

**OKI**

# **DT63K Debugger and DTS63K Simulator**

*User's Manual*

---

**Program Development Support Software**

Third Edition  
March 1999

## NOTICE

1. The information contained herein can change without notice owing to product and/or technical improvements. Before using the product, please make sure that the information being referred to is up-to-date.
2. The outline of action and examples for application circuits described herein have been chosen as an explanation for the standard action and performance of the product. When planning to use the product, please ensure that the external conditions are reflected in the actual circuit and assembly designs.
3. When developing and evaluating your product, please use our product below the specified maximum ratings and within the specified operating ranges including, but not limited to, operating voltage, power dissipation, and operating temperature.
4. **OKI assumes no responsibility or liability whatsoever for any failure or unusual or unexpected operation resulting from misuse, neglect, improper installation, repair, alteration or accident, improper handling, or unusual physical or electrical stress including, but not limited to, exposure to parameters beyond the specified maximum ratings or operation outside the specified operating range.**
5. Neither indemnity against nor license of a third party's industrial and intellectual property right, etc. is granted by us in connection with the use of product and/or the information and drawings contained herein. No responsibility is assumed by us for any infringement of a third party's right which may result from the use thereof.
6. The products listed in this document are intended only for use in development and evaluation of control programs for equipment and systems. These products are not authorized for other use (as an embedded device and a peripheral device).
7. Certain products in this document may need government approval before they can be exported to particular countries. The purchaser assumes the responsibility of determining the legality of export of these products and will take appropriate and necessary steps at their own expense for these.
8. No part of the contents contained herein may be reprinted or reproduced without our prior permission.
9. MS-DOS, Microsoft, Windows and WindowsNT are registered trademarks of Microsoft Corporation. Other product names and company names are trademarks of registered trademarks of their respective owners.

# Contents

<b>1. Introduction.....</b>	<b>1</b>
1.1 About the Products.....	1
1.1.1 Common User Interface .....	1
1.1.2 DT63K Debugger Features.....	2
1.1.3 DTS63K Simulator Features.....	2
1.2 Structure of This Manual .....	3
1.3 Notational Conventions.....	5
<b>2. Before You Start .....</b>	<b>6</b>
2.1 System Configuration.....	6
2.2 Using Products in Development Process.....	8
2.3 Installing DT63K and DTS63K.....	10
<b>3. Loading, Preparation, and Projects .....</b>	<b>11</b>
3.1 Loading the Software.....	11
3.2 Preparing for Debugging.....	13
3.2.1 Environment Settings.....	13
3.3 Preparing for Emulation .....	13
3.3.1 Specifying Target Device .....	13
3.3.2 Specifying Emulator Options.....	13
3.4 Preparing for Simulation .....	15
3.4.1 Loading LCD Data File.....	15
3.4.2 Assigning Ports .....	15
3.4.3 Specifying Simulator Options.....	19
3.5 Loading/Saving Program Code.....	22
3.5.1 Loading Program Code .....	22
3.5.2 Saving Program Code .....	23
3.5.3 Comparing Code Memory to File.....	24
3.6 Project Files (.d63 and .s63).....	25

<b>4. Program Execution .....</b>	<b>26</b>
4.1 Program Execution.....	26
4.2 Resets.....	28
<b>5. Break Functions .....</b>	<b>29</b>
5.1 Breakpoints.....	29
5.1.1 Setting Breakpoints .....	29
5.1.2 Listing and Suspending Breakpoints .....	30
5.2 Executing to Cursor .....	31
5.3 Forced Break.....	31
5.4 DT63K Break Conditions .....	31
5.4.1 Break Condition Check Boxes .....	32
5.4.2 Break Parameters .....	33
5.5 DTS63K Break Conditions .....	37
5.5.1 Break Condition Check Boxes .....	37
5.5.2 Break Parameters .....	38
<b>6. Displaying/Modifying Memory and Registers.....</b>	<b>40</b>
6.1 Program Counter .....	40
6.2 Registers .....	40
6.3 Peripheral Registers (SFRs).....	42
6.4 Code, Data, and External Memory.....	43
<b>7. Watch Windows .....</b>	<b>45</b>
7.1 Watch Window Usage .....	45
7.2 Candidates .....	45
7.3 Adding Items.....	46
7.4 Saving Watch List.....	47
7.5 Loading Watch List .....	47
7.6 Modifying Watch List .....	47
<b>8. Tracing .....</b>	<b>48</b>

8.1	Overview .....	48
8.2	Trace Windows.....	48
8.3	Tracing Options.....	51
8.4	Saving Trace Data.....	52
8.5	Searching Trace Data .....	53
<b>9.</b>	<b>Cycle Counter .....</b>	<b>54</b>
9.1	Overview .....	54
9.2	Options .....	55
<b>10.</b>	<b>Source Level Debugging .....</b>	<b>57</b>
10.1	Source Windows .....	57
10.2	Links to Other Windows .....	58
<b>11.</b>	<b>Symbolic Debugging and Input Formats .....</b>	<b>59</b>
11.1	Symbols .....	59
11.2	Expressions.....	60
11.2.1	Constants .....	61
11.2.2	Operators .....	62
11.3	Value Lists .....	62
<b>12.</b>	<b>Other Functions .....</b>	<b>63</b>
12.1	Macros .....	63
12.2	Log Window.....	64
12.3	On-Line Help.....	64
12.4	Version Information .....	65
12.5	Reading/Verifying EPROMs .....	66
<b>13.</b>	<b>DTS63K Simulation Functions .....</b>	<b>67</b>
13.1	Instruction Execution .....	67
13.2	Resets .....	68
13.3	Onboard Peripherals.....	69

## Contents

13.4 Interrupt Simulation .....	72
13.5 LCD Simulation.....	72
<b>14. LCDTOOL Utility.....</b>	<b>74</b>
14.1 Overview .....	74
14.2 Procedure.....	74
14.2.1 Creating .BMP File.....	74
14.2.2 Loading .BMP Data .....	75
14.2.3 Creating Dot Matrix.....	75
14.2.4 Making Assignments .....	77
14.2.5 Saving LCD Data File.....	79
14.3 Editing LCD Data Files .....	81
14.4 Other Functions.....	82
14.4.1 Zoom.....	82
14.4.2 Operating Settings.....	82
14.4.3 On-Line Help .....	83
<b>15. Appendices.....</b>	<b>84</b>
15.1 DT63K Debugger .....	84
15.1.1 Menus .....	84
15.1.2 Context Menus .....	86
15.1.3 Tool Bar .....	88
15.1.4 Shortcut Keys .....	89
15.1.5 Macro Commands.....	90
15.2 DTS63K Simulator .....	93
15.2.1 Menus .....	93
15.2.2 Context Menus .....	95
15.2.3 Tool Bar .....	97
15.2.4 Shortcut Keys .....	99
15.2.5 Macro Commands.....	99
15.3 LCDTOOL Utility .....	102
15.3.1 Menus .....	102
15.3.2 Context Menu .....	102
15.3.3 Tool Bar .....	103

15.3.4	Shortcut Keys .....	103
<b>15.4</b>	<b>Script Command Reference .....</b>	<b>104</b>
15.4.1	Projects .....	104
15.4.2	Display/Change CPU Internals.....	104
15.4.3	Code Memory .....	105
15.4.4	Data Memory .....	108
15.4.5	External Data Memory .....	109
15.4.6	Execution .....	110
15.4.7	Resets.....	112
15.4.8	Breaks.....	113
15.4.9	Tracing.....	114
15.4.10	Performance and Coverage .....	116
15.4.11	EPROM .....	118
15.4.12	Macros.....	119
15.4.13	Symbols.....	120
15.4.14	Miscellaneous .....	121

# 1. Introduction

---

## 1.1 About the Products

DT63K  
DTS63K

The DT63K debugger and DTS63K simulator are software development support tools for evaluating and debugging embedded user application programs created with the Oki SASM63K structured assembler for the Oki OLMS-63K series of 4-bit CMOS microcontrollers. The DT63K debugger requires an in-circuit emulator; the DTS63K simulator emulates the target device entirely in software. The user interface and debugging facilities available are otherwise identical. The developer is therefore able to freely switch between the two within the same project in accordance with current needs.

### 1.1.1 Common User Interface

DT63K  
DTS63K

The DT63K debugger and DTS63K simulator share a common user interface providing rapid, interactive access to all the features offered by these advanced debugging environments.

#### ■ Windows 95 or Windows NT operating environment

These tools are 32-bit applications providing a rich, flexible debugging environment.

#### ■ Multiwindow interface

You can freely modify the layout of all windows on the screen by changing their sizes and positions. Multidocument interface (MDI) support also means that you can have multiple windows of the same type open at the same time.

#### ■ Extensive on-screen help

Help consists of the quick on-screen help, simple introductory comments that appear on the status line when a menu item is selected, and the far more comprehensive on-line help.

#### ■ Source level and symbolic debugging

The DT63K debugger and DTS63K simulator provide source level debugging functions that permit debugging while viewing the SASM63K structured assembler source code. You can also display and use the symbols used for labels and



## 1. Introduction

segments in assembly language programs developed with this package.

### ■ Macro functions

Macro support provides automated operation using text files written in a scripting language. These macro functions relieve you of the tedium of procedures that have to be repeated, purely mechanical operation over extended periods, and other forms of batch processing.

## 1.1.2 DT63K Debugger Features

DT63K

The DT63K debugger, in conjunction with an Oki Electric in-circuit emulator, supports evaluation and debugging of microcontroller embedded application software in a hardware environment approaching that of the finished application.

### ■ Real-time emulation

This mode provides full-speed execution that is closest to the final application environment.

### ■ Single-step execution

This mode executes program code one instruction at a time. When you wish to trace program execution in detail, use single-step execution and examine the contents of peripheral registers (SFR), registers, and data memory locations after each instruction.

### ■ Cycle counter

If your system demands real-time response, you frequently wish to know precisely how long certain sequences take to execute. The in-circuit emulator can help by counting execution cycles during emulation execution.

## 1.1.3 DTS63K Simulator Features

DTS63K

The DTS63K simulator offers LCD panel simulation, I/O port simulation, and other simulation capabilities not available with the DT63K debugger.

### ■ LCD Panel Simulation

The software simulates an LCD panel on the development host screen using data file from the LCDTOOL utility.

### ■ Timer and Port Simulation

The software's simulation capabilities extend to the operation of time base and other counters and of I/O ports.

### ■ Melody and Buzzer Simulation

The software's simulation capabilities extend to the operation of melody and buzzer drivers.

### ■ Close Compatibility with DT63K

A user interface largely identical to that of the DT63K debugger allows the developer to freely switch between the two within the same project.

## 1.2 Structure of This Manual

DT63K  
DTS63K

This manual consists of the following chapters.

### ■ Chapter 1 Introduction

This Chapter.

### ■ Chapter 2 Before You Start

This Chapter gives the system requirements for the products, their place in the development process, and the installation procedure.

### ■ Chapter 3 Loading, Preparation, and Projects

This Chapter discusses loading the products, preparing to execute programs, and the structure of project files.

### ■ Chapter 4 Program Execution

This Chapter discusses program execution and closely related functions.

### ■ Chapter 5 Break Functions

This Chapter describes the various break conditions available for breaking program execution.

### ■ Chapter 6 Displaying/Modifying Memory and Registers

This Chapter describes the procedures for viewing the contents of memory locations, the program counter, and other registers and then modifying those contents.

### ■ Chapter 7 Watch Windows

This Chapter describes the procedures for specifying watch items—memory locations, registers, and other objects—for monitoring in a Watch window.

## 1. Introduction

### ■ Chapter 8 Tracing

This Chapter describes the use of the tracing capabilities built into the products.

### ■ Chapter 9 Cycle Counter

This Chapter describes the use of the cycle counter built into the products.

### ■ Chapter 10 Source Level Debugging

This Chapter describes the procedures for source level debugging with Source windows.

### ■ Chapter 11 Symbolic Debugging and Input Formats

The products allow the use of expressions in place of numerical values. Especially important here is the description of the procedures for writing expressions incorporating the symbols used in symbolic debugging.

### ■ Chapter 12 Other Functions

This Chapter covers macros, logging, on-line help, and other miscellaneous functions.

### ■ Chapter 13 DTS63K Simulation Functions

This Chapter describes the simulation of peripheral functions by the DTS63K simulator.

### ■ Chapter 14 LCDTOOL Utility

This Chapter describes the use and functions of the LCDTOOL utility.

### ■ Chapter 15 Appendices

This Chapter provides detailed listings of menus (both main and local), tool bars, shortcut keys, etc. for the DT63K debugger, DTS63K simulator, and LCDTOOL utility.

### ■ Index

## 1.3 Notational Conventions

A key is represented by the legend on the key. If keys must be pressed together, they are joined with a plus sign (+). The notation [Shift]+[F1] in the text, for example, means that you are to hold down a [Shift] key and press the [F1] key.

This Manual uses the following icons in the left margin to indicate the scope of the accompanying material.

<b>DT63K DTS63K</b>	Material applicable to both the DT63K debugger and DTS63K simulator
<b>DT63K</b>	Material applicable to the DT63K debugger only
<b>DTS63K</b>	Material applicable to the DTS63K simulator only
<b>LCDTOOL</b>	Material applicable to the LCDTOOL utility only

This Manual does not use any other icons or other notational conventions that need explaining.

This Manual also uses the numeric formats, units, and notations shown in the Table.

Classification	Notation	Meaning
Numbers	xxh, xxH	Hexadecimal notation, where xx denotes a sequence of hexadecimal digits.
	Word	1 word = 16 bits
	Byte	1 byte = 2 nibbles = 8 bits
	Nibble	1 nibble = 4 bits
	kilo or K	$2^{10}=1024$

## 2. Before You Start

---

### 2.1 System Configuration

DT63K  
DTS63K

The DT63K debugger and DTS63K simulator share the following operating environment requirements.

#### ■ Development Languages Supported

SASM63K structured assembler                      Version 2.21 or later

#### ■ Operating Environment

- Operating System                      Windows 95, Windows NT 4.0 and later
- Hardware                                      Environment sufficient to run the above operating system reliably and efficiently
- Display                                        Graphics adapter and display device with a minimum resolution of 640×480 (VGA)
- Hard disk space                            At least 10 megabytes free
- 3.5-inch floppy disk drive
- Mouse

DT63K

#### ■ DT63K Debugger Environment Requirements

The DT63K debugger has the following additional requirements.

- Serial port
- Compatible in-circuit emulator (A list of supported models appears as a separate document.)

DT63K

#### ■ Checking DT63K Debugger Hardware Environment

Before you can even load the DT63K debugger, you must first connect and configure the development host, in-circuit emulator, and user application circuit. Items to be checked include the following.

- Power supply to the in-circuit emulator
- Clock operating frequencies

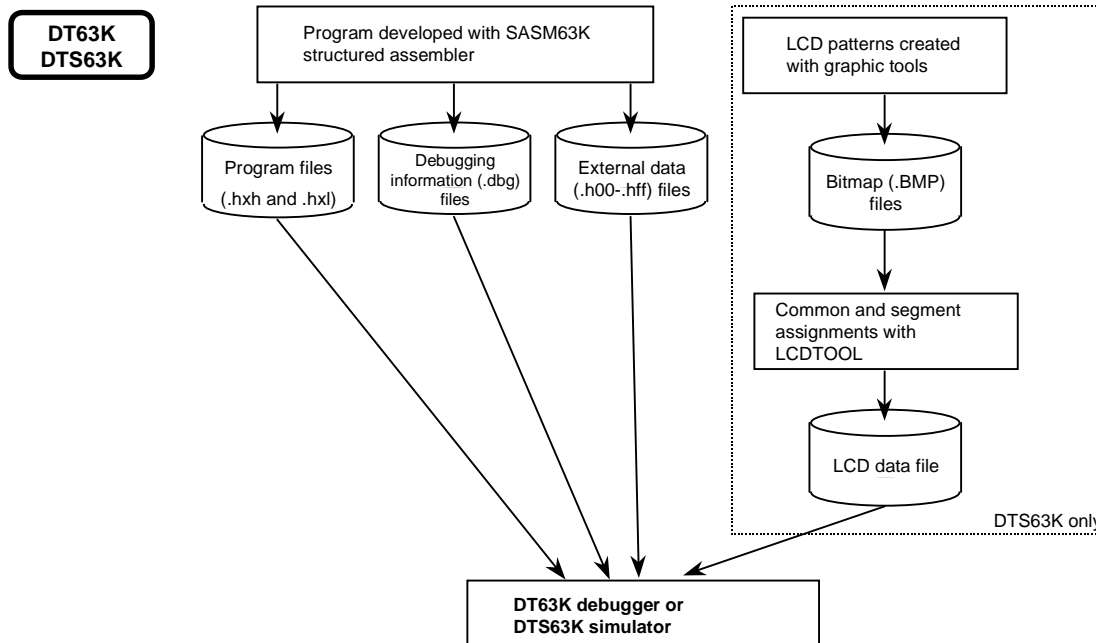
## 2.1 System Configuration

- Target device settings
- User cable joining in-circuit emulator to user application circuit
- Probe cable connections

Note that the above is a minimal list. For further requirements, refer to the manual included with the in-circuit emulator.

## 2. Before You Start

### 2.2 Using Products in Development Process



The products read program code files in two basic formats: the Intel HEX format and the Oki Electric source level debugging (.dbg) format.

The Intel HEX format is a general-purpose ASCII format widely supported by third-party PROM writers. Note, however, that this format has an address space of only 64 kilobytes.

The Oki Electric source level debugging (.dbg) format is Oki Electric's own binary format, which the SASM63K structured assembler uses when producing Oki Electric source level debugging (.dbg) format program code files containing source line information.

DT63K  
DTS63K

#### ■ Intel HEX Half-Word (.hxx and .hxl) Files

Because the OLMS-63K series uses 16-bit instructions, this format uses two Intel HEX format program code files: a .hxx file for the higher 8 bits and a .hxl file for the lower.

Oki has extended the format specifications to allow users to include user symbol information in program code files in this format. Because these extensions are in compliance with Intel HEX format specifications, third-party utilities and PROM writers should simply ignore the symbol information in such files.

## 2.2 Using Products in Development Process

DT63K  
DTS63K

### ■ External Data (.h00–.hff) Files

The SASM63K structured assembler uses the Intel HEX format for addresses in the external data memory space, but uses numbered extensions to give the bank number in hexadecimal. (Note that each bank contains 64 kilobytes.)

DT63K  
DTS63K

### ■ Oki Electric Source Level Debugging (.dbg) Files

The SASM63K structured assembler uses the extension .dbg for files containing debugging information.

Before you can view a source file or debug in a DT63K/DTS63K Source window, you must first load this type of file.

DTS63K

### ■ LCD Data (.LCD) Files

DTS63K simulator capabilities include LCD panel simulation using common and segment specifications from these files. To prepare them, create the desired displays as Windows bit-mapped files (.BMP)—with MS Paint, for example—and convert them with the LCDTOOL utility.



## 2. Before You Start

### 2.3 Installing DT63K and DTS63K

The DT63K debugger and DTS63K simulator use standard Windows installation procedures.

DT63K  
DTS63K

#### ■ Installing the Software

1. Insert setup disk #1 in the floppy disk drive.
2. Run the setup utility (SETUP.EXE).
3. Follow the instructions that appear on the screen.

#### ■ Uninstalling the Software

- Open the Control Panels folder.
- Double-click the appropriate icon to uninstall the software.

#### ■ Backing Up the Software

We strongly recommend that you make backup copies of all installation disks and store the originals in a safe place.

## 3. Loading, Preparation, and Projects

---

### 3.1 Loading the Software

DT63K  
DTS63K

#### ■ Start Menu

The setup utility automatically creates a program group under Programs on the Start menu.

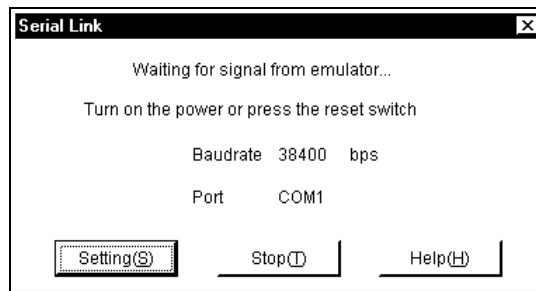
Double-click the corresponding icon to load either the DT63K debugger or the DTS63K simulator.

Normally, the software automatically loads the last project used in the preceding session. The first time, however, there will be no such project.

DT63K

#### ■ DT63K Debugger

The DT63K debugger starts with the following dialog box and waits for a reset code from the in-circuit emulator.

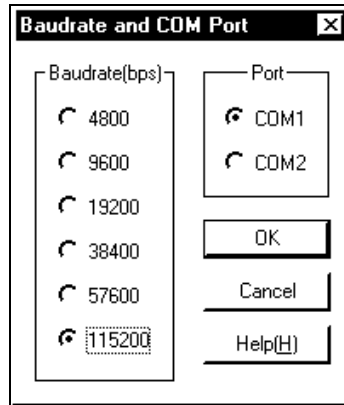


Clicking the left mouse button on the Cancel button immediately closes the software.

Clicking the left mouse button on the Settings button displays the following dialog box specifying the serial link speed (bps) and the port.

Note that the speeds available depend on the hardware and other factors. Members of the NEC PC-9801 and PC-9821 series, for example, only support speeds up to 38,400 bps.

### 3. Loading, Preparation, and Projects

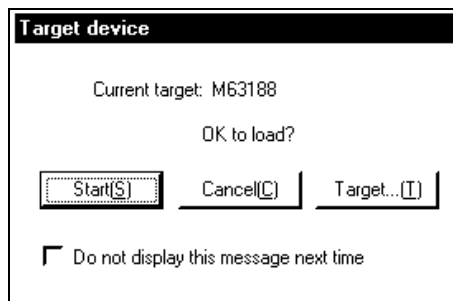


Clicking the left mouse button on the Cancel button immediately closes the dialog box, canceling any changes.

