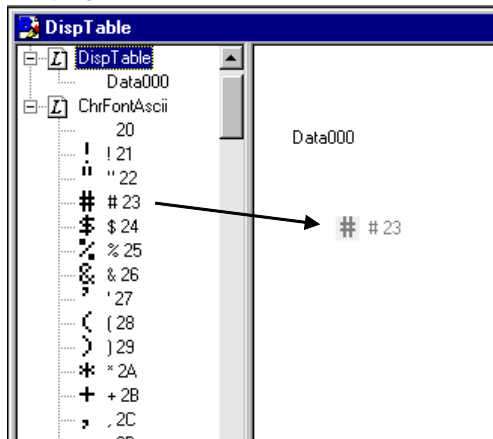
Moving data within the same label (rearranging data)



1. Select a desired data item from the tree view.
2. Drag and drop the data within the same label. As you drag the data, another pointer appears in tree view to indicate locations in tree view where you may position the data.

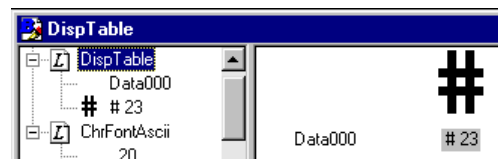
Note: Dropping the data within another label copies rather than moves the data. You are also copying the data if you drag it from the list or the tree view and drop it into another.

Copying data 1 (from the tree view to the list view)

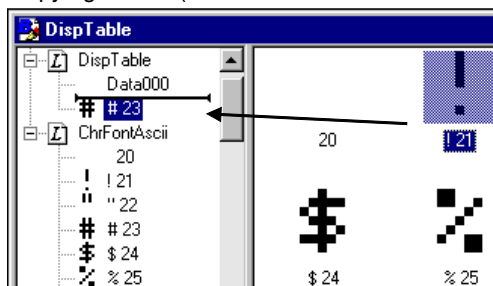


1. In the tree view, select a label to which to copy data. The list of data defined in the destination label is displayed in the list view.
2. Drag the desired data from the tree view to the list view. Drop it into the list view pane to copy the data.

Copying data in this way adds it as the last piece of data in the label.



Copying data 2 (from the list view to the tree view)



1. In the tree view, select a label from which to copy data. The list of data defined in that label is displayed in the list view.
2. Drag the desired data from the list view and drop it into the tree view. As you drag the data, another pointer appears in the tree view to indicate locations in the tree view pane where you can position the data.

Use this technique to copy data to a desired location.

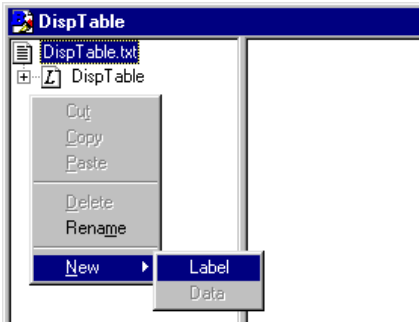
When using data copying technique 1 or 2, if the destination data size differs from the source data size, a confirmation dialog box will prompt you for confirmation to continue. You can choose to cancel copying. If you choose to continue and copy the data, blank dots are added to the right-hand and bottom sections of the copied data if the destination data size exceeds the source data size. If the destination data size is less than the source data size, the copied data is clipped.

When two or more windows are open, you can copy data between windows by dragging and dropping.

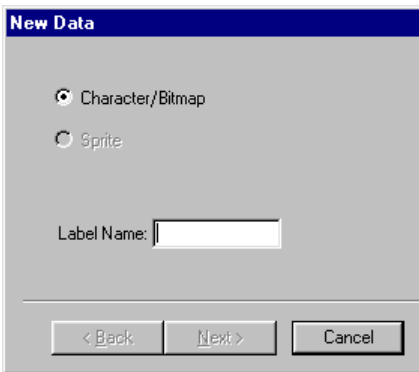
Adding, deleting, and copying a label

Adding a label

To add a label, follow the steps given below.