**DTS63K**

#### ■ DTS63K Simulator

The DTS63K simulator starts with the following dialog box.



Clicking the left mouse button on the Start button loads the software configured for the specified target device.

Clicking the left mouse button on the Cancel button immediately closes the software.

Clicking the left mouse button on the Target... button displays a dialog box for specifying a new target device.

## 3.2 Preparing for Debugging

### 3.2.1 Environment Settings

Open the Tools menu and choose the Environment settings menu command.



#### ■ Project

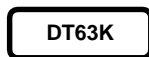
Click this tab to display the project configuration panel.

#### ■ Directories

Click this tab to display the default directories for the various file types.

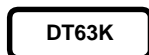
## 3.3 Preparing for Emulation

### 3.3.1 Specifying Target Device



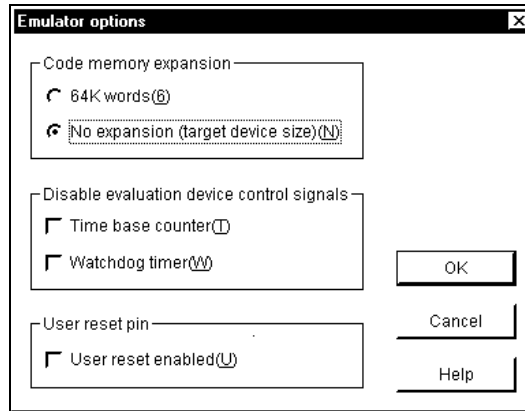
The DT63K debugger determines which device-specific field to read from the target device setting in the in-circuit emulator. To view this setting, open the View menu and choose the Status menu command to open a Status window. If this setting does not match the target device, switch the in-circuit emulator to SETICE mode using the procedure in the in-circuit emulator manual, reload the DT63K debugger, and change the setting with the File menu's Load TCD file menu command.

### 3.3.2 Specifying Emulator Options



The DT63K Tools menu's Options menu command displays a dialog box controlling in-circuit emulator options.

### 3. Loading, Preparation, and Projects



#### ■ Code Memory Expansion

These radio buttons control the use of the in-circuit emulator's function for expanding code memory to 64K words. The default is to limit code memory size to that of the program memory for the target device.

#### ■ Timer/Counter Controls

These check boxes control the use of time base counter (TBC) and watchdog timer (WDT) signals during emulation and single-step execution. The default is to enable these signals. Selecting a check box disables the corresponding signal.

#### ■ User Reset Pin

Selecting this check box disables the user cable's RESET pin input during real-time emulation.

## 3.4 Preparing for Simulation

### 3.4.1 Loading LCD Data File

**DTS63K**

LCD panel simulation with the DTS63K simulator requires an LCD data file created from Windows bit-mapped files with the LCDTOOL utility. (See Chapter 14 “LCDTOOL Utility.”)

Load this file with the File menu’s Load LCD data file menu command.

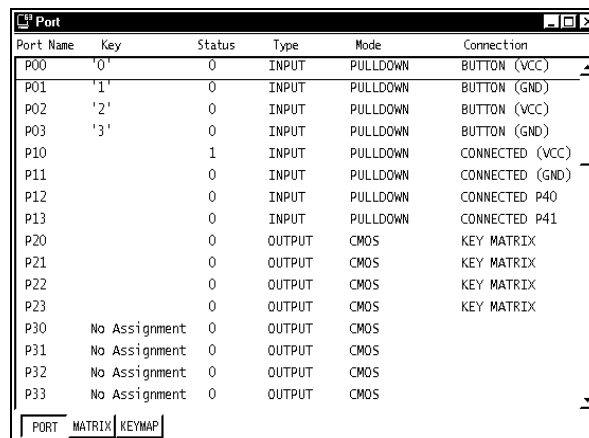
### 3.4.2 Assigning Ports

**DTS63K**

I/O port simulation with the DTS63K simulator requires assigning port inputs to keys on the development host’s keyboard.

#### ■ Displaying Port Assignments

Clicking the left mouse button on the Port button at the bottom of the Port window displays the current port-key assignments by port number.



Port Name	Key	Status	Type	Mode	Connection
P00	'0'	0	INPUT	PULLDOWN	BUTTON (VCC)
P01	'1'	0	INPUT	PULLDOWN	BUTTON (GND)
P02	'2'	0	INPUT	PULLDOWN	BUTTON (VCC)
P03	'3'	0	INPUT	PULLDOWN	BUTTON (GND)
P10		1	INPUT	PULLDOWN	CONNECTED (VCC)
P11		0	INPUT	PULLDOWN	CONNECTED (GND)
P12		0	INPUT	PULLDOWN	CONNECTED P40
P13		0	INPUT	PULLDOWN	CONNECTED P41
P20		0	OUTPUT	CMOS	KEY MATRIX
P21		0	OUTPUT	CMOS	KEY MATRIX
P22		0	OUTPUT	CMOS	KEY MATRIX
P23		0	OUTPUT	CMOS	KEY MATRIX
P30	No Assignment	0	OUTPUT	CMOS	
P31	No Assignment	0	OUTPUT	CMOS	
P32	No Assignment	0	OUTPUT	CMOS	
P33	No Assignment	0	OUTPUT	CMOS	

**Port Name:** Port name

**Key:** Key assigned to port as push button or on/off switch

**Status:** Current port status (0: “Low” level; 1: “High” level)

**Type:** Port type (INPUT or OUTPUT)

**Mode:** Port configuration

**Pull-up:** Input with pull-up resistor

**Pull-down:** Input with pull-down resistor

**Pull-up/down:** Input with pull-up/down resistor

### 3. Loading, Preparation, and Projects

**Highinp:** High-impedance input or output

**CMOS:** CMOS output

**NchOD:** n-channel open drain output

**PchOD:** p-channel open drain output

**Connection:** Port connection

**No assignment:** Unassigned

**Button (VDDI):** “High” level push button

**Button (GND):** “Low” level push button

**Switch (VDDI):** “High” level on/off switch

**Switch (GND):** “Low” level on/off switch

**Connected (VDDI):** Always “High” level

**Connected (GND):** Always “Low” level

**Connected Pxx:** Connected to specified output port

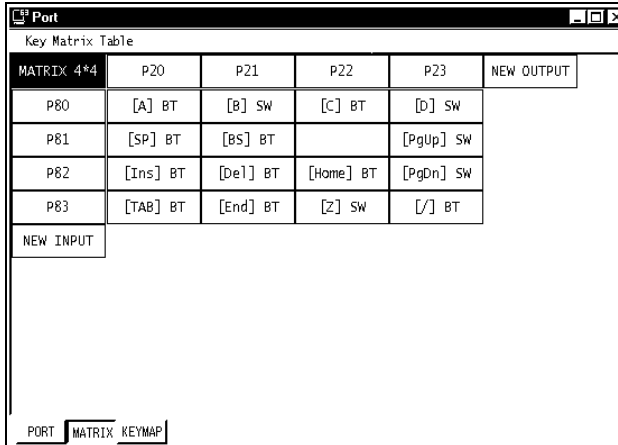
**Pull up:** Pull-up connection

**Pull down:** Pull-down connection

**Key matrix:** Key matrix connection

#### ■ Displaying Key Matrix

Clicking the left mouse button on the Matrix button at the bottom of the Port window displays the current key matrix table.



The screenshot shows a window titled "Port" with a "Key Matrix Table" displayed. The table has columns for ports P20, P21, P22, P23, and a "NEW OUTPUT" column. The rows are labeled P80, P81, P82, and P83. A "NEW INPUT" field is located below the table. At the bottom of the window, there are buttons for "PORT", "MATRIX", and "KEYMAP".

MATRIX 4*4	P20	P21	P22	P23	NEW OUTPUT
P80	[A] BT	[B] SW	[C] BT	[D] SW	
P81	[SP] BT	[BS] BT		[PgUp] SW	
P82	[Ins] BT	[Del] BT	[Home] BT	[PgDn] SW	
P83	[TAB] BT	[End] BT	[Z] SW	[/] BT	

#### ■ Displaying Key Assignments

Clicking the left mouse button on the Keymap button at the bottom of the Port window displays the current port-key assignments by key name.

Key Name	Input Port	Output Port	Connection
BS	P81	P21	KEY MATRIX
Tab	P83	P20	KEY MATRIX
Space	P81	P20	KEY MATRIX
PgUp	P81	P23	KEY MATRIX
PgDn	P82	P23	KEY MATRIX
End	P83	P21	KEY MATRIX
Home	P82	P22	KEY MATRIX
Ins	P82	P20	KEY MATRIX
Del	P82	P21	KEY MATRIX
'0'	P00		BUTTON (VCC)
'1'	P01		BUTTON (GND)
'2'	P02		BUTTON (VCC)
'3'	P03		BUTTON (GND)
'4'			
'5'			
'6'			

PORT MATRIX KEYMAP

#### ■ Simple Key-Port Assignments

There are three ways to display the dialog box for assigning port inputs to keys on the development host's keyboard.

- Click the left mouse button on the Port button, right-click, and choose the Change key assignments... menu command from the context menu.
- Click the left mouse button on the Keymap button, right-click, and choose the Change key assignments... menu command from the context menu.
- Choose the Tools menu's Change key assignments... menu command.

All three methods lead to the same dialog box.

**Change key assignments**

Input port	Output port	Key
P00	P21	BS
P01	P22	Tab
P02	P23	Space
P03	P30	PgUp
P10	P31	PgDn
P11	P32	End
P12	P33	Home
P13	P40	Ins
P80	P41	Del
P81		n

Connection:

- None
- Push button High input
- Push button Low input
- ON/OFF switch High input
- ON/OFF switch Low input
- Always High
- Always Low
- Connect to other port

Buttons: OK, Cancel, Help, Apply(A)

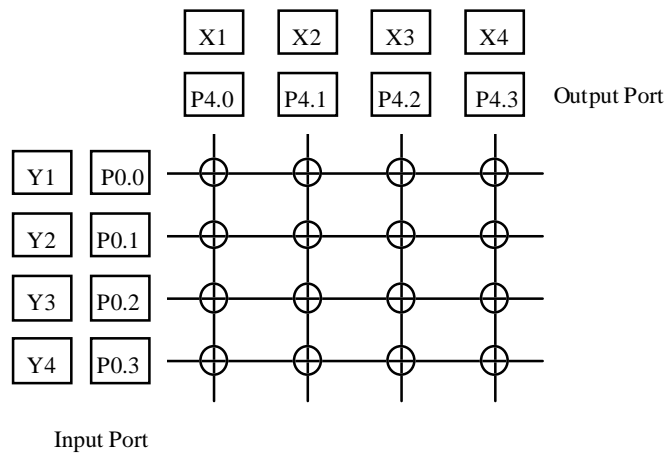


### 3. Loading, Preparation, and Projects

- Input Port:** Port to assign to key
- Output Port:** Connection to output port (Only valid for Connected to other port radio button)
- Key:** Key to assign
- Connection:** Input port connection
- Apply button:** Saves assignment without closing dialog box

#### ■ Setting Up a Key Matrix

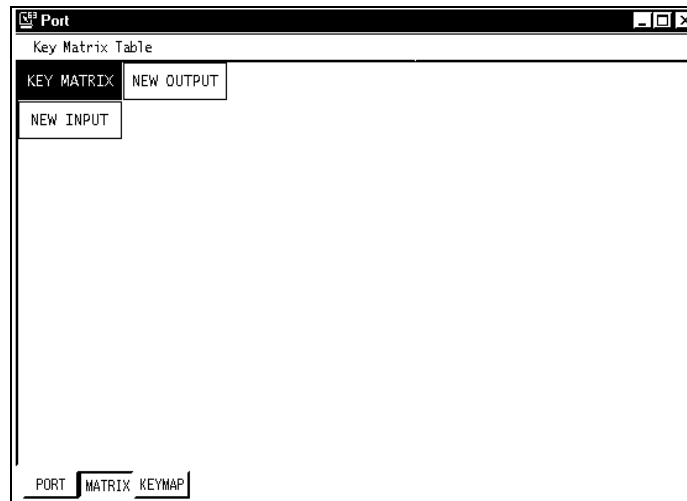
The above one-to-one approach is insufficient for a user application program requiring numerous input keys. A key matrix then becomes more efficient.



Key matrix designs generally expand the number of input gates available by using a rectangular grid connecting input ports along the y-axis to output ports along the x-axis. The circles in the figure indicate the positions of push buttons or on/off switches.

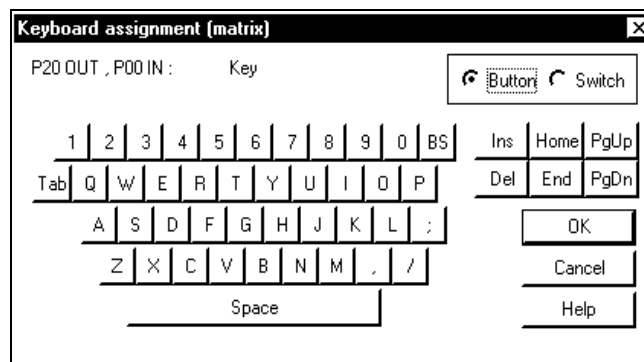
To simulate such a key matrix with the DTS63K simulator, press the Matrix button at the bottom of the Port window. If there is no key matrix defined, the window displays the following empty key matrix.

## 3.4 Preparing for Simulation



To add a port to the matrix, right-click on the appropriate New button and select the menu's Add menu command to add a row or column to the grid.

When the grid is complete, assign keys by selecting an intersection and hitting the [Enter] key to display the following dialog box.



**Button:** Simulate push button

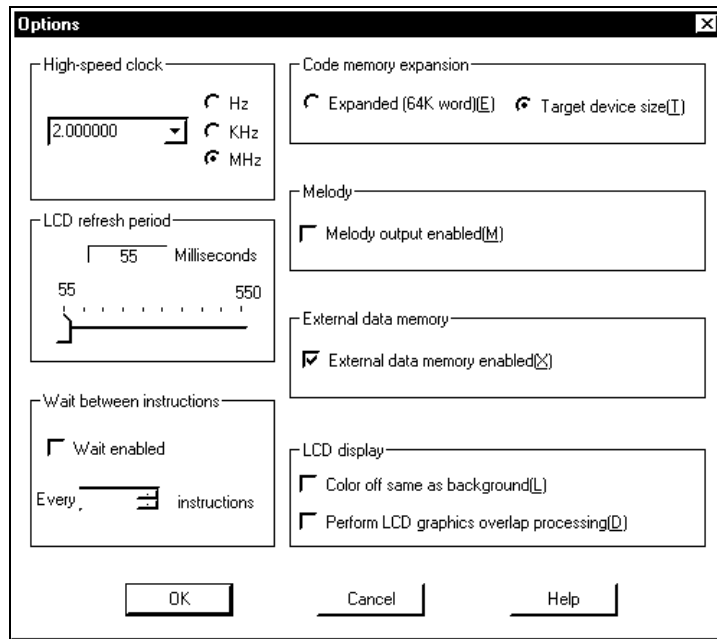
**Switch:** Simulate on/off switch

### 3.4.3 Specifying Simulator Options

**DTS63K**

The DTS63K Tools menu's Options menu command displays a dialog box controlling simulator options.

### 3. Loading, Preparation, and Projects



#### ■ High-Speed Clock

If the user application program uses the high-speed clock, specify the frequency here.

#### ■ LCD Refresh Period Setting

Sets the refresh period for the LCD window during simulation. The LCD window refresh period can be set to a value in the range 55 to 550 ms.

#### ■ Wait

##### • “Wait setting” check box

Check this check box to have a wait inserted every time the specified number of instructions are executed. This function allows the program execution time to be made close to that of the actual microprocessor execution time.

#### ■ Code Memory Expansion

These radio buttons control the use of the function for expanding code memory to 64K words. The default is to limit code memory size to that of the program memory for the target device.

#### ■ Melody Output

Selecting this check box enables melody output.

#### ■ External Data Memory

Selecting this check box enables the use of external data memory.

### ■ LCD Settings

- **Set pixel off color to the background color check box**

Sets whether or not LCD pixels that are turned off are displayed in the same color as the background color.

When this check box is checked, the simulated image will match the image actually displayed.

- **Perform LCD graphics overlap processing check box**

Sets whether or not bitmap graphics are displayed faithfully during simulation. Since LCD graphics display becomes somewhat slower when this check box is checked, clear this check box if higher speed LCD graphics is desired.

### 3. Loading, Preparation, and Projects

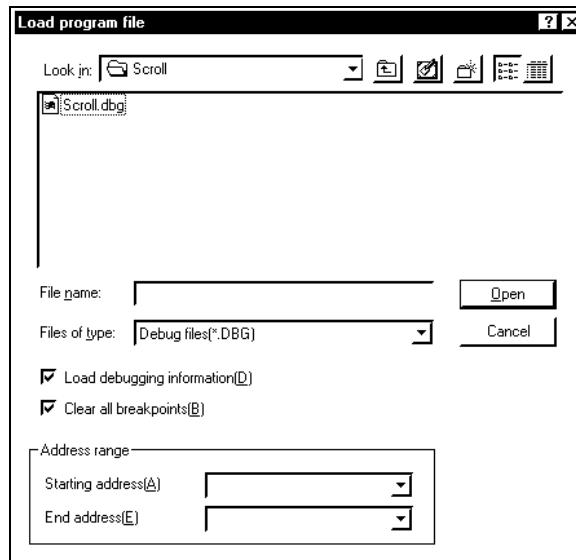
## 3.5 Loading/Saving Program Code

### 3.5.1 Loading Program Code

DT63K  
DTS63K

The DT63K debugger and DTS63K simulator accept program code created with the SASM63K structured assembler.

The File menu's Load program file... menu command displays a dialog box for specifying the program file to debug or simulate.



#### ■ Loading Debugging Information Too

Selecting “Debug file (\*.dbg)” as the file type loads the specified debugging information file and the corresponding program code (.hxx and .hxl) files.

Note that loading this file automatically clears all current debugging information (source line information and symbol table).

#### ■ Loading Only Program Code

Selecting “Program code (\*.hxx)” as the file type and double-clicking a file from the list or selecting “All files (\*.\*)” and double-clicking a file with the extension .hxx or .hxl loads both members of the program code file pair.

Note that loading these files does not clear the current debugging information.

#### ■ Clearing Breakpoints

Because loading a new program normally renders current breakpoints useless, it is

usually a good idea to select the Clear all breakpoints check box.

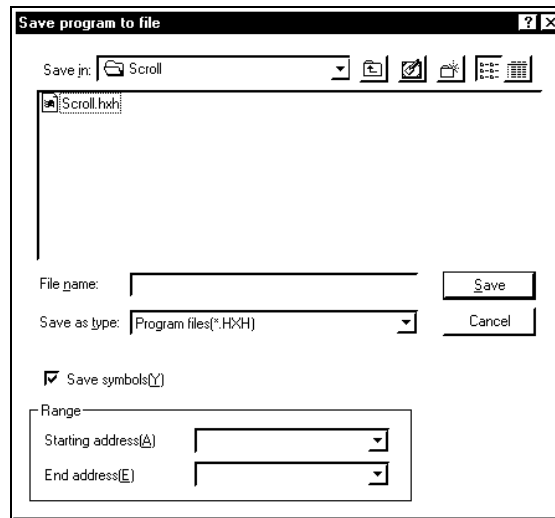
### ■ Limiting the Range

One situation where you might not want to clear breakpoints is when reloading only a ROM data table or a subroutine using the address range specification list boxes at the bottom of the dialog box.

## 3.5.2 Saving Program Code

DT63K  
DTS63K

The File menu's Save program file... menu command displays a dialog box for specifying the destination for saving a program file with adjusted parameters in the ROM table or modified program.



### ■ Saving Program Code Only

Specifying a file with the extension .hxh or .hxl saves the program code to the corresponding program code file pair.

### ■ Saving Symbol Table

To save the symbol table with the program code, select the Save symbols check box.

### ■ Limiting the Range

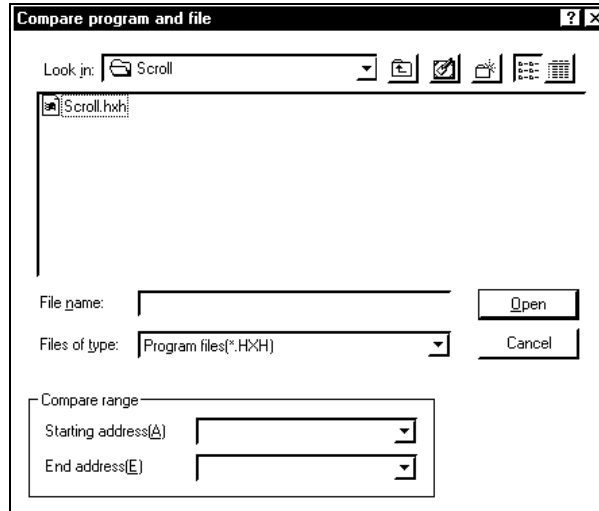
To save only the program code within a specific address range—a ROM table or subroutine variant, for example—use the range list boxes at the bottom of the dialog box.

### 3. Loading, Preparation, and Projects

#### 3.5.3 Comparing Code Memory to File

DT63K  
DTS63K

The File menu's Compare program and file... menu command displays a dialog box for specifying the program file for the comparison.



##### ■ Comparing Complete Program

Specifying a file with the extension .hxx or .hxl compares the program code to the contents of the corresponding program code file pair.

##### ■ Limiting the Range

To compare only the program code within a specific address range, use the range list boxes at the bottom of the dialog box.

## 3.6 Project Files (.d63 and .s63)

DT63K  
DTS63K

The DT63K debugger and DTS63K simulator provide numerous options for debugging and simulating user application programs. To simplify the management of these options, they therefore provide project (.d63 and .s63) files for holding these settings from one session to the next. This approach also allows the developer to switch sets of option settings to match the current needs of the work at hand.