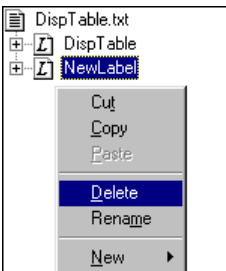
1. Click the file name displayed at the uppermost line of the tree view pane. No data is currently displayed in the list view.
2. Select [New - Label] from the [File] menu, or right-click and select [New - Label] from the pop-up menu.



3. A wizard for creating new labels appears. You can use this wizard in the same way as the wizard described in "New Data Wizard" of Section 6.6.1. Specify a label name and select the data size and so on in this wizard. When you finish, a new label containing a single piece of blank data is created.
- * When you create a new label based on an existing label (including defined data), you can add a label by copying and pasting the existing label.

Deleting a label

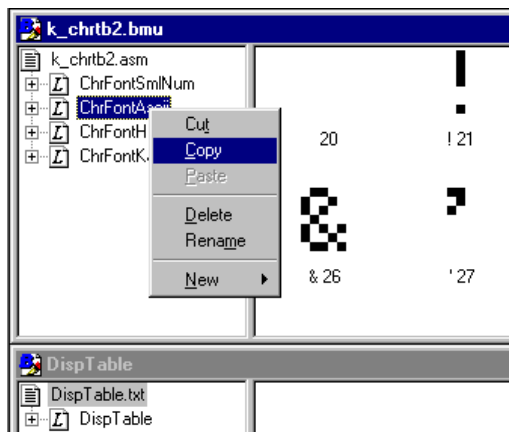
To delete a label, follow the steps given below.



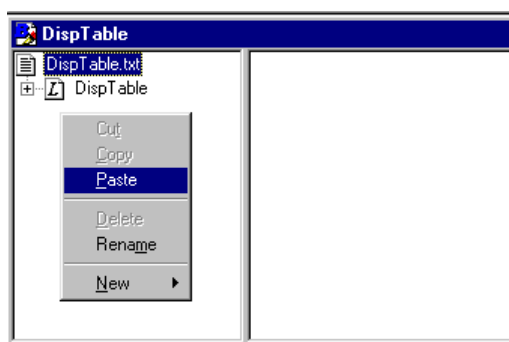
1. In the tree view, select a label you wish to delete.
2. Select [Delete] from the [Edit] menu, or right-click and select [Delete] from the pop-up menu.
The selected label and all data defined in the label are deleted from the tree view pane.

Cutting, copying and pasting a label

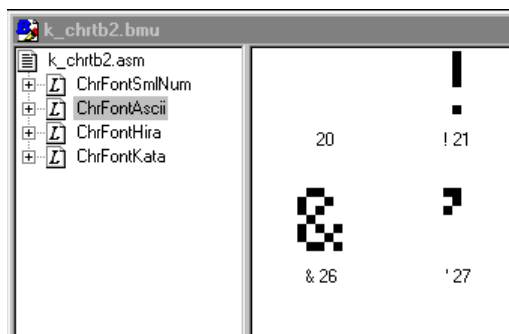
You can cut (or copy) and paste a label, including data defined in that label. To do this, follow the steps given below.



1. In the tree view, select a label you wish to cut or copy.
2. Select [Cut] (or [Copy]) from the [Edit] menu, or right-click and select [Cut] (or [Copy]) from the pop-up menu. (You can also use the keyboard shortcut [Ctrl] + [X] or [Ctrl] + [C].)

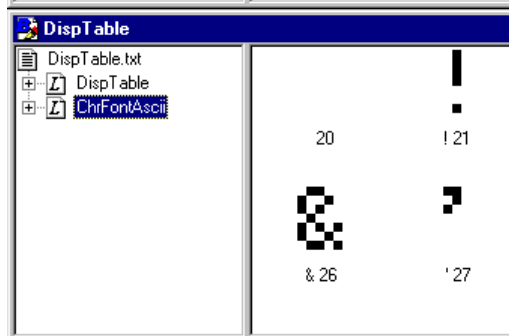


3. In the tree view, select the assembly source file name. (You can also select file names in other windows.) Then select [Paste] from the [Edit] menu, or right-click and select [Paste] from the pop-up menu. (You can also use the keyboard shortcut [Ctrl] + [V].)



The label is pasted at the end of the tree view. The list of data defined in the label is displayed in the list view.

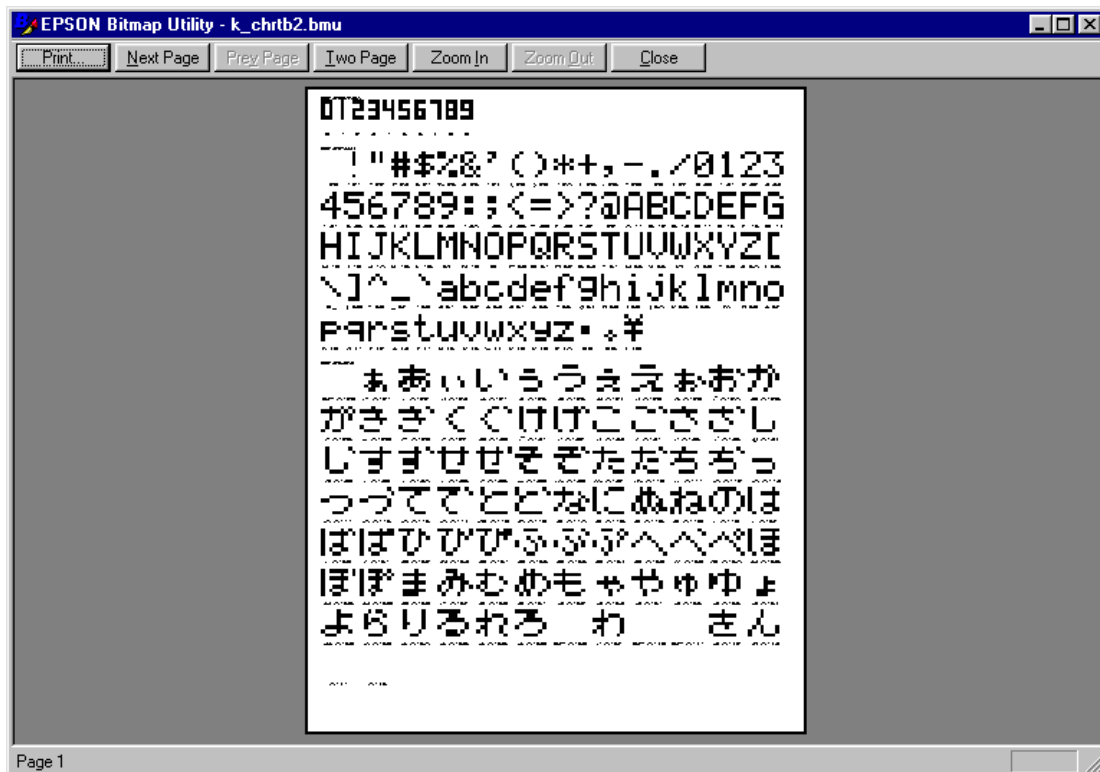
Since the cut or copied label and data are saved in the clipboard, you can paste them repeatedly.



Printing data

Print the created data by selecting [Print] from the [File] menu, or by clicking the [Print] button in the toolbar.

All data defined in the open file are printed. You can view the print image by selecting [Print Preview] from the [File] menu.



6.7 Assembly Source File

This section shows an example of an assembly source file. This file contains specified labels, each followed by bitmap data (in byte unit) arranged in the specified output sequence. The name assigned to each piece of data is used as a comment. Load the created file into the user program or link it to the program together with other objects after assembling.

Example)

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
```

COM/SEG layout:



Width: 3 dots, Height: 6 dots

```
! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H
I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o p q
r s t u v w x y z . : ;
```