DT63K

### ■ DT63K Debugger Project Files

DT63K debugger project files, with the extension .d63, hold the following option settings.

- Target device
- Most recently loaded source line information file
- In-circuit emulator settings from the Tools menu's Options menu command
- Desktop

DTS63K

### ■ DTS63K Simulator Project Files

DTS63K simulator project files, with the extension .s63, hold the following option settings.

- Target device
- Most recently loaded source line information file
- Settings from the Tools menu's Options menu command
- Most recently loaded LCD data file
- Most recently loaded port data file
- Desktop



# 4. Program Execution

---

## 4.1 Program Execution

DT63K  
DTS63K

The DT63K debugger, which requires connection to an in-circuit emulator, executes the user application program in a hardware environment approaching that of the finished application.

The DTS63K simulator, on the other hand, simulates instructions one at a time. Because this simulation is at the instruction and not the instruction state level, it makes no guarantees as to the accuracy of timing for such things as peripheral I/O.

### ■ Continuous Execution

The Run menu's Run program menu command starts continuous execution. This execution continues until a break condition is satisfied or you force a break.

DTS63K

The DTS63K simulator's LCD window simulates an LCD panel by tracking the contents of the display registers (DSPR).

Instruction execution speed with the DTS63K simulator depends on development host performance. The Tools menu's Options menu command therefore provides a wait setting for slowing down the simulation.

DT63K  
DTS63K

### ■ Single-Step Execution

The Run menu's Step into menu command executes program code one instruction at a time. You are therefore able to trace program execution in detail, examining the contents of peripheral registers (SFR), registers, and data memory locations after each instruction.

If the instruction is a subroutine call, execution stops at the entrance to that subroutine.

If an interrupt intervenes, the instruction executed is the first one in the interrupt service routine.

DTS63K

### ■ Step Over Execution

This variant of single-step execution is only available with the DTS63K simulator. The Run menu's Step over menu command executes program code one instruction

## 4.1 Program Execution

at a time unless that instruction is a subroutine call. In that case, it treats the entire subroutine—everything from the call instruction through to the return instruction—as a single instruction.

If an interrupt intervenes, the instruction executed is the first one after the interrupt service routine returns.

Because this variant skips over subroutines and interrupt service routines, it is most useful for testing mainline execution.

DT63K  
DTS63K

### ■ Suspending Program Execution

Temporarily suspending program execution is called a break. The software provides a wide variety of automatic breaks based on conditions that you specify. For further details, see Chapter 5 “Break Functions.”

DTS63K

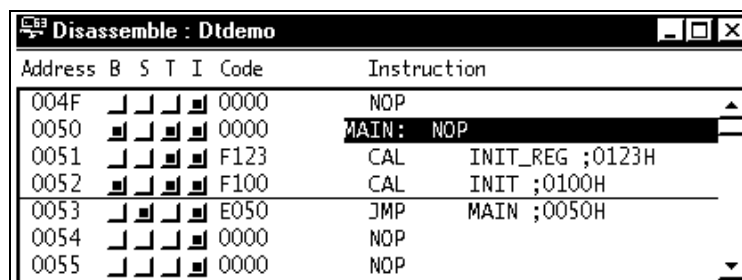
### ■ Stop melody playback

When the DTS63K simulator melody function is used, melody playback continues even after program execution is interrupted by a break. Select Stop melody play from the Execution menu to forcibly stop melody playback.

DT63K  
DTS63K

### ■ IE Bits

Each code memory address has an Instruction Executed (IE) bit indicating whether the instruction at that address has been executed. The current state appears in the I column in Disassembly windows.



Address	B	S	T	I	Code	Instruction
004F	↓	↓	↓	↓	0000	NOP
0050	↓	↓	↓	↓	0000	MAIN: NOP
0051	↓	↓	↓	↓	F123	CAL INIT_REG ;0123H
0052	↓	↓	↓	↓	F100	CAL INIT ;0100H
0053	↓	↓	↓	↓	E050	JMP MAIN ;0050H
0054	↓	↓	↓	↓	0000	NOP
0055	↓	↓	↓	↓	0000	NOP

IE bit. A mark indicates that the instruction has been executed.

## 4. Program Execution

### 4.2 Resets

DT63K

#### ■ CPU Reset

Offered only by the DT63K debugger, the Run menu's Reset CPU menu command resets the target device in the in-circuit emulator.

DT63K

#### ■ System Reset

Offered only by the DT63K debugger, the Run menu's Reset system menu command resets the in-circuit emulator hardware and software.

DTS63K

#### ■ Simulator Reset

Offered only by the DTS63K simulator, the Run menu's Reset simulator menu command resets the software. For further details, see Section 13.2 "Resets."

DT63K  
DTS63K

#### ■ Specifying Reset Scope

The Run menu's Choose reset object... menu command displays a dialog box with options for clearing the cycle counter, IE bits, and trace memory as part of system (DT63K) or simulator (DTS63K) reset.

# 5. Break Functions

## 5.1 Breakpoints

DT63K  
DTS63K

A breakpoint is a user-specified address at which program execution is temporarily suspended whenever the user application program attempts to execute the instruction at that address. You can have multiple address specifications in effect at the same time.

### 5.1.1 Setting Breakpoints

DT63K  
DTS63K

#### ■ On the Source Window

The current breakpoint state for each instruction appears in the BP column in Source windows and in the B column in Disassembly windows. A dark square indicates a breakpoint. Clicking a box toggles this setting on and off.

Line	BP	Address		
00079	<input checked="" type="checkbox"/>	0101H	MOV	L, #(mr_adr&0FH) ;
00080	<input type="checkbox"/>			
00081	<input type="checkbox"/>	MLBCD20:		
00082	<input type="checkbox"/>	0102H	MOV	[HL+], #0H ;
00083	<input type="checkbox"/>	0103H	CMP	L, A ; ;
00084	<input type="checkbox"/>	0104H	BNE	MLBCD20 ;
00085	<input type="checkbox"/>			
00086	<input checked="" type="checkbox"/>	0105H	MOV	H, #(ms_adr&0F0H)>>4; HL <- source a
00087	<input type="checkbox"/>	0106H	MOV	L, #((ms_adr+0)&0FH);
00088	<input type="checkbox"/>	0107H	MOV	Y, #7H ;
00089	<input checked="" type="checkbox"/>	0108H	JMP	MLBCD40 ;
00090	<input type="checkbox"/>			
00091	<input type="checkbox"/>	MLBCD30:		
00092	<input checked="" type="checkbox"/>	0109H	PUSH	HL ;
00093	<input type="checkbox"/>	010AH	MOV	H, #(mr_adr&0F0H)>>4; [HL+0 - 7] + n
00094	<input type="checkbox"/>	010BH	MUL_MACRO()	
00095	<input type="checkbox"/>	011BH	ADCD	[HL+], 10 ; [HL+8] + Carry

#### ■ On the Disassembly Window

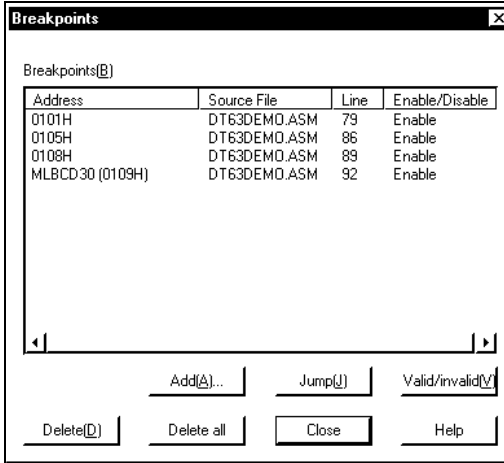
Address	B	I	Code	Instruction
0101	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0120	MOV L, #0H
0102	<input type="checkbox"/>	<input type="checkbox"/>	0760	MLBCD20: MOV [HL+], #0H
0103	<input type="checkbox"/>	<input type="checkbox"/>	2AFB	CMP L, A ; 0FBH
0104	<input type="checkbox"/>	<input type="checkbox"/>	0DFD	BNZ MLBCD20 ; 0102H
0105	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0131	MOV H, #1H
0106	<input type="checkbox"/>	<input type="checkbox"/>	0120	MOV L, #0H
0107	<input type="checkbox"/>	<input type="checkbox"/>	0107	MOV Y, #7H
0108	<input checked="" type="checkbox"/>	<input type="checkbox"/>	E11D	JMP MLBCD40 ; 011DH
0109	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0010	MLBCD30: PUSH HL
010A	<input type="checkbox"/>	<input type="checkbox"/>	0132	MOV H, #2H
010B	<input type="checkbox"/>	<input type="checkbox"/>	D200	MOV A, DM200 ; 200H
010C	<input type="checkbox"/>	<input type="checkbox"/>	052C	ADCD [HL+], A

## 5. Break Functions

### 5.1.2 Listing and Suspending Breakpoints

DT63K  
DTS63K

The Run menu's List breakpoints... menu command lists all current breakpoint settings.



- Toggle button:** Clicking the left mouse button on this button temporarily disables the selected breakpoints without deleting them from the list. Clicking it a second time re-enables them.
- Add... button:** Clicking the left mouse button on this button displays a dialog box for adding a new breakpoint to the list.
- Jump button:** Clicking the left mouse button on this button displays the Source window containing the specified address.

## 5.2 Executing to Cursor

DT63K  
DTS63K

The Run menu's Execute to cursor menu command produces execution from the current program counter position through to the line containing the cursor in the active Source or Disassembly window. These windows indicate this line with an underline.

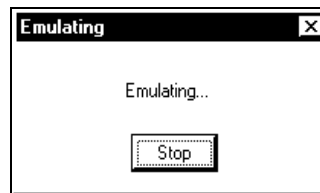
## 5.3 Forced Break

DT63K  
DTS63K

Execution normally continues until the program hits a breakpoint or satisfies a break condition, but you can force a break at any time.

DT63K

In the DT63K debugger, click the left mouse button on the Stop button in the dialog box on the screen.



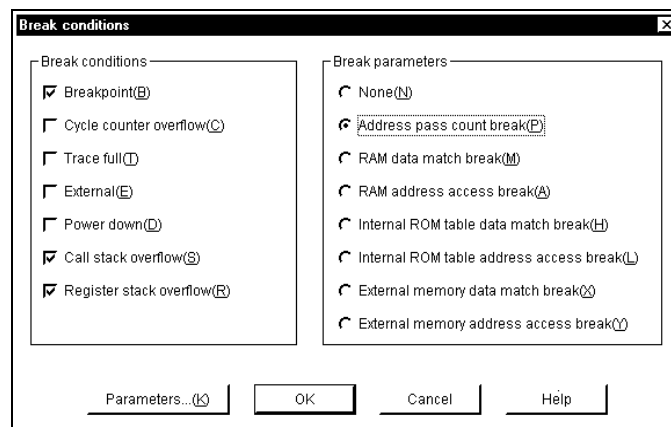
DTS63K

In the DTS63K simulator, choose the Run menu's Stop program menu command.

## 5.4 DT63K Break Conditions

DT63K

The Run menu's Break conditions... menu command displays the dialog box controlling break conditions.



## 5. Break Functions

The check boxes on the left represent break conditions provided by the in-circuit emulator. These are totally independent, so you can specify as many or as few as you like. They are also independent of the setting in the right half.

The radio buttons on the right represent mutually exclusive choices applicable only during execution with the Run menu's Run program menu command, not with the Execute to cursor menu command.

Clicking the left mouse button on the Parameters button displays the dialog box for specifying the parameters for the selected radio button.

### 5.4.1 Break Condition Check Boxes

DT63K

#### ■ Breakpoint

Selecting this check box produces a break when the program counter hits an enabled breakpoint.

#### ■ Cycle counter overflow

Selecting this check box produces a break when the cycle counter increments past 16777215.

#### ■ Trace Full

Selecting this check box produces a break when the trace memory overflows.

#### ■ External

Selecting this check box produces a break when the in-circuit emulator receives input from the probe cable's external break pin.

#### ■ Power Down

Selecting this check box produces a break when the program switches to HALT mode with the HALT instruction. Note that the in-circuit emulator automatically forces the target device out of the HALT mode.

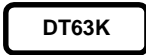
#### ■ Call Stack Overflow

Selecting this check box produces a break when the program counter stack under- or overflows.

#### ■ Register Stack Overflow

Selecting this check box produces a break when the register stack under- or overflows.

## 5.4.2 Break Parameters



### ■ None

Selecting this radio button produces no breaks in this category.

### ■ Address Pass Count Break

Selecting this radio button produces a break when the program counter has visited the specified address the specified number of times. Clicking the left mouse button on the Parameters button displays the dialog box for specifying the address and the count.

### ■ RAM Data Match Break

Selecting this radio button produces a break when an instruction in the specified code memory address range writes the specified masked data to the specified data memory address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.

**Address:** Data memory address to monitor

**Data:** Write data to monitor

**Mask:** Mask to apply to write data

**Count:** Matches to count

**Break address range:** Code memory addresses to monitor. Leaving both these

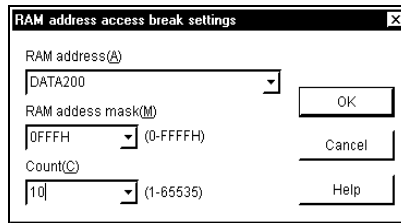


## 5. Break Functions

fields blank extends coverage to the entire code memory. Leaving the second one blank monitors the single address in the first one.

### ■ RAM Address Access Break

Selecting this radio button produces a break when the user application program accesses (write access only) the specified masked data memory address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.



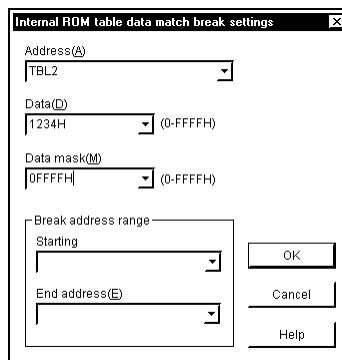
**Address:** Data memory address to monitor

**Mask:** Mask to apply to address

**Count:** Matches to count

### ■ Internal ROM Table Data Match Break

Selecting this radio button produces a break when an instruction in the specified code memory address range reads the specified masked data from the specified internal ROM table address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.



**Address:** Internal ROM table address to monitor

**Data:** Read data to monitor

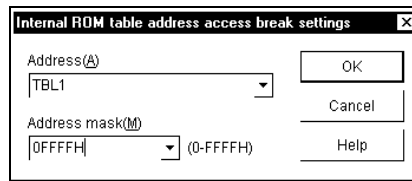
**Mask:** Mask to apply to read data

**Count:** Matches to count

**Break address range:** Code memory addresses to monitor. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank monitors the single address in the first one.

■ **Internal ROM Table Address Access Break**

Selecting this radio button produces a break when the user application program accesses (read only) the specified masked internal ROM table address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.

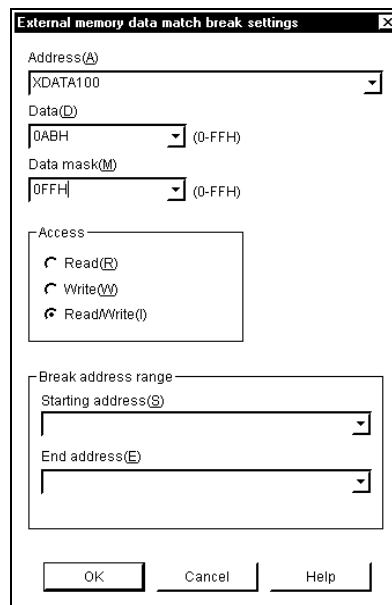


**Address:** Internal ROM table address to monitor

**Mask:** Mask to apply to address

■ **External Memory Data Match Break**

Selecting this radio button produces a break when an instruction in the specified code memory address range reads/writes the specified masked data from/to the specified external memory address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.



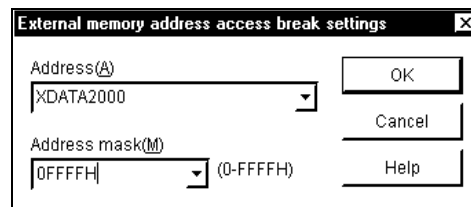
**Address:** External memory address to monitor

## 5. Break Functions

- Data:** Read/write data to monitor
- Mask:** Mask to apply to read/write data
- Access:** Type of access
- Break address range:** Code memory addresses to monitor. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank monitors the single address in the first one.

### ■ External Memory Address Access Break

Selecting this radio button produces a break when the user application program accesses (for either a read or a write) the specified masked external memory address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.



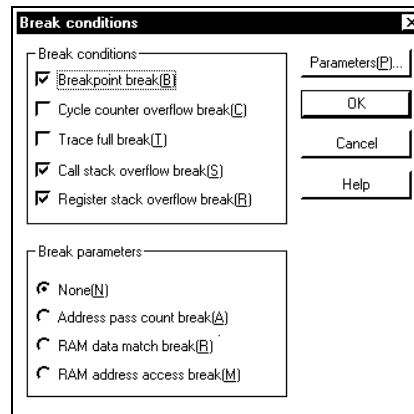
**Address:** External memory address to monitor

**Mask:** Mask to apply to address

## 5.5 DTS63K Break Conditions

DTS63K

The Run menu's Break conditions... menu command displays the dialog box controlling break conditions.



Clicking the left mouse button on the Parameters button displays the dialog box for specifying the parameters for the selected radio button.

### 5.5.1 Break Condition Check Boxes

DTS63K

#### ■ Breakpoint

Selecting this check box produces a break when the program counter hits an enabled breakpoint.

#### ■ Cycle counter overflow

Selecting this check box produces a break when the cycle counter increments past 2147483647.

#### ■ Trace Full

Selecting this check box produces a break when the trace memory overflows.

#### ■ Call Stack Overflow

Selecting this check box produces a break when the program counter stack under- or overflows.

#### ■ Register Stack Overflow

Selecting this check box produces a break when the register stack under- or overflows.

## 5. Break Functions

### 5.5.2 Break Parameters

DTS63K

#### ■ None

Selecting this radio button produces no breaks in this category.

#### ■ Address Pass Count Break

Selecting this radio button produces a break when the program counter has visited the specified address the specified number of times. Clicking the left mouse button on the Parameters button displays the dialog box for specifying the address and the count.

#### ■ RAM Data Match Break

Selecting this radio button produces a break when an instruction in the specified code memory address range writes the specified masked data to the specified data memory address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.

RAM data match break settings

RAM address(A)  
DM200

RAM data(D)  
4H (0-F)

RAM data mask(M)  
0FH (0-F)

Count(C)  
6

RAM access  
 Read(R)  
 Write(W)  
 Read/write(I)

Break address range  
Starting address(S)  
End address(E)

OK Cancel Help

**Address:** Data memory address to monitor

**Data:** Write data to monitor

**Mask:** Mask to apply to write data

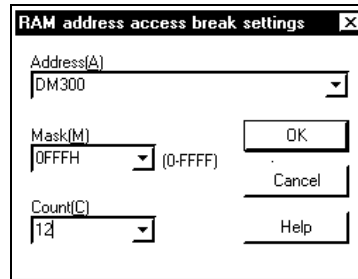
**Count:** Matches to count

**Access:** Type of access

**Break address range:** Code memory addresses to monitor. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank monitors the single address in the first one.

### ■ RAM Address Access Break

Selecting this radio button produces a break when the user application program accesses (for either a read or a write) the specified masked data memory address the specified number of times. Clicking the left mouse button on the Parameters button displays the following dialog box.



**Address:** Data memory address to monitor

**Mask:** Mask to apply to address

**Count:** Matches to count

## 6. Displaying/Modifying Memory and Registers

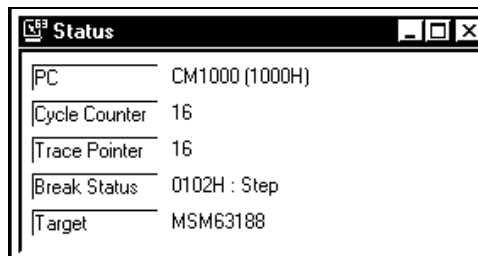
---

### 6.1 Program Counter

DT63K  
DTS63K

The current contents of the program counter appear in the Status window.

To change them, right-click on the window and choose the Change program counter... menu command.



It also possible to monitor the contents by adding the program counter to the watch list in a Watch window.

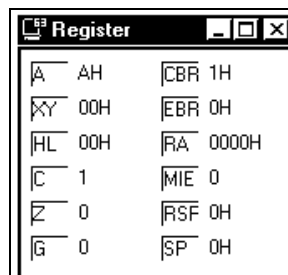
Source and Disassembly windows highlight the line containing the program counter.

### 6.2 Registers

DT63K  
DTS63K

The registers appear in a Register window.

To change the contents of a register, right-click on the window and choose the appropriate Change xxx... menu command.



### ■ Accumulator (A)

This field shows the contents of the accumulator. Modify them with the Change A... menu command.

### ■ XY Register

This field shows the contents of the XY register. Modify them with the Change XY... menu command. The dialog box splits the register into X and Y registers.

### ■ HL Register

This field shows the contents of the HL register. Modify them with the Change HL... menu command. The dialog box splits the register into H and L registers.

### ■ Carry (C) Flag

This field shows the contents of the carry flag. Modify them with the Change C flag... menu command.

### ■ Zero (Z) Flag

This field shows the contents of the zero flag. Modify them with the Change Z flag... menu command.

### ■ G Flag

This field shows the contents of the G flag. Modify them with the Change G flag... menu command.

### ■ Current Bank Register (CBR)

This field shows the contents of the current bank register. Modify them with the Change CBR... menu command.

### ■ Extra Bank Register (EBR)

This field shows the contents of the extra bank register. Modify them with the Change EBR... menu command.

### ■ RA Registers (RAx)

This field shows the contents of the RA registers. Modify them with the Change RAx... menu command.

### ■ Master Interrupt Enable (MIE) Flag

This field shows the contents of the master interrupt enable flag. Modify them with the Change MIE flag... menu command.



## 6. Displaying/Modifying Memory and Registers

### ■ Register Stack Pointer (RSP)

This field shows the contents of the register stack pointer. Modify them with the Change RSP... menu command.

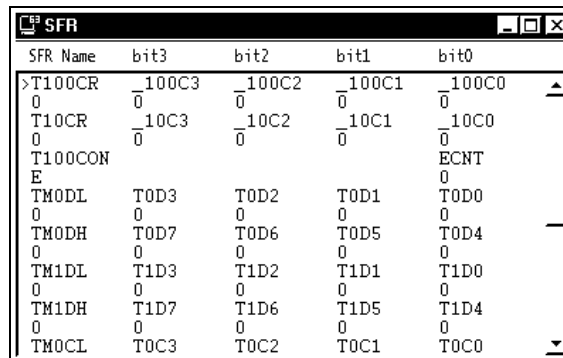
### ■ Stack Pointer (SP)

This field shows the contents of the stack pointer. Modify them with the Change SP... menu command.

## 6.3 Peripheral Registers (SFRs)

DT63K  
DTS63K

The contents of peripheral registers appear in an SFR window. To modify a register, right-click on it and choose the Change value menu command.



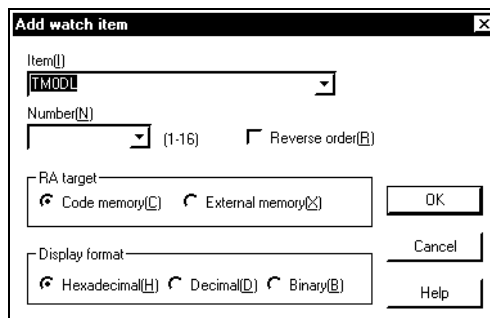
SFR Name	bit3	bit2	bit1	bit0
>T100CR	_100C3	_100C2	_100C1	_100C0
0	0	0	0	0
T10CR	_10C3	_10C2	_10C1	_10C0
0	0	0	0	0
T100CON				ECNT
E				0
TM0DL	T0D3	T0D2	T0D1	T0D0
0	0	0	0	0
TM0DH	T0D7	T0D6	T0D5	T0D4
0	0	0	0	0
TM1DL	T1D3	T1D2	T1D1	T1D0
0	0	0	0	0
TM1DH	T1D7	T1D6	T1D5	T1D4
0	0	0	0	0
TM0CL	T0C3	T0C2	T0C1	T0C0

### ■ Searching SFRs

The context menu also contains a Find menu command for searching the SFR list by name.

### ■ Adding to Watch List

To add an SFR to the watch list in the Watch window, right-click on it and choose the Add to watch menu command to display the Add watch item dialog box with the selected SFR in the Item field.



## 6.4 Code, Data, and External Memory

DT63K  
DTS63K

### ■ Dumping Memory

The View menu contains three items that open windows for displaying memory for editing: Code, Data, and External data.

An External data window provides a dump listing of external data memory contents.

A Data window is the counterpart for data memory. Note, however, that it blocks modification in the SFR region. Use an SFR window instead.

A Code window is the counterpart for program memory. Its primary use is for editing ROM table data. A Disassembly window is more useful for instructions because it displays instruction mnemonics and not just the machine code.

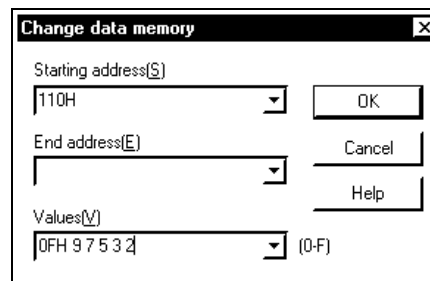
DTS63K

The DTS63K simulator requires that external memory first be enabled with the Tools menu's Options menu command.

DT63K  
DTS63K

### ■ Modifying Contents

Right-clicking on the window and choosing the Change menu command displays the following dialog box for modifying window contents.



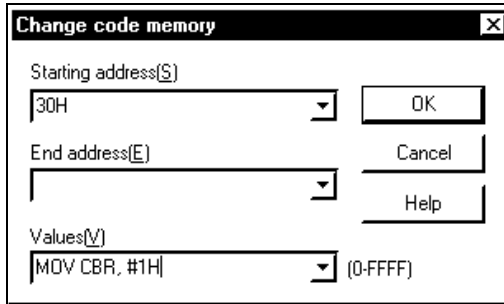
The Values list box accepts a list delimited with spaces, commas, or semicolons.

## 6. Displaying/Modifying Memory and Registers

DT63K  
DTS63K

### ■ Code Memory Assembler

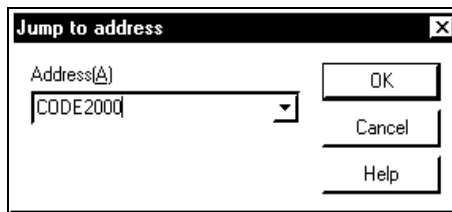
The Code window Values list box also accepts an assembly language instruction. In this case, clicking the left mouse button on the OK button does not close the dialog box. The dialog box acts as a single-line editor, incrementing the starting address field in preparation for the next instruction.



DT63K  
DTS63K

### ■ Jumping to an Address

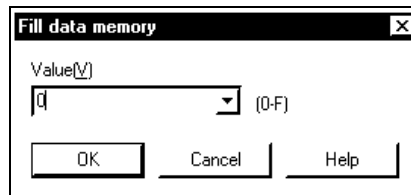
The Jump menu command on dump window context menus displays a dialog box for jumping to a specific address in that window.



DT63K  
DTS63K

### ■ Filling Memory

The Fill menu command on dump window context menus displays a dialog box for filling the entire memory with the same value—the quick way to initialize data or external data memory.



# 7. Watch Windows

---

## 7.1 Watch Window Usage

DT63K  
DTS63K

A Watch window is for displaying the contents of such items as data addresses, registers, and peripheral registers (SFRs).

Item specifications can use label symbols, indirect addressing with the HL, XY, and RA registers, and other complex addressing modes.

The View menu's Watch menu command opens a new Watch window. Note that you can open multiple Watch windows, each with its own list of items. When you first open a Watch window, this list is empty. To add items, right-click on the window and choose the Add to watch menu command.

## 7.2 Candidates

DT63K  
DTS63K

The following means are available for adding to Watch windows.

- Program counter
- Registers: A, H, L, HL, X, Y, XY, CBR, EBR, RA0, RA1, RA2, RA3, RA, RSP, and SP
- Flag bits: C, Z, G
- Data memory address
- Peripheral registers (by SFR name)
- Indirect address in HL
- Extra bank indirect address in HL
- Indirect address in XY
- Extra bank indirect address in XY
- Indirect program memory address in RA register
- Indirect external memory address in RA register
- PC-based address (PC+A)

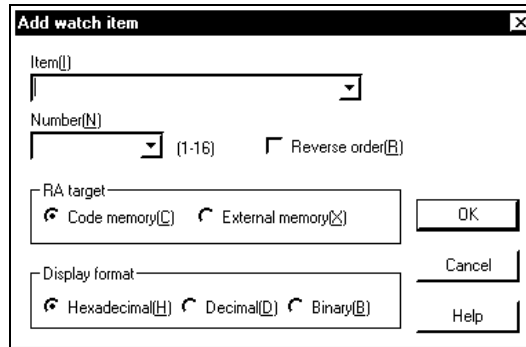
## 7. Watch Windows

### 7.3 Adding Items

DT63K  
DTS63K

#### ■ Watch Window

Right-clicking on the window and choosing the Add to watch menu command displays the following dialog box.

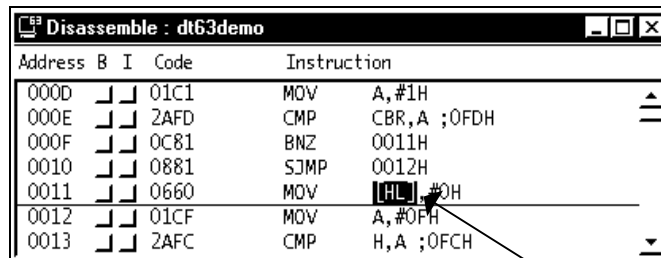


- Item:** Item to add
- Number:** Number of data items to display (data addresses only)
- Reverse:** Display data items in descending address order
- RA target:** Destination for RA register indirect addressing: code or external data memory
- Display format:** Radix for displaying data

#### ■ Source Window or Disassembly Window

To add a label or other identifier to the watch list, right-click on it and choose the Add to watch menu command.

To add text consisting of more than just a single word—[HL], for example—highlight it before right-clicking.



Selected text

#### ■ SFR Window

To add an SFR to the watch list, right-click on it and choose the Add to watch

menu command.

### ■ Symbol List

The Tools menu's Symbol list menu command displays a dialog box with all symbols. To add a symbol to the watch list, select it and click the left mouse button on the Add to watch menu command.

## 7.4 Saving Watch List

DT63K  
DTS63K

Right-clicking on the window and choosing the Save watch list menu command displays a dialog box for saving the current list of watch items to a disk file.

## 7.5 Loading Watch List

DT63K  
DTS63K

Right-clicking on the window and choosing the Load watch list menu command displays a dialog box for reading a watch item list back in—to avoid having the rebuild the same list several days in a row, for example. Note, however, that what is saved is only the specifications lists. Loading a watch list displays the current contents of the items, not the ones in effect when the file was saved.

## 7.6 Modifying Watch List

The Watch window context menu provides additional commands for modifying the list in the window.

DT63K  
DTS63K

### ■ Reordering Items

The Up menu command moves the selected item higher in the list; Down, lower.

### ■ Changing Radix

The Properties... menu command displays a dialog box with radio buttons for specifying the radix for the selected item.

### ■ Deleting Items

The Delete menu command removes the selected item from the list; Delete all, all items.

# 8. Tracing

---

## 8.1 Overview

DT63K  
DTS63K

When the microcomputer is connected to an embedded system busy controlling equipment, it is not always possible to interrupt execution with breaks for checking purposes or to use single-step execution. In such a situation, you can use the in-circuit emulator's trace function to take samples during emulation execution and debug by analyzing the results of that sampling. This function records operational data at key points in trace memory. To view the contents, open a Trace window. (See Section 8.2 "Trace Windows" below.)

### ■ Displaying Trace Pointer

The trace pointer indicates the current trace entry position in the trace memory. The current contents appear in the Status window.

### ■ Clearing Trace Memory

The Trace menu's Clear trace memory command clears the trace memory.

### ■ Trace Memory Size

The in-circuit emulator attached to the DT63K debugger has room for 8192 trace entries; the DTS63K simulator, 65,536.

When the trace memory is full, the trace pointer automatically wraps to 0 upon overflow so that new trace entries overwrite the old.

## 8.2 Trace Windows

DT63K  
DTS63K

The View menu's Trace menu command opens a Trace window. If the trace memory is empty, so is the window.

This window displays the trace items in chronological order: oldest item at the top, most recent item at the bottom.

Across the top of the window are labels for the individual fields.

### ■ Trace Pointer (Loc)

This column specifies the trace pointer, an index to the trace memory entries. It also provides a rough time scale for the contents.

### ■ Program Counter (PC)

This column normally gives the program counter location at the time of the trace.

### ■ Accumulator (A)

This column normally gives the contents of the Accumulator (ACC) after execution of the instruction at the code address in the PC column.

### ■ RAM Address (RADR)

This column gives the last data address accessed by an instruction writing to memory.

### ■ RAM Data (RD)

This column normally gives, depending on the instruction traced, either the last value written to data memory.

### ■ Flags (CGZ)

These columns give the contents of the carry (C), G, and zero (Z) flags after execution of the instruction at the code address in the PC column.

### ■ Master Interrupt Enable Flag Register (MI)

This column gives the contents of the Master Interrupt Enable (MIE) flag after execution of the instruction at the code address in the PC column.

### ■ Melody Request Flag (MD)

This column gives the contents of the Melody Request Flag (MD) after execution of the instruction at the code address in the PC column.

### ■ Current Bank Register (CB)

This column gives the contents of the Current Bank Register (CBR) after execution of the instruction at the code address in the PC column.

### ■ Extra Bank Register (EB)

This column gives the contents of the Extra Bank Register (EBR) after execution of the instruction at the code address in the PC column.



## 8. Tracing

### ■ HL Register (HL)

This column gives the contents of the HL Register (HL) after execution of the instruction at the code address in the PC column.

### ■ XY Register (XY)

This column gives the contents of the XY Register (XY) after execution of the instruction at the code address in the PC column.

### ■ PC Stack Pointer (SP)

This column gives the contents of the Stack Pointer (SP) after execution of the instruction at the code address in the PC column.

### ■ Register Stack Pointer (RS)

This column gives the contents of the Register Stack Pointer (RSP) after execution of the instruction at the code address in the PC column.

DT63K

### ■ External Probe Data (XP)

This column gives the contents of the states of external probe pins after execution of the instruction at the code address in the PC column.

DT63K  
DTS63K

### ■ Interrupt Notification (I)

A “1” in this column indicates an interrupt during execution of the instruction at the code address in the PC column. Tracing switches to the interrupt service routine from the next line.

### ■ External/ROM Data Access Notification (E)

A “1” in this column indicates that this entry contains the external data memory address or internal ROM table address accessed by the instruction at the code address in the PC column for the preceding entry—in other words, that the entry represents a continuation of the preceding entry and has that address in the PC column.

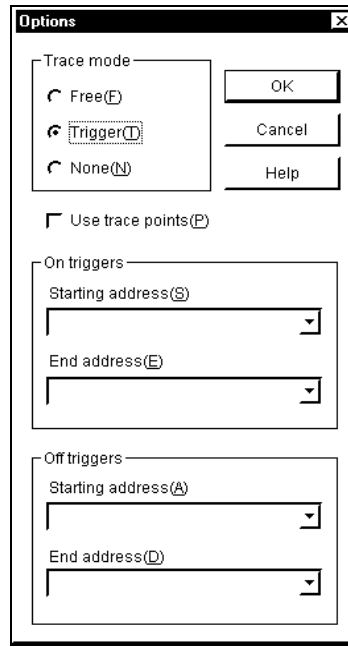
### ■ Source/Disassembler

The context menu’s Properties... menu command offers a choice for this column: source code or disassembler output for the instruction currently at the code address in the PC column.

## 8.3 Tracing Options

DT63K

The Trace menu's Settings menu command displays the following dialog box.



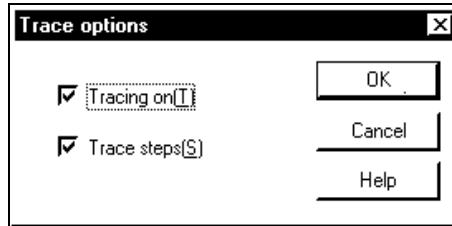
- Free:** Remove address limits on tracing
- Trigger:** Toggle tracing using address ranges specified below
- None:** Disable tracing
- Use trace points:** Limit tracing to trace points. Note that this option can be freely mixed with the address triggers.
- On triggers:** Enable tracing whenever the program counter visits a code memory address in the specified range. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank makes the first one the only trigger.
- Off triggers:** Disable tracing whenever the program counter visits a code memory address in the specified range. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank makes the first one the only trigger.

Note that the trigger trace mode traces the execution of instruction at on trigger addresses, but not the ones at off trigger addresses.

## 8. Tracing

DTS63K

The DTS63K simulator Trace window's Options menu command displays the following dialog box.



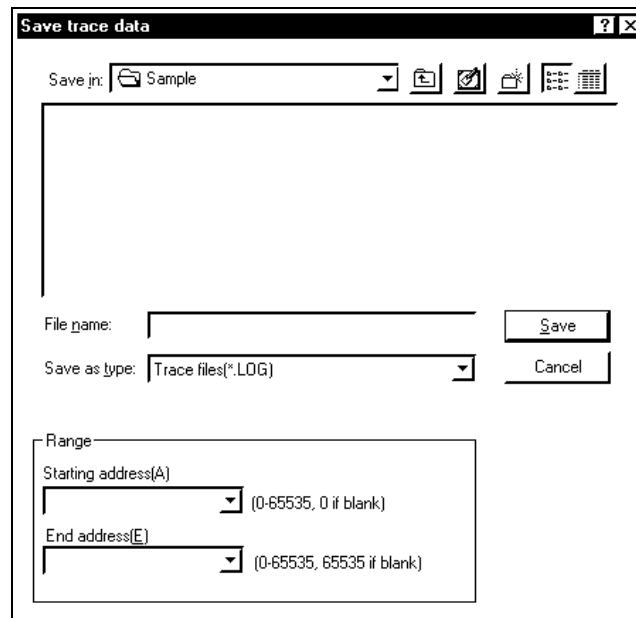
Selecting the Tracing on check box enables tracing.

Selecting the Trace steps check box enables tracing during single-step execution. Note that this check box is only available when the first is selected.

## 8.4 Saving Trace Data

DT63K  
DTS63K

The Trace menu's Save trace data menu command displays the following dialog box for saving the trace memory contents to a disk file.

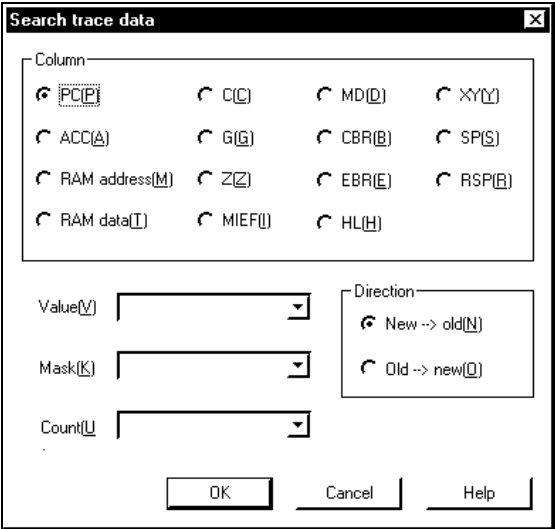


The list boxes at the bottom allow you to limit the operation to a specific address range. Leaving them blank extends coverage to the entire trace memory.

## 8.5 Searching Trace Data

DT63K  
DTS63K

Right-clicking on the window and choosing the Search menu command displays the following dialog box for searching individual columns for data matching a specific masked value.



- Column:** Column to search
- Value:** Value to match
- Mask:** Mask for data
- Count:** Repetition count
- Direction:** Direction to search

# 9. Cycle Counter

---

## 9.1 Overview

DT63K  
DTS63K

If your system demands real-time response, you frequently wish to know precisely how long certain sequences take to execute. The software can help by counting execution cycles.

This execution cycle counting function works by counting the clock cycles fed to the target device while it is executing. Note that with OLMS-63K microcomputers, the number of cycles depends on the instruction. The cycle counter counts all such cycles.

### ■ Displaying Cycle Counter

The current cycle counter contents appear in the Status window.

The context menu's Properties... menu command provides a choice of count or time formats.

### ■ Changing Cycle Counter

To change the cycle counter, choose the CycleCounter menu's Change menu command.

DT63K

### ■ DT63K Debugger Cycle Counter

The cycle counter in the in-circuit emulator is a 24-bit unsigned one that overflows after 16,777,215.

DTS63K

### ■ DTS63K Simulator Cycle Counter

The cycle counter in the DTS63K simulator is a 31-bit unsigned one that overflows after 2,147,483,647. In the DTS63K, when the cycle counter is set to be displayed as a time, both the number of cycles executed in high-speed program operation and the number of cycles executed in low-speed program execution are displayed converted to time values.

DT63K  
DTS63K

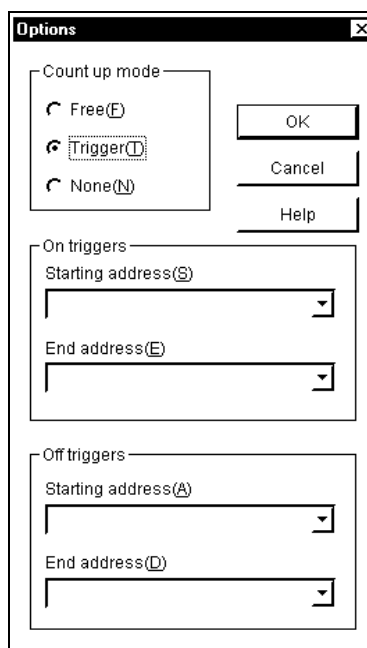
### ■ Cycle Counter Overflow

Both the DT63K debugger and DTS63K simulator allow the use of cycle counter overflow to trigger a break. For further details, see Chapter 5 "Break Functions."

## 9.2 Options

DT63K

The DT63K debugger CycleCounter menu's Options menu command displays the following dialog box.



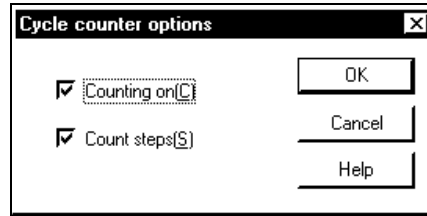
- Free:** Remove address limits on counting
- Trigger:** Toggle counting using address ranges specified below
- None:** Disable counting
- On triggers:** Enable counting whenever the program counter visits a code memory address in the specified range. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank makes the first one the only trigger.
- Off triggers:** Disable counting whenever the program counter visits a code memory address in the specified range. Leaving both these fields blank extends coverage to the entire code memory. Leaving the second one blank makes the first one the only trigger.

Note that the trigger count mode counts the execution of instruction at on trigger addresses, but not the ones at off trigger addresses.

## 9. Cycle Counter

DTS63K

The DTS63K simulator CycleCounter menu's Options menu command displays the following dialog box.



Selecting the Counting on check box enables counting.

Selecting the Count steps check box enables counting during single-step execution. Note that this check box is only available when the first is selected.