```
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
```

COM/SEG layout:



Width: 5 dots, Height: 8 dots

6 BITMAP UTILITY

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
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

EPSON International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.

- HEADQUARTERS -

1960 E. Grand Avenue
El Segundo, CA 90245, U.S.A.
Phone: +1-310-955-5300 Fax: +1-310-955-5400

- SALES OFFICES -

West

150 River Oaks Parkway
San Jose, CA 95134, U.S.A.
Phone: +1-408-922-0200 Fax: +1-408-922-0238

Central

101 Virginia Street, Suite 290
Crystal Lake, IL 60014, U.S.A.
Phone: +1-815-455-7630 Fax: +1-815-455-7633

Northeast

301 Edgewater Place, Suite 120
Wakefield, MA 01880, U.S.A.
Phone: +1-781-246-3600 Fax: +1-781-246-5443

Southeast

3010 Royal Blvd. South, Suite 170
Alpharetta, GA 30005, U.S.A.
Phone: +1-877-EEA-0020 Fax: +1-770-777-2637

EUROPE

EPSON EUROPE ELECTRONICS GmbH

- HEADQUARTERS -

Riesstrasse 15
80992 Munich, GERMANY
Phone: +49-(0)89-14005-0 Fax: +49-(0)89-14005-110

- GERMANY -

SALES OFFICE

Altstadtstrasse 176
51379 Leverkusen, GERMANY
Phone: +49-(0)2171-5045-0 Fax: +49-(0)2171-5045-10

- UNITED KINGDOM -

UK BRANCH OFFICE

Unit 2.4, Doncastle House, Doncastle Road
Bracknell, Berkshire RG12 8PE, ENGLAND
Phone: +44-(0)1344-381700 Fax: +44-(0)1344-381701

- FRANCE -

FRENCH BRANCH OFFICE

1 Avenue de l'Atlantique, LP 915 Les Conquerants
Z.A. de Courtaboeuf 2, F-91976 Les Ulis Cedex, FRANCE
Phone: +33-(0)1-64862350 Fax: +33-(0)1-64862355

ASIA

- CHINA -

EPSON (CHINA) CO., LTD.

28F, Beijing Silver Tower 2# North RD DongSanHuan
ChaoYang District, Beijing, CHINA
Phone: 64106655 Fax: 64107319

SHANGHAI BRANCH

4F, Bldg., 27, No. 69, Gui Jing Road
Caohejing, Shanghai, CHINA
Phone: 21-6485-5552 Fax: 21-6485-0775

- HONG KONG, CHINA -

EPSON HONG KONG LTD.

20/F., Harbour Centre, 25 Harbour Road
Wanchai, HONG KONG
Phone: +852-2585-4600 Fax: +852-2827-4346
Telex: 65542 EPSCO HX

- TAIWAN -

EPSON TAIWAN TECHNOLOGY & TRADING LTD.

10F, No. 287, Nanking East Road, Sec. 3
Taipei, TAIWAN
Phone: 02-2717-7360 Fax: 02-2712-9164
Telex: 24444 EPSONTB

HSINCHU OFFICE

13F-3, No. 295, Kuang-Fu Road, Sec. 2
HsinChu 300, TAIWAN
Phone: 03-573-9900 Fax: 03-573-9169

- SINGAPORE -

EPSON SINGAPORE PTE., LTD.

No. 1 Temasek Avenue, #36-00
Millenia Tower, SINGAPORE 039192
Phone: +65-337-7911 Fax: +65-334-2716

- KOREA -

SEIKO EPSON CORPORATION KOREA OFFICE

50F, KLI 63 Bldg., 60 Yoido-dong
Yongdeungpo-Ku, Seoul, 150-763, KOREA
Phone: 02-784-6027 Fax: 02-767-3677

- JAPAN -

SEIKO EPSON CORPORATION

ELECTRONIC DEVICES MARKETING DIVISION

Electronic Device Marketing Department

IC Marketing & Engineering Group

421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-(0)42-587-5816 Fax: +81-(0)42-587-5624

ED International Marketing Department Europe & U.S.A.

421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-(0)42-587-5812 Fax: +81-(0)42-587-5564

ED International Marketing Department Asia

421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-(0)42-587-5814 Fax: +81-(0)42-587-5110



In pursuit of “**Saving**” **Technology**, Epson electronic devices.
Our lineup of semiconductors, liquid crystal displays and quartz devices
assists in creating the products of our customers’ dreams.
Epson IS energy savings.

EPSON

SEIKO EPSON CORPORATION
ELECTRONIC DEVICES MARKETING DIVISION

■ EPSON Electronic Devices Website
<http://www.epson.co.jp/device/>

Issue JUNE 2000, Printed in Japan  A