# 10. Source Level Debugging

## 10.1 Source Windows

DT63K  
DTS63K

The software supports basic debugging from Source windows displaying the source code for the user application program. Debugging is much more efficient when you have the original easy-to-read, fully commented, formatted source code to refer to instead of just a disassembler listing.

The View menu's Source menu command opens a Source window.

```

C:\User\dt63k\INIT.ASM
Line  BP Address
00009   0006H      [EBR] = 0
00010
00011   ;          RAM clear
00012   0007H      HL = 0
00013   0009H      [CBR] = 1
00014
00015   ; This repetition is repeated until CBR=3. [100H 2FFH] <-- 0H
00016   REPEAT
00017   000AH      IF ([H] < 0AH) || ([CBR] != 1)
00018   ; skip addresses 1A0H - 1FFH
00019   0011H      [HL] = 0
00020   ENDI
00021
00022   0012H      IF ([H] == 0FH) && ([L] == 0FH)
00023   0018H      [CBR]++
00024   ENDI
  
```

**Line:** Line number in the source file

**BP:** Check box for setting breakpoints

**Address:** Code address

The green band indicates the line containing the program counter; the underline, the current cursor position.

### ■ Preparing Source Files

Source windows require source files in a special format from the SASM63K structured assembler. For further details, see Chapter 2 “Before You Start.”

### ■ Switching Source Files

If the user application program combines files using INCLUDE directives, you will need a separate Source window for each one. Switch between them with the Change source windows menu command.



## 10. Source Level Debugging

### ■ Setting Breakpoints

The BP column contains check boxes for breakpoints. Clicking one changes the color to red to indicate a breakpoint. Clicking a second time cancels the setting.

### ■ Executing to Cursor Line

To execute the program from the current program counter position up to a particular address in a Source window, right-click on the line for that address and choose the Execute to cursor menu command from the context menu. If the line does not contain executable code, however, nothing happens.

Alternate methods for moving the cursor offered by the context menu are the Jump to address and Jump to line menu commands.

## 10.2 Links to Other Windows

DT63K  
DTS63K

### ■ Switching to Disassembly Window

Choosing the Disassemble menu command on the Source window context menu displays the cursor line in a Disassembly window. If the line does not contain executable code, however, nothing happens.

### ■ Switching from Trace Window

Double-clicking on an address in the PC column of a Trace window displays the corresponding line in a Source window. If there is no source line information file loaded, however, the corresponding line in a Disassembly window appears instead.

### ■ Adding to Watch Window

Selecting text in a Source window and choosing the context menu's Watch menu command adds that text to the Watch window. For further details, see Chapter 7 "Watch Windows."

# 11. Symbolic Debugging and Input Formats

---

## 11.1 Symbols

DT63K  
DTS63K

Symbols are stand-ins for specific numerical values. They allow you to use easy-to-remember names instead of numbers for addresses and constants.

The software uses the following rules to recognize symbols in expressions.

- A symbol begins with a letter (A–Z or a–z) or certain special characters ( \_, \$, and ?).
- Subsequent characters may be the above or digits (0–9).
- The following characters serve as delimiters indicating the end of the symbol:
  - whitespace characters tab (09H), line feed (0AH), carriage return (0DH), and space (20H)
  - operators +, -, \*, /, %, &, |, ^, !, ., <, >, =, (, and )
  - other symbols ¥, [, ], {, }, :, ;, @, #, ,, ' , and comma (,)

**Note:** Only the first 32 characters of a symbol are valid.

### ■ Preparing Symbol Tables

Symbolic debugging requires debugging files in a special format prepared by supplying the /d to the SASM63K structured assembler.

### ■ Symbol Value

This is the value assigned to the symbol.

### ■ Symbol Attributes

Symbols have other attributes beside value.

**CODE:** Code address symbol

**DATA:** Data address symbol

**XDATA:** External data address symbol

## 11. Symbolic Debugging and Input Formats

**BIT:** Bit address symbol

**NUMBER:** Numerical value

### ■ Symbol Type

This gives the scope in which the symbol is defined. This type plays an important role in symbol searches. Types available are: reserved word and user-defined symbol.

Reserved words are symbols defined in the device-specific .dcl file that the software reads in during start-up. They are usually the names of peripheral registers (SFRs) for the microcomputer being evaluated. You cannot modify these symbols.

User-defined symbols are symbols defined in the user application program as labels or with SASM63K structured assembler directives EQU, CODE, DATA, XDATA, etc. They are thus eligible for modification or even deletion.

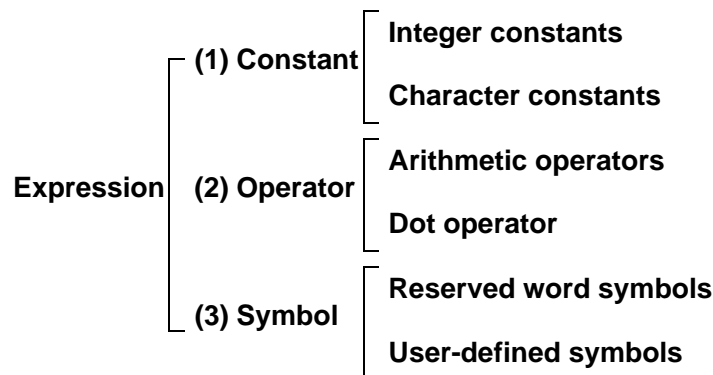
### ■ Displaying Symbol List

The Tools menu's Symbol list menu command displays a dialog box listing all reserved words and user-defined symbols. This dialog box also provides facilities for adding, deleting, and changing the values assigned to user-defined symbols.

## 11.2 Expressions

DT63K  
DTS63K

The software supports expressions built up from three basic components: constants, operators, and symbols. Before discussing the expression syntax itself, let us define these components.



### 11.2.1 Constants

DT63K  
DTS63K

Constants directly specify addresses and data values. There are two basic types: integer and character constants.

#### ■ Integer Constants

These are strings beginning with a digit (0–9) and consisting entirely of the characters listed for the radix (2, 8, 10, or 16) in the table below.

If a string starts with a character other than a digit—i.e., A–F, a–f, or underscore—prefix it with a zero to prevent parsing as a symbol. Underscores and case are ignored.

Radix	Permissible Characters	Suffixes	Examples
2	0, 1, _	B	1010B
8	0~7, _	O, Q	271O, 514Q
10	0 to 9, _		1234
16	0 to 9, A to F, a to f, _	H	753H, 0C6E7H

#### ■ Character Constants

These consist of a pair of single quotes (') around a single character or a (pre-ANSI) C-like escape sequence starting with ASCII character 5CH, the backslash (or yen sign in Japanese fonts).

In the first case, the integer constant has the 8-bit value corresponding to the ASCII code for that character.

A backslash after the opening quote introduces an escape sequence, which specifies the integer constant as an 8-bit octal or hexadecimal constant between 00H and 0FFH.

## 11. Symbolic Debugging and Input Formats

### 11.2.2 Operators

DT63K  
DTS63K

The software evaluates expressions using unsigned 32-bit arithmetic, automatically converting negative values to their two's complement form and ignoring any overflow.

#### ■ Arithmetic Operators

- + Binary addition
- Binary subtraction
- & Binary AND

#### ■ Dot Operator

- . Equivalent to left operand \* (number of bits) + right operand

This operator expresses a bit address relative to a data address.

## 11.3 Value Lists

DT63K  
DTS63K

In a dump list window, the context menu's Change xxx... menu command displays a dialog box with a Values list box. This accepts a list delimited with spaces, commas, or semicolons.

# 12. Other Functions

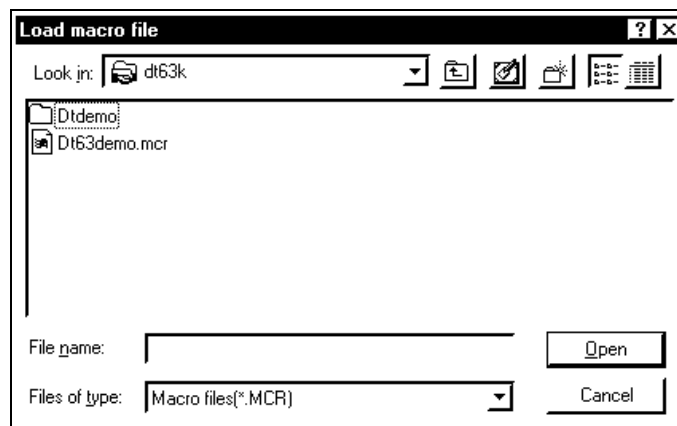
---

## 12.1 Macros

DT63K  
DTS63K

The software includes a text-based macro facility that executes text-based scripts created in a text editor. All output goes to the Log window. A complete command listing appears in Section 15.4 “Macro Command Reference.”

The Tools menu’s Macro menu command displays the following dialog box for choosing a macro file.



During macro execution, all output goes to the Log window. You can, if you wish, save the contents of this window to a disk file.

## 12. Other Functions

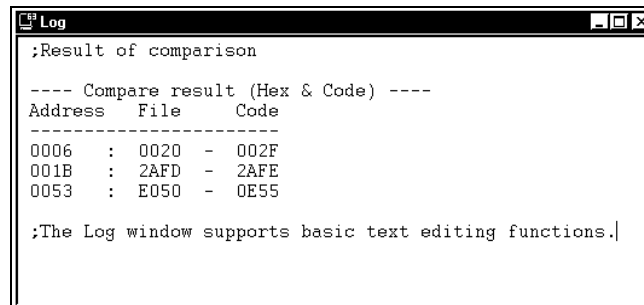
### 12.2 Log Window

DT63K  
DTS63K

The Log window keeps a record of debugging and simulation operations.

It also provides a simple text editor for editing the window's contents: deleting unnecessary portions, searching and replacing strings, adding comments to macro output, etc.

The Tools menu's Log menu command opens the Log window.



```
Log
;Result of comparison
---- Compare result (Hex & Code) ----
Address  File      Code
-----
0006   : 0020 - 002F
001B   : 2AFD - 2AFE
0053   : E050 - 0E55

;The Log window supports basic text editing functions. |
```

This window holds the following data.

- Output from macro execution
- Results of comparing code memory and program files
- Results of comparing code memory and EPROM (DT63K debugger only)
- Text

### 12.3 On-Line Help

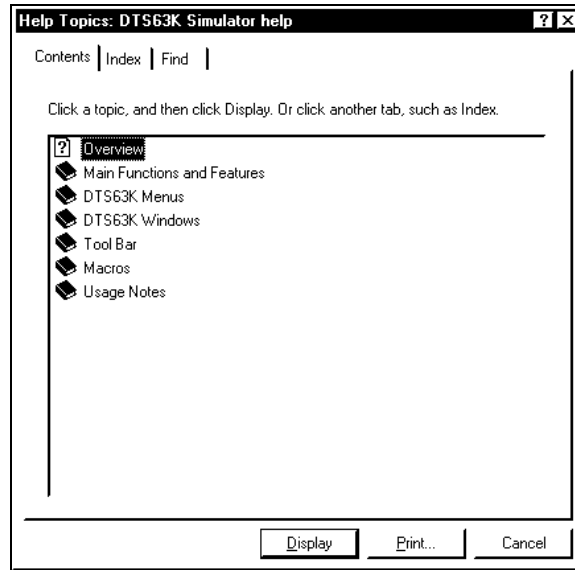
DT63K  
DTS63K

The software includes a Windows help file providing assistance for all menus, context menus, windows, and dialog boxes.

When a menu or dialog box is on the screen, pressing the F1 key displays the related portion of this file. Otherwise, the key is equivalent to the Help menu's Help topic menu command, which displays an index for the help file.

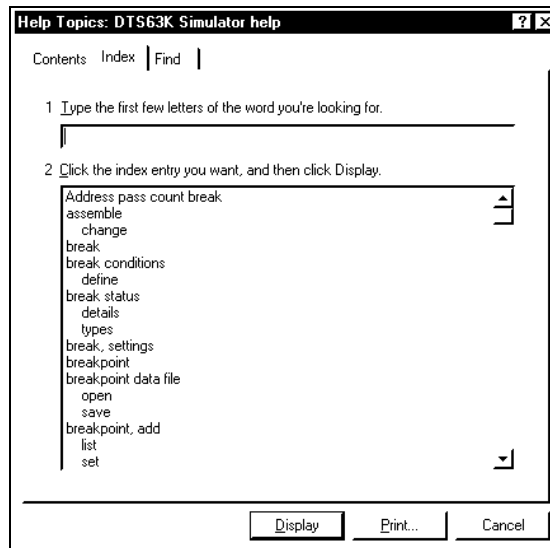
#### ■ Contents

Clicking the Contents tab displays the table of contents for the help file. Double-clicking an item either expands that portion of the outline or displays the text for that topic.



■ **Keywords**

Clicking the Keywords tab displays an index that automatically searches for keywords as you type.



## 12.4 Version Information

**DT63K  
DTS63K**

The Help menu's About menu command displays a version number and other information about the software.

**DT63K**

The DT63K debugger displays hardware and firmware version numbers for the connected emulator as well.



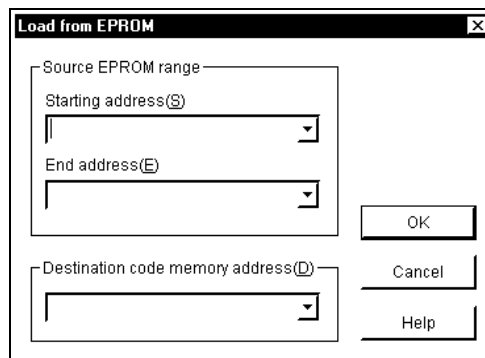
## 12.5 Reading/Verifying EPROMs

DT63K

These functions are available only for in-circuit emulators with EPROM sockets.

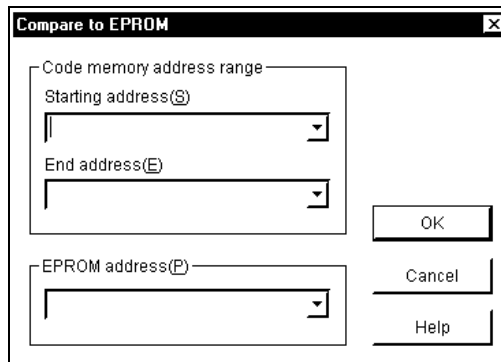
### ■ Loading Code from EPROM

The Tools menu's Load from EPROM menu command displays the dialog box for loading all or some of the contents of the EPROM into code memory.



### ■ Comparing Code to EPROM

The Tools menu's Compare to EPROM menu command displays the dialog box for comparing all or some of the contents of code memory to the EPROM.



# 13. DTS63K Simulation Functions

## 13.1 Instruction Execution

**DTS63K**

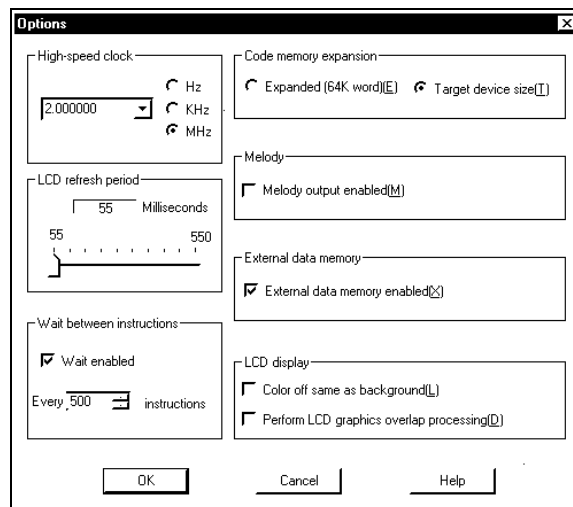
The software simulates nX-4/250 core instructions at the instruction level and makes no guarantees as to the accuracy of timing. Hardware-dependent portions require particular care because the timing depends on development host performance during the computations necessary for the simulation, not the target device's true execution cycle.

The Run menu's Run program menu command starts continuous execution. This execution continues until a break condition is satisfied or you force a break with the Run menu's Stop execution menu command.

This section describes the features of this simulation.

### ■ Operating Frequency

The only operating frequency under user control is the high-speed clock. The controls are on the dialog box displayed by the Tools menu's Options menu command.



### ■ Wait Between Instructions

Because the simulation of the target device and peripherals takes place entirely within the memory of the development host, instruction execution speed can be

## 13. DTS63K Simulation Functions

much faster. The dialog box displayed by the Tools menu's Options menu command therefore provides a wait setting for slowing down the simulation.

When a wait time is set, operation timing in which waits can be inserted once every few instructions can be specified by checking the "Wait setting" check box.

### ■ Instruction Set

The simulation covers all nX-4/250 core instructions.

### ■ Memory Spaces

The simulation supports the following memory spaces.

- Program memory (max. 64K words)
- Data memory (max. 4K nybbles)
- External data memory (max. 64K bytes)

### ■ Register Set

The software simulates the operation of the following registers.

- Accumulator (A)
- Flag register (C, Z, and G flags)
- Master interrupt enable (MIE) flag
- Current bank register (CBR)
- Extra bank register (EBR)
- HL register
- XY register
- Program counter (PC)
- RA registers (RA0, RA1, RA2, and RA3)
- Stack pointer (SP)
- Register stack pointer (RSP)

## 13.2 Resets

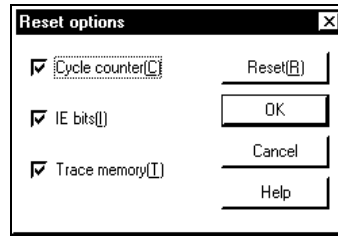
DTS63K

When the DTS63K simulator loads or you choose the Run menu's Reset simulator menu command, the simulation resets itself to imitate the target device's reset cycle when the device first receives power or subsequently receives reset input.

The reset sequence includes the following steps.

- Resetting the nX-4/250 core
- Initializing peripheral registers (SFRs)
- Resetting the break status to “Not yet executed”

The Run menu’s Reset options menu command provides controls for optionally clearing the cycle counter, IE bits, and trace memory as well.



Note that the DTS63K simulator does not support resets with RESET pin input.

## 13.3 Onboard Peripherals

**DTS63K**

The software simulates the following peripheral functions built into the target device.

### ■ Time Base Counter (TBC)

The software simulates a 15-bit time base counter. This counter increments at the falling edge of the 32.769-kHz time base clock signal. Counter outputs include the time base interrupt and operating clock signals for circuit modules.

After a reset, the counter contains the value 0000H.

### ■ Timers

The software simulates four 8-bit timers, numbered 0 to 3, but only in their auto reload modes. It does not support the capture mode offered by timers 0 and 1 or the frequency measurement mode offered by all four.

The only choices for the timer input clocks are the 32.769-kHz time base clock and the high-speed clock. There is no support for external clocks.

### ■ 100-Hz Timer/Counter

The software uses internal calculations to simulate a 100-Hz timer/counter. The one in actual devices derives its input clock from the 512-Hz TBC6 output with a

## 13. DTS63K Simulation Functions

frequency divider.

### ■ Watchdog Timer (WDT)

The software simulates a watchdog timer, running off TBC7 output, for automatically interrupting a program that has run out of control.

### ■ Ports

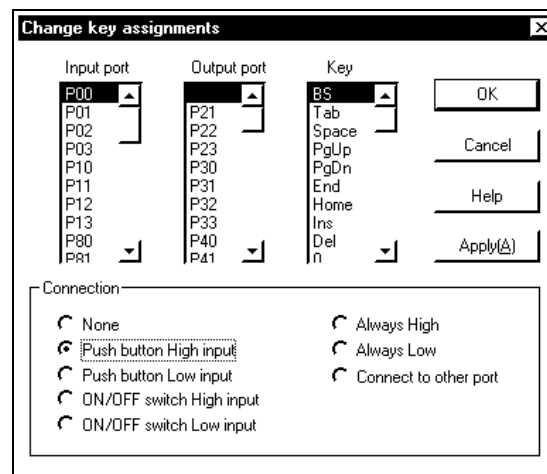
The software simulates all port registers, port control registers, port interrupt enable registers, and port mode registers available for the target device.

It also simulates most secondary port functions—except for the following.

- Extended LCD driver frame output
- Extended LCD driver clock output
- 32.768-kHz crystal oscillator clock output
- High-speed oscillator clock output
- Shift register data input
- Shift register data output
- Shift register clock I/O
- Capture input for timers 0 and 1
- External clock input for timers 0 to 3

The I/O port simulation supports two methods of assigning port inputs to keys on the development host's keyboard: one-to-one mapping and a key matrix.

The first approach uses the Change key assignments... menu command from the Tools menu context or the Port window's context menu.



Select the input port and key from the corresponding columns and specify the connection with the radio buttons: push-button or on/off switch, “High” level input or “Low.”

The Connect to other port radio button enables the output port column, connecting the selected output port to the input port.

The key matrix approach uses the matrix variant of the Port window. For further details, see Section 3.4.2 “Assigning Ports.”

The port registers, port control registers, port interrupt enable registers, and port mode registers are accessible in the SFR window.

### ■ External Data Memory

The software uses 64K of memory in place of the external data memory and simulates the related secondary port functions. It does not simulate expansion with a chip select (CS) or the operation of the external circuitry, however.

### ■ Melody Driver

The software simulates the melody driver, but the speed difference between simulated and hardware instruction produces a large difference in the timing for the end of the melody.

The software converts the melody data from the ROM table into .WAV format for output to the development host speaker.

### ■ Buzzer Driver

The software simulates the buzzer driver, converting buzzer output into beeps from the development host speaker.

### ■ Serial Ports

The simulation does not cover these.

### ■ Shift Registers

The simulation does not cover these.

### ■ LCD Driver

The simulation covers the following portions of the LCD driver.

- Maximum number of dots supported by target device
- Duty selection
- All segments off

## 13. DTS63K Simulation Functions

- All segments on
- 16 contrast levels
- Power down mode

It does not support bias selection.

The simulated LCD screen appears in the LCD window. For further details, see Section 13.5 “LCD Simulation” below.

### ■ Multiplier/Divider Circuit

The simulation covers the multiplier/divider circuit, substituting the appropriate calculations. It does not simulate the circuit itself.

### ■ Battery Low Detector

The simulation does not include the circuit for detecting low battery level. The corresponding control register, however, still appears in the SFR window.

### ■ Backup Circuit

The simulation does not cover the backup circuit. The corresponding control register, however, still appears in the SFR window.

## 13.4 Interrupt Simulation

DTS63K

The software simulates the interrupt service routines for processing interrupts arriving during program execution. This simulation completely supports stack operations, the disable interrupt instruction, and other related details.

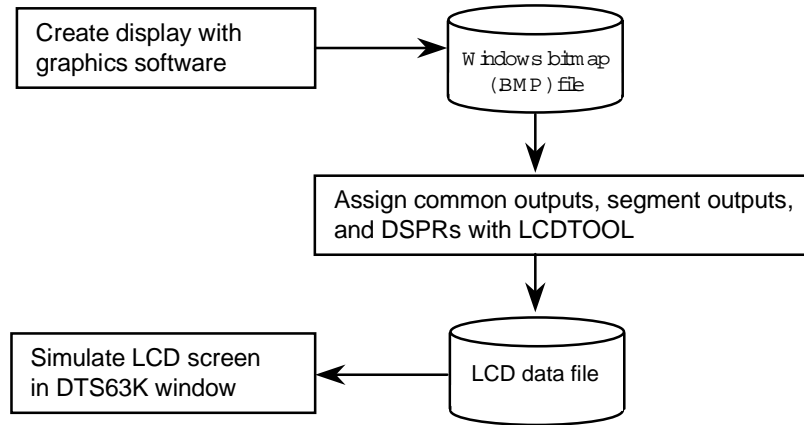
The simulation covers all interrupts, but external interrupts are limited to those generated from keys mapped to the corresponding ports in the Port window.

An interrupt during HALT mode release the simulated target device from that mode.

## 13.5 LCD Simulation

DTS63K

A major simulator feature is its simulation of an LCD screen in the LCD window. Using it, however, requires some outside preparation.



1. Create the LCD screen with a graphics package and save it in Windows bitmap (.BMP) format.
2. Assign common outputs, segment outputs, and display registers (DSPRs) with LCDTOOL. If desired, add a dot matrix. Save the results as an LCD data file.
3. Load the LCD data file into the simulator with the File menu's Load LCD graphics menu command.
4. Open the LCD window to display the bitmap.
5. Run the program to see its effects on the simulated LCD screen in the LCD window.

For further details on the LCDTOOL utility, see Chapter 14 "LCDTOOL Utility" below.

#### ■ LCD Screen Refresh Period Setting

The DTS63K simulator supports setting the LCD window screen refresh period during simulation.

The LCD screen refresh period can be set to a value in the range 55 to 550 ms.

To set the LCD screen refresh period, select Operation settings from the Tools menu and use the LCD Display item in the dialog box.



# 14. LCDTOOL Utility

---

## 14.1 Overview

**LCDTOOL**

LCDTOOL is a utility that prepares LCD data files for the DTS63K simulator. Either load it directly (LCDTOOL.EXE) or with the Tools menu's LCDTOOL menu command.

The utility takes as input a Windows bitmap (.BMP) created with a suitable graphics package. Its function is to assign common outputs, segment outputs, and display registers (DSPRs) to drive this display.

When these assignments are complete, save the results as an LCD data file for loading into the simulator.

## 14.2 Procedure

**LCDTOOL**

The procedure for creating an LCD data file for loading into the simulator consists of the following steps.

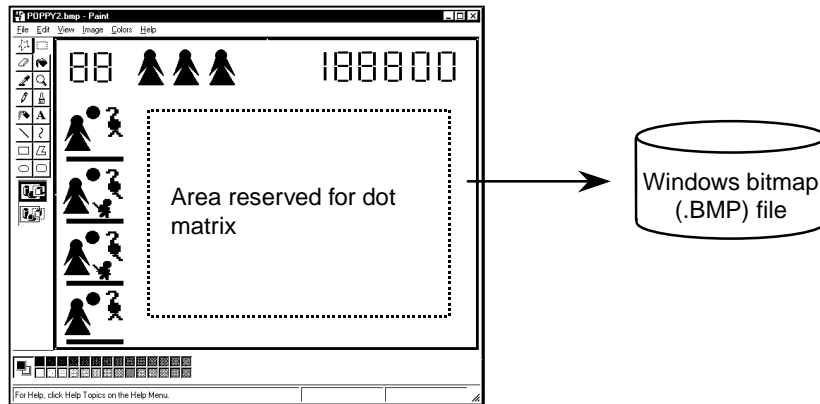
1. Create the .BMP file.
2. Load this file into LCDTOOL.
3. (Optional) Add dot matrix.
4. Make assignments.
5. Save results as LCD data file.

### 14.2.1 Creating .BMP File

**LCDTOOL**

The first step is creating the desired LCD screen image. One tool for the job is MS Paint, included with Windows, but any graphics package capable of saving in Windows bitmap (.BMP) format will do.

If the display uses a dot matrix, leave that area blank. LCDTOOL contains a convenient tool for automatically adding dot matrices.

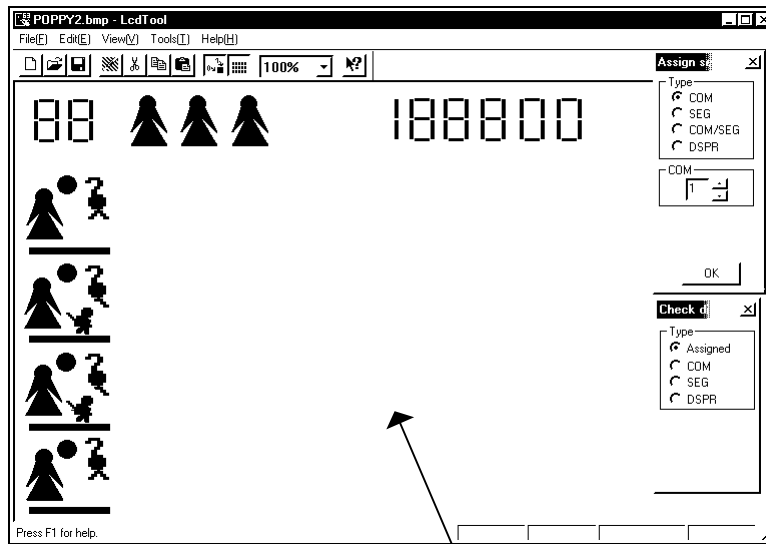


### 14.2.2 Loading .BMP Data

**LCDTOOL**

The next step is to read the graphics data into the utility with the File menu's Open menu command.

The graphic then appears on the screen with tool palettes.



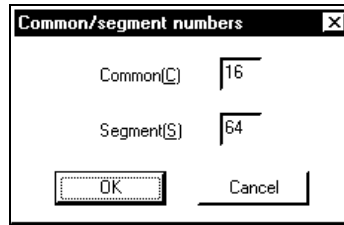
Graphics data read

### 14.2.3 Creating Dot Matrix

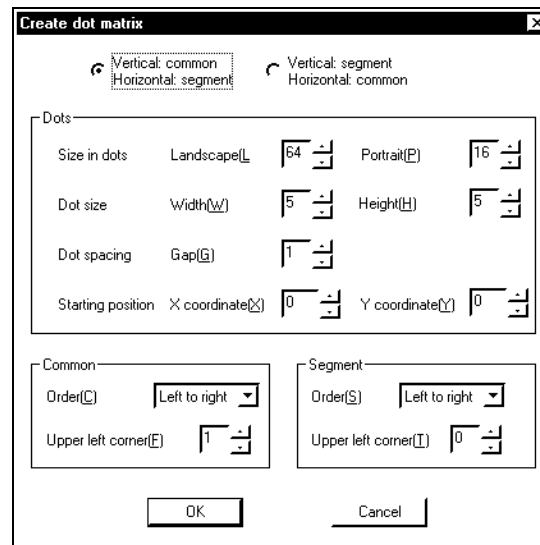
**LCDTOOL**

While the size of the dot matrix is set to a default of 16 common lines by 32 segment lines, those values can be changed using the Common/segment numbers item in the Tools menu.

## 14. LCDTOOL Utility



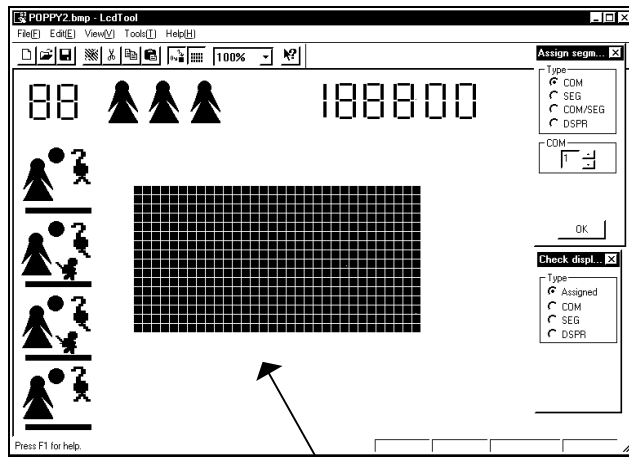
To add a dot matrix to this display, choose the Tools menu's Create dot matrix menu command to display the following dialog box for specifying the size and other parameters.



The two radio buttons at the top specify the orientation for common vs. segment outputs.

The Dot count controls specify the number of dots in the vertical and horizontal directions. The maximum values of those settings are the number of common lines (vertical) and number of segment lines (horizontal) settings specified with the Common/segment numbers item in the Tools menu. The two controls in the second line specify the dot height and width in pixels. The third line gives the spacing between dots in pixels; the fourth, the coordinates of the upper left corner of the matrix.

The controls in the Common and Segment areas specify the order for assigning outputs from left to right and the starting number. Selecting None for the former disables automatic assignment.



New dot matrix

### 14.2.4 Making Assignments

**LCDTOOL**

To assign common and segment outputs to a bit map region, click on it to display the Assign segments dialog box. To select multiple regions, hold down the Ctrl key for each one except the last. Note that assigned regions appear in light blue.

The Type radio buttons portion of the dialog box specifies the type of assignment and the layout of the bottom half.

**COM:** Common

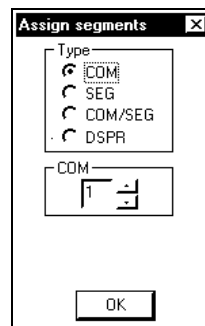
**SEG:** Segment

**COM/SEG:** Both common and segment

**DSPR:** Display register value

#### ■ Assigning Common Outputs

Selecting the COM radio button displays the following dialog box.

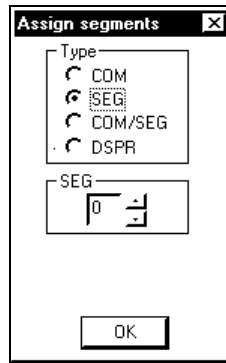


Specify the common output number and click the left mouse button on the Assign button.

## 14. LCDTOOL Utility

### ■ Assigning Segment Outputs

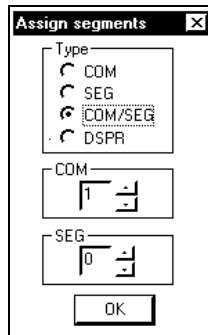
Selecting the SEG radio button displays the following dialog box.



Specify the segment output number and click the left mouse button on the Assign button.

### ■ Assigning Common and Segment Outputs

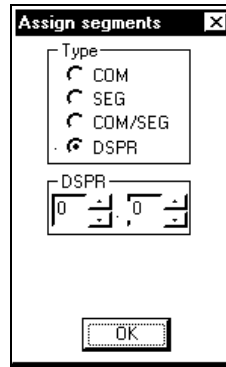
Selecting the COM/SEG radio button displays the following dialog box.



Specify the common and segment output numbers and click the left mouse button on the Assign button.

### ■ Assigning by DSPR Value

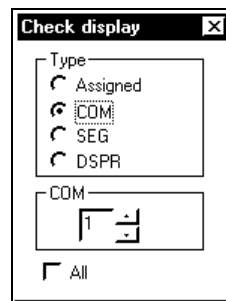
Selecting the DSPR radio button displays the following dialog box.



Specify the display register and bit numbers and click the left mouse button on the Assign button. The mapping between display register bits and outputs appears in the User's Manual for the target device.

### ■ Displaying Assignments

The Force display dialog box is for checking assignments for bitmap regions.



**Assigned:** All assigned bitmap regions

**COM:** Bitmap regions for specified common output

**SEG:** Bitmap regions for specified segment output

**DSPR:** Bitmap regions for specified display register

Selecting the All check box lights up all bitmap regions with assignments of the specified type.

Moving the mouse cursor to a bitmap region displays its current common, segment, and DSPR assignments on the status bar.

## 14.2.5 Saving LCD Data File

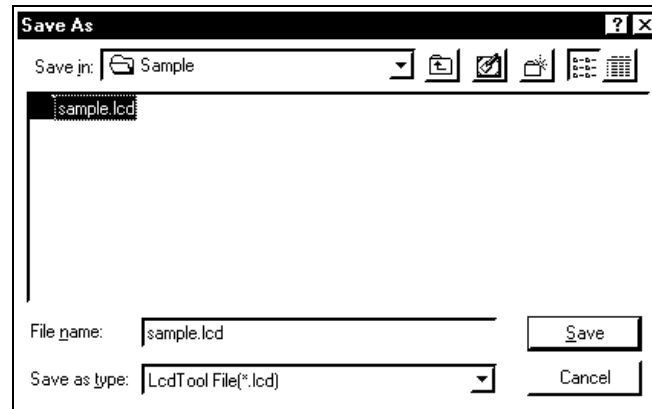
**LCDTOOL**

When the assignments are complete, save the results as an LCD data file for loading into the simulator.

The File menu provides two choices: Save and Save as. The latter is the only one

## 14. LCDTOOL Utility

available when you have just loaded the .BMP file. The former is available only if you have already saved the data and then edited it.

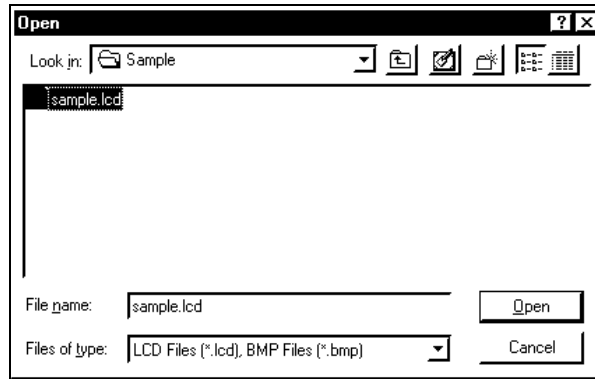


The default extension for LCD data files is .LCD.

## 14.3 Editing LCD Data Files

**LCDTOOL**

The File menu's Open menu command is for loading a previously saved LCD data file for additional editing.



### ■ Undoing an Edit Operation

The Undo item on the Edit menu undoes the effect of the last edit operation.

### ■ Moving Bitmaps

To move a bitmap region, click on it and drag it to the new position.

### ■ Deleting Bitmaps

The Edit menu offers two choices for deleting the selected bitmap region from the screen: Cut and Delete. The only difference is that the former moves the data to the clipboard, making it available for pasting with the Paste menu command.

### ■ Copying Bitmaps

The Edit menu's Copy menu command copies the selected bitmap region to the clipboard, from which it is available for pasting with the Paste menu command.

### ■ Bitmap grouping

Multiple bitmaps can be grouped and handled as a single object by selecting the bitmaps to be grouped and then selecting the Group item on the Edit menu. The grouped items can be restored to be individual items by selecting the Ungroup item on the Edit menu.

### ■ Canceling Assignments

To cancel assignments to a bit map region, select it and choose the Edit menu's Cancel assignment menu command.

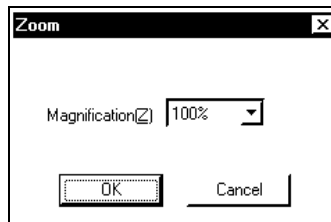


## 14.4 Other Functions

### 14.4.1 Zoom

LCDTOOL

The software provides a convenient zoom control function for enlarging or reducing the size of the image on the screen for ease in editing. Choose the View menu's Zoom menu command to display the following dialog box and select the new magnification factor.

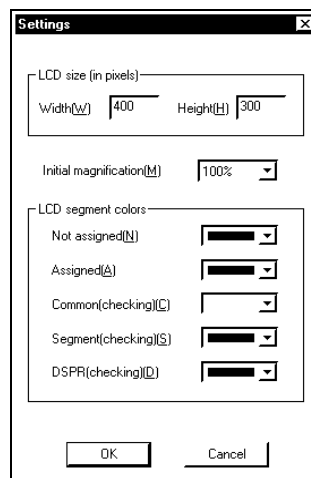


The list box offers the following choices: 10%, 25%, 50%, 75%, 100%, 150%, 200%, 400%, and 800%.

### 14.4.2 Operating Settings

LCDTOOL

The Tools menu's Settings menu command displays the following dialog box for configuring operation.



The controls in the first box specify the height and width of the LCD screen in pixels.

Next comes the default magnification to use when the simulator loads the LCD data file.

The large box specifies the colors for indicating assignments—with the Force display dialog box, for example.

### 14.4.3 On-Line Help



The Help menu's Help topics... menu command displays a dialog box for searching the on-line help by topic. Enter a keyword and click the left mouse button on the Display button.

Clicking the left mouse button on the help button on the tool bar changes the cursor icon. Clicking on a window, dialog box, tool bar icon, etc. then displays the corresponding portion of the help file.

# 15. Appendices

---

## 15.1 DT63K Debugger

### 15.1.1 Menus

DT63K

#### ■ File (F)

- O = Open project...
- S = Save project
- A = Save project as...
- L = Load program file...
- V = Save program to file...
- C = Compare program and file...
- D = Load external memory data file...
- E = Save external memory data to file...
- R = Load RAM data file...
- M = Save RAM data to file...
- X = Exit

#### ■ Edit (E)

- T = Cut
- C = Copy
- P = Paste
- D = Delete
- L = Select all
- F = Find...
- E = Replace...

#### ■ View (V)

- S = Source
- A = Disassembly
- U = Status
- R = Registers
- F = SFRs
- C = Code
- D = Data
- X = External data
- E = Trace memory
- W = Watch list

G = Log  
T = Tool bar  
B = Status bar

■ **Run (R)**

U = Reset CPU  
R = Reset system  
O = Reset options...  
E = Execute program  
G = Execute to cursor  
T = Reset and restart  
S = Step into  
C = Change program counter...  
B = Toggle breakpoint  
K = List breakpoints...  
A = Clear all breakpoints  
D = Break conditions...  
Y = List sync out points  
L = Clear all sync out points  
I = Clear IE bits

■ **CycleCounter (C)**

C = Clear  
H = Change...  
S = Settings...

■ **Trace (T)**

C = Clear trace memory  
P = List trace points...  
A = Clear all trace points  
S = Settings...

■ **Tools (L)**

M = Macro  
P = Stop macro  
S = List symbols...  
E = Load from EPROM...  
C = Compare to EPROM...  
O = Options...  
D = Environment settings

■ **Window (W)**

O = Close active window  
A = Close all  
R = Replicate window  
C = Cascade  
V = Tile vertically  
H = Tile horizontally

## 15. Appendices

### ■ Help (H)

F = Find help on topic

A = About

## 15.1.2 Context Menus

DT63K

### ■ Source Window

G = Execute to cursor

B = Toggle breakpoint

K = List breakpoints...

L = Jump to line...

A = Jump to address...

U = Jump to PC line

D = Disassemble

W = Watch...

M = Move to file...

R = Replicate window

P = Properties...

### ■ Disassembly Window

G = Execute to cursor

B = Toggle breakpoint

K = List breakpoints...

T = Toggle trace point...

L = List trace points...

O = Toggle sync out point

Y = List sync out points...

I = Clear IE bits

A = Jump to address...

U = Jump to PC line

M = Move to source window

W = Watch...

R = Replicate window

P = Properties...

### ■ Status Window

R = Change program counter...

C = Change cycle counter...

T = Clear trace pointer

B = Break details...

P = Properties...

### ■ Register Window

A = Change A...

C = Change C flag...

Z = Change Z flag...

G = Change G flag...

H = Change H register...  
 L = Change L register...  
 N = Change HL index register...  
 X = Change X register...  
 Y = Change Y register...  
 I = Change XY index register...  
 M = Change MIE flag...  
 B = Change CBR...  
 E = Change EBR...  
 0 = Change RA0 register...  
 1 = Change RA1 register...  
 2 = Change RA2 register...  
 3 = Change RA3 register...  
 R = Change RA register...  
 S = Change SP...  
 P = Change RSP...

#### ■ SFR Window

C = Change...  
 F = Find...  
 W = Watch...  
 R = Replicate window

#### ■ Code Window

J = Jump...  
 C = Change...  
 F = Fill...  
 R = Replicate window  
 P = Properties...

#### ■ Data Window

J = Jump...  
 C = Change...  
 F = Fill...  
 O = Copy...  
 L = Load RAM data file...  
 S = Save RAM data to file...  
 R = Replicate window  
 P = Properties...

#### ■ External Data Window

J = Jump...  
 C = Change...  
 F = Fill...  
 O = Copy...  
 R = Replicate window  
 P = Properties...

## 15. Appendices

### ■ Trace Window

J = Jump...  
C = Clear trace memory  
S = Search...  
A = Save trace data...  
R = Replicate window  
P = Properties...











### ■ Watch Window
















I = Add item...  
E = Delete item  
A = Delete all  
P = Item properties...  
U = Up  
D = Down  
L = Load watch list...  
S = Save watch list...  
R = Replicate window

### ■ Log Window

T = Cut  
C = Copy  
P = Paste  
D = Delete  
L = Select all  
E = Clear log  
S = Save log...

### 15.1.3 Tool Bar

		Open project
		Load program file
		Load RAM data file
		Execute program
		Reset and restart
		Step into
		Execute to cursor
		Reset system
		Reset CPU

	Reset options...
	Watch window
	Register window
	Status window
	Code window
	Data window
	External data window
	SFR window
	Cycle counter settings
	Trace options
	Options
	Macro
	Stop macro
	List symbols
	Help topics

### 15.1.4 Shortcut Keys

DT63K	F1	Help
	F2	Macro
	F5	Execute program
	F7	Execute to cursor
	F8	Step into
	F9	Toggle breakpoint
	[Ctrl]+F2	Stop macro
	[Ctrl]+F3	Clear IE bits
	[Ctrl]+F5	Reset CPU
	[Ctrl]+F6	Next window
	[Ctrl]+[Shift]+F6	Previous window
	[Ctrl]+[X]	Cut
	[Ctrl]+[C]	Copy
	[Ctrl]+[V]	Paste



### 15.1.5 Macro Commands

DT63K

#### ■ Projects

PJLOD Load project  
PJSAV Save project

#### ■ Display/Change CPU Internals

D Display register or SFR  
C Change register or SFR  
DPC Display program counter  
CPC Change program counter  
KILL Control TBC or WDT operation

#### ■ Code Memory

DCM Display code memory  
CCM Change code memory  
MCM Copy code memory  
LOD Load program file  
SAV Save program file  
VER Compare code memory to program file  
ASM Assemble string  
STARTASM Start assembling block  
ENDASM End assembly language block  
DASM Disassemble  
EXPAND Expand code memory

#### ■ Data Memory

DDM Display data memory  
CDM Change data memory  
MDM Copy data memory  
DLOD Load RAM data file  
DSAV Save RAM data file

#### ■ External Data Memory

DXM Display external data memory  
CXM Change external data memory  
MXM Copy external data memory  
EXLOD Load external data memory file  
EXSAV Save external data memory file

#### ■ Emulation

G Execute program  
STP Step into

#### ■ Resets

RST Reset system  
RST E Reset CPU  
SRC Select reset source

URST                    Configure user cable RESET pin

#### ■ Emulation Breaks

SBC                    Specify break conditions  
 DBP                    Display breakpoints  
 SBP                    Set breakpoints  
 RBP                    Release breakpoints  
 DBS                    Display break status

#### ■ Tracing

DTM                    Display trace memory  
 STT                    Specify trace triggers  
 STC                    Specify trace conditions  
 DTR                    Display trace point list  
 STR                    Set trace point  
 RTR                    Release trace point  
 DTP                    Display trace pointer  
 RTP                    Reset trace pointer to zero  
 TRSAV                Save trace memory contents to disk file  
 S                      Search trace memory

#### ■ Performance and Coverage

DCC                    Display cycle counter contents  
 CCC                    Change cycle counter contents  
 SCC                    Specify cycle counter count conditions  
 SCLK                   Specify high-speed clock frequency  
 CSTAT                Specify cycle counter display mode  
 SCT                    Specify cycle counter triggers  
 DSO                    Display sync out point list  
 SSO                    Set sync out point  
 RSO                    Release sync out point  
 DIE                    Display IE bits  
 RIE                    Reset IE bits

#### ■ EPROM

TPR                    Load program from EPROM  
 VPR                    Compare program to EPROM

#### ■ Macros

PAUSE                Pause macro execution  
 ECHO                Send specified string to log  
 GOTO                Branch to label  
 IF                    Conditional branch

#### ■ Symbols

DSYM                Display symbol value  
 CSYM                Change symbol value  
 RSYM                Release symbol

## 15. Appendices

SSYM                      Set symbol

### ■ Miscellaneous

VERSION                  Display DT63K version number

EXIT                      Exit debugger

SAVELOG                  Save macro results to disk file

CLRLOG                   Clear Log window

## 15.2 DTS63K Simulator

### 15.2.1 Menus

DTS63K

#### ■ File (F)

O = Open project...  
 S = Save project  
 A = Save project as...  
 L = Load program file...  
 V = Save program to file...  
 C = Compare program and file...  
 D = Load external memory data file...  
 E = Save external memory data to file...  
 R = Load RAM data file...  
 M = Save RAM data to file...  
 G = Load LCD graphics...  
 P = Load port assignment file...  
 I = Save port assignments to file  
 T = Save port assignments as...  
 B = Load breakpoint file...  
 K = Save breakpoints to file...  
 X = Exit

#### ■ Edit (E)

T = Cut  
 C = Copy  
 P = Paste  
 D = Delete  
 L = Select all  
 F = Find...  
 E = Replace...

#### ■ View (V)

S = Source  
 A = Disassembly  
 U = Status  
 R = Registers  
 F = SFRs  
 C = Code  
 D = Data  
 X = External data  
 E = Trace memory  
 W = Watch list  
 G = Log

## 15. Appendices

L = LCD  
P = Port  
T = Tool bar  
B = Status bar

### ■ Run (R)

R = Reset simulator  
O = Reset options...  
X = Stop execution  
M = Stop melody  
E = Execute program  
G = Execute to cursor  
T = Reset and restart  
S = Step into  
O = Step over  
C = Change program counter...  
B = Toggle breakpoint  
K = List breakpoints...  
A = Clear all breakpoints  
D = Break conditions...  
I = Clear IE bits

### ■ CycleCounter (C)

C = Clear  
H = Change...  
S = Settings...

### ■ Trace (T)

C = Clear trace memory  
T = Save trace data  
S = Settings...

### ■ Tools (L)

M = Macro  
P = Stop macro  
S = List symbols...  
L = LCDTOOL  
O = Options...  
D = Environment settings

### ■ Window (W)

O = Close active window  
A = Close all  
R = Replicate window  
C = Cascade  
V = Tile vertically  
H = Tile horizontally

■ **Help (H)**

F = Find help on topic

A = About

## 15.2.2 Context Menus

DTS63K

■ **Source Window**

G = Execute to cursor

B = Toggle breakpoint

K = List breakpoints...

L = Jump to line...

A = Jump to address...

U = Jump to PC line

D = Disassemble

W = Watch...

M = Move to file...

R = Replicate window

P = Properties...

■ **Disassembly Window**

G = Execute to cursor

B = Toggle breakpoint

K = List breakpoints...

I = Clear IE bits

C = Change program counter...

L = Assemble...

A = Jump to address...

U = Jump to PC line

M = Move to source window

W = Watch...

R = Replicate window

P = Properties...

■ **Status Window**

R = Change program counter...

C = Change cycle counter...

T = Clear trace pointer

B = Break details...

P = Properties...

■ **Register Window**

A = Change A...

C = Change C flag...

Z = Change Z flag...

G = Change G flag...

H = Change H register...

L = Change L register...

## 15. Appendices

N = Change HL index register...  
X = Change X register...  
Y = Change Y register...  
I = Change XY index register...  
M = Change MIE flag...  
B = Change CBR...  
E = Change EBR...  
0 = Change RA0 register...  
1 = Change RA1 register...  
2 = Change RA2 register...  
3 = Change RA3 register...  
R = Change RA register...  
S = Change SP...  
P = Change RSP...

### ■ SFR Window

C = Change...  
F = Find...  
W = Watch...  
R = Replicate window

### ■ Code Window

J = Jump...  
C = Change...  
F = Fill...  
R = Replicate window  
P = Properties...

### ■ Data Window

J = Jump...  
C = Change...  
F = Fill...  
O = Copy...  
R = Replicate window  
P = Properties...

### ■ External Data Window

J = Jump...  
C = Change...  
F = Fill...  
O = Copy...  
R = Replicate window  
P = Properties...

### ■ Trace Window

J = Jump...  
C = Clear trace memory  
S = Search...

R = Replicate window

P = Properties...

#### ■ Watch Window

I = Add item...

E = Delete item

A = Delete all

P = Item properties...

U = Up

D = Down

L = Load watch list...

S = Save watch list...

R = Replicate window

#### ■ Log Window

T = Cut

C = Copy

P = Paste

D = Delete

L = Select all

E = Clear log

S = Save log...

#### ■ LCD Window

G = Load LCD data file...

L = LCDTOOL

#### ■ Port Window (Port or Key Map)

C = Change key assignments...

L = Load port assignment file...

S = Save port assignments to file

A = Save port assignments as...

R = Replicate window

#### ■ Port Window (Matrix)

I = Add input port...

O = Add output port...

D = Delete port

L = Load port assignment file...

S = Save port assignments to file

A = Save port assignments as...

R = Replicate window

### 15.2.3 Tool Bar



























Open project



Load program file



## 15. Appendices

	Load LCD data file
	Load port assignment file
	Load RAM data file
	Execute program
	Reset and restart
	Step into
	Step over
	Execute to cursor
	Stop execution
	Reset simulator
	Reset options...
	Watch window
	Register window
	Status window
	Code window
	Data window
	External data window
	SFR window
	List symbols
	Options
	LCDTOOL
	Macro
	Stop macro
	Help topics

## 15.2.4 Shortcut Keys

DTS63K		
	F1	Help
	F2	Macro
	F5	Execute program
	F7	Execute to cursor
	F8	Step into
	F9	Toggle breakpoint
	[Ctrl]+F2	Stop macro
	[Ctrl]+F3	Clear IE bits
	[Ctrl]+F5	Reset CPU
	[Ctrl]+F6	Next window
	[Ctrl]+F8	Step over
	[Ctrl]+[Shift]+F6	Previous window
	[Ctrl]+[X]	Cut
	[Ctrl]+[C]	Copy
	[Ctrl]+[V]	Paste
	[ESC]	Stop program

## 15.2.5 Macro Commands

DTS63K		
	<b>■ Projects</b>	
	PJLOD	Load project
	PJSAV	Save project
	<b>■ Display/Change CPU Internals</b>	
	D	Display register or SFR
	C	Change register or SFR
	DPC	Display program counter
	CPC	Change program counter
	MELODY	Enable/disable melody function
	<b>■ Code Memory</b>	
	DCM	Display code memory
	CCM	Change code memory
	MCM	Copy code memory
	LOD	Load program file
	SAV	Save program file
	VER	Compare code memory to program file
	ASM	Assemble string
	STARTASM	Start assembling block
	ENDASM	End assembly language block
	DASM	Disassemble
	EXPAND	Expand code memory
	<b>■ Data Memory</b>	
	DDM	Display data memory
	CDM	Change data memory

## 15. Appendices

MDM	Copy data memory
DLOD	Load RAM data file
DSAV	Save RAM data file

### ■ External Data Memory

DXM	Display external data memory
CXM	Change external data memory
MXM	Copy external data memory
EXLOD	Load external data memory file
EXSAV	Save external data memory file

### ■ Simulation

G	Execute program
STP	Step into
STPO	Step over

### ■ Resets

RST	Reset simulator
SRC	Select reset source

### ■ Simulation Breaks

SBC	Specify break conditions
DBP	Display breakpoints
SBP	Set breakpoints
RBP	Release breakpoints
DBS	Display break status
BLOD	Load breakpoint file
BSAV	Save breakpoint file

### ■ Tracing

DTM	Display trace memory
STC	Specify trace conditions
DTP	Display trace pointer
RTP	Reset trace pointer to zero
TRSAV	Save trace memory contents to disk file
S	Search trace memory

### ■ Performance and Coverage

DCC	Display cycle counter contents
CCC	Change cycle counter contents
SCC	Specify cycle counter count conditions
SCLK	Specify high-speed clock frequency
CSTAT	Specify cycle counter display mode
DIE	Display IE bits
RIE	Reset IE bits
WAIT	Specify wait between instructions

### ■ Macros

PAUSE	Pause macro execution
-------	-----------------------

ECHO	Send specified string to log
GOTO	Branch to label
IF	Conditional branch

### ■ Symbols

DSYM	Display symbol value
CSYM	Change symbol value
RSYM	Release symbol
SSYM	Set symbol

### ■ Miscellaneous

LLOD	Load LCD data file
PLOD	Load port assignment file
PSAV	Save port assignment file
VERSION	Display DTS63K version number
EXIT	Exit simulator
SAVELOG	Save macro results to disk file
CLRLOG	Clear Log window

## 15.3 LCDTOOL Utility

### 15.3.1 Menu

LCDTOOL

■ **File (F)**

N = New  
O = Open...  
S = Save  
A = Save as...  
E = Export...  
X = Exit

■ **Edit (E)**

U = Undo  
T = Cut  
C = Copy  
P = Paste  
D = Delete  
A = Cancel assignment  
G = Group  
R = Release group

■ **View (V)**

Z = Zoom...  
T = Tool bar  
S = Status bar  
A = Segment assignment tool  
C = Force display tool

■ **Tools (T)**

D = Dot matrix...  
S = Settings...  
C = Common/segment numbers

■ **Help (H)**

H = Help topics...  
A = About

### 15.3.2 Context Menu

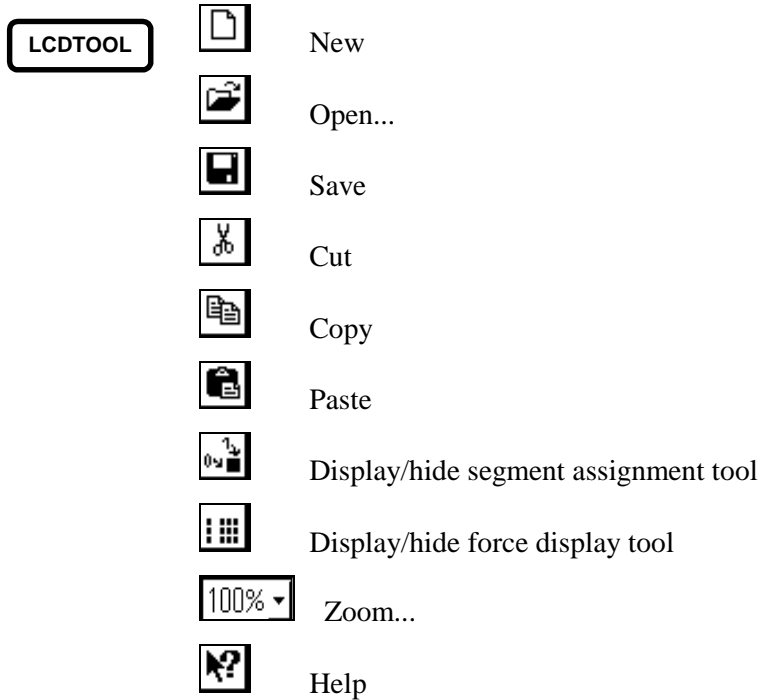
LCDTOOL

T = Cut  
C = Copy  
P = Paste  
D = Delete  
A = Cancel assignment

G = Group

R = Ungroup

### 15.3.3 Tool Bar



### 15.3.4 Shortcut Keys

Shortcut Key	Function
[Ctrl]+[N]	New
[Ctrl]+[O]	Open...
[Ctrl]+[S]	Save
[Ctrl]+[Z]	Undo
[Ctrl]+[X]	Cut
[Ctrl]+[C]	Copy
[Ctrl]+[V]	Paste
[Del]	Delete

## 15.4 Script Command Reference

### 15.4.1 Projects

DT63K  
DTS63K

#### ■ Load Project

PJLOD *filename*

*filename*: Name of project file to load

DT63K  
DTS63K

#### ■ Save Project

PJSAV *filename*

*filename*: Name for project file

### 15.4.2 Display/Change CPU Internals

DT63K  
DTS63K

#### ■ Display Register or SFR

D *parm*

*parm*: *Reg\_name* or *SFR\_name* from C command

DT63K  
DTS63K

#### ■ Change Register or SFR

C *Reg\_name* = *data*

<i>Reg_name</i> :	ACC	Accumulator
	C	Carry (C) flag
	Z	Zero (Z) flag
	G	G flag
	H	H register
	L	L register
	HL	HL register (byte)
	X	X register
	Y	Y register
	XY	XY register (byte)
	CBR	Current bank register
	EBR	Extra bank register
	RA0–RA3	RA register
	RA	RA3–RA0 (RA registers (word))
	RSP	Register stack pointer

*data*: New contents

DT63K  
DTS63K■ **Change SFR**C *Sfr\_name* = *data**Sfr\_name*: Name of register assigned to SFR region*data*: New contentsDT63K  
DTS63K■ **Display Program Counter**

DPC

DT63K  
DTS63K■ **Change Program Counter**CPC *address**address*: New code address for program counter

DTS63K

■ **Enable/Disable Melody Function**

MELODY

MELODY *parm*

<i>parm</i> :	ON	Enable melody function
	OFF	Disable melody function

DT63K

■ **Control TBC or WDT Operation**

KILL

KILL *parm*

<i>parm</i> :	TBC	Enable time base counter
	WDT	Enable watchdog timer
	*	Enable both time base counter and watchdog timer
	~TBC	Disable time base counter
	~WDT	Disable watchdog timer
	~*	Disable both time base counter and watchdog timer

## 15.4.3 Code Memory

DT63K  
DTS63K■ **Display Code Memory**DCM *address**address*: Code address to displayDCM [*start-address end-address*]*start-address*: First code address to display



## 15. Appendices

*end-address*: Last code address to display

DT63K  
DTS63K

### ■ Change Code Memory

CCM *address = data*

*address*: Code address to modify

*data*: New contents

CCM [*start-address end-address*] = *data*

*start-address*: First code address to modify

*end-address*: Last code address to modify

*data*: New contents

DT63K  
DTS63K

### ■ Copy Code Memory

MCM [*start-address end-address*] *trans-address*

*start-address*: First code address to copy

*end-address*: Last code address to copy

*trans-address*: Destination code address

DT63K  
DTS63K

### ■ Load Program File

LOD *filename* {*option ... option*}

*filename*: Name of program file to load

*option*: /NS Skip symbol table  
/B Clear all breakpoints

DT63K  
DTS63K

### ■ Save Program File

SAV *filename*{*[start-address end-address]*}{*option ... option*}

*filename*: Name for program file

*start-address*: First code address to save

*end-address*: Last code address to save

*option*: /NS Skip symbol table

DT63K  
DTS63K

### ■ Compare Code Memory to Program File

VER *filename*

VER *filename* [*start-address end-address*]

*filename*: Name of program file to compare

*start-address*: First code address to compare

*end-address*: Last code address to compare

DT63K  
DTS63K

#### ■ Assemble String

ASM *address* “*assemble-string*”

ASM “*assemble-string*”

*address*: Code memory address

“*assemble-string*” Assembly language instruction to assemble

**Note:** The second form is for use inside a block starting with STARTASM and ending with ENDASM.

DT63K  
DTS63K

#### ■ Start Assembling Block

STARTASM *address*

*address*: Code memory address

DT63K  
DTS63K

#### ■ End Assembly Language Block

ENDASM

DT63K  
DTS63K

#### ■ Disassemble

DASM *address* {*option ... option*}

DASM [*start-address end-address*]{*option ... option*}

*address*: Code memory address

*start-address*: First code address to disassemble

*end-address*: Last code address to disassemble

*option*: /NC Skip instruction codes  
/NL Skip addresses

DT63K  
DTS63K

#### ■ Expand Code Memory

EXPAND

EXPAND *parm*

## 15. Appendices

<i>parm:</i>	ON	Enable code memory expansion
	OFF	Disable code memory expansion

### 15.4.4 Data Memory

DT63K  
DTS63K

#### ■ Display Data Memory

DDM *address*

DDM [*start-address end-address*]

*address:* Data address to display

*start-address:* First data address to display

*end-address:* Last data address to display

DT63K  
DTS63K

#### ■ Change Data Memory

CDM *address = data*

CDM [*start-address end-address*] = *data*

*address:* Data address to modify

*start-address:* First data address to modify

*end-address:* Last data address to modify

*data:* New contents

DT63K  
DTS63K

#### ■ Copy Data Memory

MDM [*start-address end-address*] *trans-address*

*start-address:* First data address to copy

*end-address:* Last data address to copy

*trans-address:* Destination data address

DT63K  
DTS63K

#### ■ Load RAM Data File

DLOD *filename*

*filename:* Name of RAM data file to load

DT63K  
DTS63K

#### ■ Save RAM Data File

DSAV *filename*{*[start-address end-address]*}

*filename*: Name for RAM data file  
*start-address*: First data address to save  
*end-address*: Last data address to save

**Note:** Addresses in SFR area cannot be specified for start- and end-addresses.

### 15.4.5 External Data Memory

DT63K  
DTS63K

#### ■ Display Current Bank in External Data Memory

DXM *address*

DXM [*start-address end-address*]

*address*: External data address to display  
*start-address*: First external data address to display  
*end-address*: Last external data address to display

DT63K  
DTS63K

#### ■ Change Current Bank in External Data Memory

CXM *address = external\_data*

CXM [*start-address end-address*] = *external\_data*

*address*: External data address to modify  
*start-address*: First external data address to modify  
*end-address*: Last external data address to modify  
*external\_data*: New contents

DT63K  
DTS63K

#### ■ Copy External Data Memory

MXM [*start-address end-address*] *trans-address*

*start-address*: First external data address to copy  
*end-address*: Last external data address to copy  
*trans-address*: Destination external data address

DT63K  
DTS63K

#### ■ Load External Data File

EXLOD *filename*

*filename*: Name of external data file to load

DT63K  
DTS63K

#### ■ Save External Data File

## 15. Appendices

EXSAV *filename*

EXSAV *filename* [*start-address end-address*]

*filename*: Name for external data file

*start-address*: First external data address to save

*end-address*: Last external data address to save

### 15.4.6 Execution

DT63K  
DTS63K

#### ■ Execute from Current PC Position without Breakpoint

G

DT63K  
DTS63K

#### ■ Execute from Specified Address without Breakpoint

G *address*

*address*: Starting code memory address

DT63K  
DTS63K

#### ■ Execute with Single Breakpoint

G {*address*} , *break-address*

*address*: Starting code memory address

*break-address*: Breakpoint address

DT63K  
DTS63K

#### ■ Execute with Address Pass Count Break

G {*address*} , *break-address* [*count*]

*address*: Starting code memory address

*break-address*: Breakpoint address

*count*: Number of matches before break

DT63K  
DTS63K

#### ■ Execute with RAM Data Match Break

G {*address*} , RAM {*ram-address*}{AND *mask\_data*} =  
*data*{*count*}{*break\_address*}

*address*: Starting code memory address

*ram-address*: RAM address to monitor

*mask\_data*: Mask for comparison

*data*: Data for comparison  
*count*: Number of matches before break  
*break\_address*: *address* Single address  
 [*start-address end-address*] Address range

DT63K  
DTS63K

#### ■ Execute with RAM Address Access Break

G {*address*} , RADR {AND *mask\_data*} = *ram-address* {[*count*]}

*address*: Starting code memory address  
*mask\_data*: Mask for comparison  
*ram-address*: RAM address to monitor  
*count*: Number of matches before break

DT63K

#### ■ Execute with Internal ROM Table Data Match Break

G {*address*} , TBL {*tbl-address*}{AND *mask\_data*} = *data* {*break\_address*}

*address*: Starting code memory address  
*tbl-address*: Internal ROM table address to monitor  
*mask\_data*: Mask for comparison  
*data*: Data for comparison  
*break\_address*: *address* Single address  
 [*start-address end-address*] Address range

DTS3K

#### ■ Execute with Internal ROM Table Address Access Break

G {*address*} , TADR {AND *mask\_data*} = *tbl-address*

*address*: Starting code memory address  
*mask\_data*: Mask for comparison  
*tbl-address*: Internal ROM table address to monitor

DT63K

#### ■ Execute with External Data Memory Data Match Break

G {*address*} , EXM *access* {*ext-address*}{AND *mask\_data*} = *data*{*break\_address*}

*address*: Starting code memory address  
*access*: R Read  
 W Write

## 15. Appendices

	RW	Read or write
<i>ext-address:</i>	External data memory address to monitor	
<i>mask_data:</i>	Mask for comparison	
<i>data:</i>	Data for comparison	
<i>break_address:</i>	<i>address</i>	Single address
	[ <i>start-address end-address</i> ]	Address range

DT63K

### ■ Execute with External Data Memory Address Access Break

G {*address*} , EADR {AND *mask\_data*} = *ext-address*

<i>address:</i>	Starting code memory address
<i>mask_data:</i>	Mask for comparison
<i>ext-address:</i>	External data memory address to monitor

DT63K  
DTS63K

### ■ Step Into

STP {*address*}{ , *count*}

<i>address:</i>	Starting code memory address
<i>count:</i>	Number of steps to execute

DTS63K

### ■ Step Over

STPO {*address*}{ , *count*}

<i>address:</i>	Starting code memory address
<i>count:</i>	Number of steps to execute

## 15.4.7 Resets

DT63K  
DTS63K

### ■ Reset Emulator or Simulator

RST

DT63K

### ■ Reset CPU

RST E

DT63K  
DTS63K

### ■ Select Reset Option

SRC

SRC *parm* Enable reset option  
 SRC *~parm* Disable reset option  
*parm:* CC Cycle counter  
 IE All IE bits  
 TP Trace pointer

DT63K

### ■ Configure User Cable RESET Pin

URST

URST *parm*

*parm:* ON Enable user cable RESET input  
 OFF Disable user cable RESET input

## 15.4.8 Breaks

DT63K

### ■ Specify Break Conditions

SBC

SBC *parm* Enable break conditionSBC *~parm* Disable break condition

*parm:* BP Breakpoint break  
 CC Cycle counter overflow break  
 TF Trace full break  
 XP External break  
 PD Power down break  
 PC Call stack overflow break  
 RG Register stack overflow break  
 \* All

DTS63K

### ■ Specify Break Conditions

SBC

SBC *parm* Enable break conditionSBC *~parm* Disable break condition

*parm:* BP Breakpoint break  
 CC Cycle counter overflow break  
 TF Trace full break  
 PC Call stack overflow break  
 RG Register stack overflow break



## 15. Appendices

DT63K  
DTS63K

### ■ Display Breakpoints

DBP

DT63K  
DTS63K

### ■ Set Breakpoint

SBP *address*

*address:* Code memory address

DT63K  
DTS63K

### ■ Release Breakpoints

RBP *bp-address*

*bp-address:* *address* Breakpoint address  
\* All addresses

DT63K  
DTS63K

### ■ Display Break Status

DBS

DTS63K

### ■ Load Breakpoint File

BLOD *filename*

*filename:* Name of breakpoint data file to load

DTS63K

### ■ Save Breakpoint File

BSAV *filename*

*filename:* Name for breakpoint data file

## 15.4.9 Tracing

DT63K  
DTS63K

### ■ Display Trace Memory

DTM *TP\_number steps*

*TP\_number:* Starting entry number

*steps:* Number of entries to display

DT63K  
DTS63K

### ■ Display Trace Memory Relative to Current Position

DTM BACK *steps1 steps2*

*steps1*: Number of entries to scroll back from current trace pointer position

*steps2*: Number of entries to display from that entry

DT63K

■ Specify Trace Triggers

STT

STT *parm*

*parm*: ALL Free running  
DIS Disable tracing

STT SS {*parm*}{, *parm*}

*parm*: *address* Single address  
[*start-address end-address*] Address range  
.(period) Start or end of execution

DTS63K

■ Specify Trace Conditions

STC

STC TR Enable tracing

STC ~TR Disable tracing

DT63K

■ Display Trace Point List

DTR

DT63K

■ Set Trace Point

STR *address* Single address

STR [*start-address end-address*] Address range

DT63K

■ Release Trace Point

RTR *tp-address*

*tp-address*: *address* Single address  
[*start-address end-address*] Address range  
\* All addresses

DT63K  
DTS63K

■ Display Trace Pointer

DTP

## 15. Appendices

DT63K  
DTS63K

### ■ Reset Trace Pointer to Zero

RTP

DT63K  
DTS63K

### ■ Save Trace Memory Contents to Disk File

TRSAV *filename*

TRSAV *filename* [*start-TP end-TP*]

*filename*: Name for trace data file

*start-TP*: First trace entry to save

*end-TP*: Last trace entry to save

DT63K  
DTS63K

### ■ Search Trace Memory

S *col* {AND *mask\_data*} = *data*{*count*} Search forward

S ~*col* {AND *mask\_data*} = *data*{*count*} Search backward

*col*:

ADRS	Instruction address
ACC	Accumulator
RADR	RAM address
RD	RAM data
C	Carry (C) flag
Z	Zero (Z) flag
G	G flag
MI	Master interrupt enable (MIE) flag
MD	Melody request flag
CB	Current bank register
EB	Extra bank register
HL	HL register (byte)
XY	XY register (byte)
SP	Stack pointer
RS	Register stack pointer

*mask\_data*: Mask for comparison

*data*: Data for comparison

*count*: Number of matches before break

## 15.4.10 Performance and Coverage

DT63K  
DTS63K

### ■ Display Cycle Counter Contents

DCC

DT63K  
DTS63K

## ■ Change Cycle Counter Contents

CCC *data**data:*                   New contents

DTS63K

## ■ Specify Cycle Counter Count Conditions

SCC

SCC CC {*option*}   Enable counting

SCC ~CC                   Disable counting

*option:*                   /NS   Skip symbol table

DT63K

## ■ Specify Cycle Counter Triggers

SCT *parm**parm:*   ALL                   Free running

DIS                   Disable counting

SCT SS {*parm*}{*, parm*}*parm:*                   *address*                   Single address          [*start-address end-address*] Address range

. (period)           Start or end of execution

DT63K  
DTS63K

## ■ Specify High-Speed Clock Frequency

SCLK

SCLK *value unit**value:*                   Decimal value*unit:*                   Hz, kHz, or MHzDT63K  
DTS63K

## ■ Specify Cycle Counter Display Mode

CSTAT *mode**mode:*                   CC   Count

TM   Time

DT63K

## ■ Display Sync Out Point List

DSO



*end-address*: Last EPROM address to compare

*eprom-address*: Destination EPROM address

## 15.4.12 Macros

DT63K  
DTS63K

### ■ Pause Macro Execution

PAUSE

DT63K  
DTS63K

### ■ Send Specified String to Log

ECHO *statement*

*statement*: *statement parm*

*parm*: Text in double quotes  
\$TIME Current time  
\$DATE Current date

DT63K  
DTS63K

### ■ Branch to Label

GOTO *label*

*:label* Label for destination

DT63K  
DTS63K

### ■ Conditional Branch

IF *compare statement*

*compare*: *value-comp*  
*lastline-comp*

*value-comp*: *obj cond value*

*obj*: ACC, C, Z, G, CBR, EBR, MIE, RSP, SP, HL, XY, RA  
PC, SFR name, *data-address*

*cond*: == or !=

*value*: Expression

*data-address*: Expression evaluating to a data address

*lastline-comp*: LASTLINE *cond* “*string*”  
Compare to last line

LASTLINE(*n*) *cond* “*string*”

Compare to line *n* lines back

*statement*: THEN *command*

## 15. Appendices

GOTO *label*  
*command:* Script command

DT63K  
DTS63K

### ■ Save Macro Results to Disk File

SAVELOG *filename*  
*filename:* Name for log file

DT63K  
DTS63K

### ■ Clear Log window

CLRLOG

## 15.4.13 Symbols

DT63K  
DTS63K

### ■ Display Symbol Value

DSYM *symbol*  
*symbol:* Symbol name (or mask containing wild cards)

DT63K  
DTS63K

### ■ Change Symbol Value

CSYM *symbol = data*  
*symbol:* Symbol name  
*data:* New contents

DT63K  
DTS63K

### ■ Release Symbol

RSYM *symbol*  
*symbol:* Name or user-defined symbol (or mask containing wild cards)

DT63K  
DTS63K

### ■ Set Symbol

SSYM *symbol = data {attribute}*  
*symbol:* Symbol name

<i>data:</i>	Contents
<i>attribute:</i>	Symbol attribute
	/C Code
	/D Data
	/X External data
	/B Bit data
	/N Numerical value

### 15.4.14 Miscellaneous

DTS63K

#### ■ Load LCD Data File

LLOD *filename*

*filename*: Name of LCD data file to load

DTS63K

#### ■ Load Port Assignment File

PLOD *filename*

*filename*: Name of port assignment file to load

DTS63K

#### ■ Save Port Assignment File

PSAV *filename*

*filename*: Name for port assignment file

DT63K  
DTS63K

#### ■ Display Version Number

VERSION

DT63K  
DTS63K

#### ■ Exit Debugger

EXIT



## Index

***I***

100Hz Timer/Counter .....69

***3***

32-bit applications .....1

***A***

A .....49

Accumulator .....41

Address Pass Count(DT63K) .....33

Address Pass Count(DTS63K) .....38

Arithmetic Operators .....62

ASCII format .....8

ASM Command .....107

Assigning Ports .....15

***B***

Backing Up the Software .....10

Backup Circuit .....72

Batch processing .....2

Battery Low Detector .....72

Binary format .....8

Bitmap

    Copying .....81

    Deleting .....81

    Grouping .....81

    Moving .....81

    Undoing .....81

Bit-mapped file .....15

BLOD Command .....114

Break Condition Check Boxes(DT63K) .....32

Break Condition Check Boxes(DTS63K) .....37

Break Conditions(DT63K) .....31

Break Conditions(DTS63K) .....37

Break Parameters(DT63K) .....33

Break Parameters(DTS63K) .....38

Breakpoint(DT63K) .....32

Breakpoint(DTS63K) .....37

Breakpoints .....29

BSAV Command .....114

Buzzer .....3

Buzzer Driver .....71

Byte .....5

***C***

C Command .....104

Call Stack Overflow(DT63K) .....32

Call Stack Overflow(DTS63K) .....37

Carry Flag .....41

CB .....49

CCC Command .....117

CCM Command .....106

CDM Command .....108

CGZ .....49

Change key assignments .....70

Character Constants .....61

Clearing Breakpoints .....22

Clock operating frequencies .....6

CLRLOG Command .....120

Code Memory .....43

Code Memory Assembler .....44

Code Memory Expansion .....14, 20

COM .....77

Common .....76

Comparing Code Memory to File .....24

Comparing Code to EPROM .....66

Comparing Complete Program .....24

Constants .....61

Context Menu(LCDTOOL) .....102

Context Menus(DT63K) .....86

Context Menus(DTS63K) .....95

Continuous Execution .....26

Control Panels .....10

CPC Command .....105

CPU Reset .....28

CSTAT Command .....117

CSYM Command .....120

Current Bank Register .....41

Cursor .....31

CXM Command .....109

Cycle counter .....2

Cycle Counter .....54

    Changing .....54

Displaying .....	54	DXM Command.....	109
Options .....	55	<b>E</b>	
Overview .....	54	E .....	50
Cycle Counter Overflow .....	54	EB .....	49
Cycle counter overflow(DT63K) .....	32	ECHO Command .....	119
Cycle counter overflow(DTS63K) .....	37	ENDASM Command .....	107
Cycle Counter(DT63K) .....	54	Environment Settings .....	13
Cycle Counter(DTS63K) .....	54	EPROM.....	66
<b>D</b>		EXIT Command.....	121
D Command .....	104	EXLOD Command.....	109
D63 .....	25	EXPAND Command.....	107
DASM Command .....	107	Expressions .....	60
Data Memory .....	43	EXSAV Command.....	110
DBP Command .....	114	External Data File .....	9
DBS Command .....	114	External Data Memory .....	20, 71
DCC Command.....	116	External Memory .....	43
DCM Command.....	105	External Memory Address Access(DT63K) ...	36
DDM Command.....	108	External Memory Data Match(DT63K) .....	35
Development language.....	6	External(DT63K) .....	32
Development process .....	8	Extra Bank Register .....	41
DIE Command .....	118	<b>F</b>	
Directories settings .....	13	Filling Memory .....	44
Display .....	6	Floppy disk drive .....	6
Displaying Key Assignments .....	17	Forced Break.....	31
Displaying Key Matrix.....	16	<b>G</b>	
Displaying Port Assignments .....	15	G Command .....	110
DL0D Command.....	108	G Flag .....	41
Dot matrix .....	75	GOTO Command .....	119
Dot Operator .....	62	<b>H</b>	
DPC Command .....	105	Hard disk space .....	6
DSAV Command .....	108	High-Speed Clock .....	20
DSO Command .....	117	HL .....	50
DSPR .....	26, 77	HL Register .....	41
DSYM Command .....	120	<b>I</b>	
DT63K Debugger Features .....	2	I .....	50
DT63K Project File .....	25	I/O port simulation .....	15
DTM Command .....	114	IE Bits .....	27
DTP Command .....	115	IF Command .....	119
DTR Command .....	115	In-circuit emulator.....	6
DTS63K Project File .....	25		
DTS63K Simulator Features.....	2		
Dumping Memory.....	43		

## Index

Input Formats .....	59	LCDTOOL utility .....	15
Installing.....	10	LCDTOOL Utility .....	74
Instruction Set .....	68	Limiting the Range .....	23, 24
Integer Constants.....	61	Listing Breakpoints .....	30
Intel HEX format.....	8	LLOD Command.....	121
Intel HEX Half-Word Files .....	8	Loading Code from EPROM.....	66
Internal ROM Table Address Access(DT63K).....	35	Loading Debugging Information .....	22
Internal ROM Table Data Match(DT63K).....	34	Loading DT63K Debugger.....	11
<b>J</b>			
Jumping to an Address .....	44	Loading DTS63K Simulator .....	12
<b>K</b>			
Key matrix.....	70	Loading LCD Data File .....	15
Keymap button .....	17	Loading Only Program Code.....	22
KILL Command .....	105	Loading Program Code .....	22
<b>L</b>			
LCD Data File .....	9	LOC.....	49
Saving .....	79	LOD Command .....	106
LCD Data Files		Log Window.....	64
Editing .....	81	<b>M</b>	
LCD Driver .....	71	Macro	
LCD panel simulation .....	2	Breaks .....	113
LCD pixels .....	21	Code Memory .....	105
LCD Refresh Period Setting.....	20	Data Memory .....	108
LCD Screen Refresh Period .....	73	Display/Change CPU Internals .....	104
LCD Settings.....	21	Execution .....	110
LCD window .....	73	External Data Memory.....	109
LCDTOOL .....	73	Macros .....	119
Assigning by DSPR Value .....	78	Miscellaneous .....	121
Assigning Common and Segment Outputs..	78	Performance and Coverage .....	116
Assigning Common Outputs .....	77	Projects .....	104
Assigning Segment Outputs .....	78	Resets.....	112
Canceling Assignments .....	81	Script Command Reference .....	104
Creating .BMP File.....	74	Tracing.....	114
Creating Dot Matrix.....	75	Macro Commands(DT63K) .....	90
Displaying Assignments.....	79	Macro Commands(DTS63K) .....	99
Loading .BMP Data .....	75	Macro functions.....	2
Making Assignments.....	77	Macros.....	63
On-Line Help.....	83	Master Interrupt Enable Flag.....	41
Overview.....	74	Matrix button.....	16
Procedure.....	74	MCM Command.....	106
		MD .....	49
		MDM Command .....	108
		MELODY Command .....	105
		Melody Driver .....	71
		Melody Output .....	20

Melody simulation .....	3
Memory Spaces.....	68
Menus(DT63K).....	84
Menus(DTS63K) .....	93
Menus(LCDTOOL) .....	102
MI .....	49
Mouse .....	6
Multi window interface .....	1
Multiplier/Divider Circuit.....	72
MXM Command.....	109

**N**

New button.....	19
Nibble .....	5
None(DT63K).....	33
None(DTS63K) .....	38
Notational Conventions .....	5
nX-4/250 core .....	69

**O**

OLMS-63K series .....	1
On-Line Help .....	64
Contents.....	64
keywords .....	65
On-screen help .....	1
Operating environment .....	6
Operating Frequency.....	67
Operating System.....	6
Operators .....	62

**P**

PAUSE Command .....	119
PC .....	49
PJLOD Command.....	104
PJSAV Command .....	104
PLOD Command .....	121
Port button .....	15
Port simulation.....	2
Port window.....	15
Ports.....	70
Power Down(DT63K).....	32
Preparing for Debugging.....	13
Preparing for Emulation.....	13

Preparing for Simulation.....	15
Probe cable.....	7
Program Counter.....	40
Program Execution.....	26
Project File.....	25
Project settings.....	13
PSAV Command.....	121

**Q**

Quick on-screen help .....	1
----------------------------	---

**R**

RA Registers .....	41
RADR .....	49
RAM Address Access(DT63K) .....	34
RAM Address Access(DTS63K) .....	39
RAM Data Match(DT63K).....	33
RAM Data Match(DTS63K).....	38
RBP Command .....	114
RD.....	49
Real-time emulation .....	2
Register Set .....	68
Register Stack Overflow(DT63K) .....	32
Register Stack Overflow(DTS63K) .....	37
Register Stack Pointer.....	42
Register window.....	40
Registers.....	40
Resets .....	28
RIE Command.....	118
RS .....	50
RSO Command .....	118
RST Command.....	112
RST E Command .....	112
RSYM Command.....	120
RTP Command.....	116
RTR Command .....	115

**S**

S Command.....	116
S63 .....	25
SASM63K structured assembler .....	1, 8
SAV Command.....	106
SAVELOG Command.....	120

## Index

Saving Program Code .....	23	Executing to Cursor .....	58
Saving Program Code Only.....	23	Switching from Trace Window .....	58
Saving Symbol Table .....	23	Switching to Disassembly Window.....	58
SBC Command.....	113	SP .....	50
SBP Command.....	114	Specifying Emulator Options .....	13
SCC Command.....	117	Specifying Reset Scope .....	28
SCLK Command.....	117	Specifying Simulator Options .....	19
Scripting language.....	2	Specifying Target Device .....	13
SCT Command.....	117	SRC Command.....	112
<b>SEG</b> .....	77	SSO Command.....	118
Segment.....	76	SSYM Command .....	120
Serial port.....	6	Stack Pointer .....	42
Serial Ports.....	71	Start Menu.....	11
Setting Breakpoints .....	29	STARTASM Command .....	107
Setting Up a Key Matrix .....	18	Status line .....	1
SFR .....	42	Status window .....	40
Adding to Watch List.....	42	STC Command.....	115
Searching .....	42	Step Over Execution.....	26
SFR window.....	42	Stop melody playback .....	27
Shift Registers .....	71	STP Command .....	112
Shortcut Keys(DT63K) .....	89	STPO Command .....	112
Shortcut Keys(DTS63K).....	99	STR Command.....	115
Shortcut Keys(LCDTOOL).....	103	Structure of This Manual.....	3
Simulation		STT Command.....	115
Instruction Execution.....	67	Suspending Breakpoints.....	30
Interrupt .....	72	Suspending Program Execution.....	27
LCD .....	72	Symbol	
Onboard Peripherals .....	69	Attributes .....	59
Reset .....	68	BIT.....	60
Simulator Reset.....	28	CODE .....	59
Single-step execution .....	2	DATA .....	59
Single-Step Execution .....	26	Listing .....	60
Source .....	50	NUMBER .....	60
Source Files		Type.....	60
Preparing.....	57	XDATA .....	59
Switching .....	57	Symbol Tables.....	59
Source level debugging .....	1	Symbol Value .....	59
Source Level Debugging.....	57	Symbolic debugging.....	1
Source Level Debugging File.....	9	Symbols.....	59
Source level debugging format.....	8	System configuration.....	6
Source Window.....	57	System Reset .....	28
Adding to Watch Window .....	58		
Breakpoints.....	58		

<b>T</b>	
Target device .....	7
TBC .....	14
Time base counter .....	14
Time Base Counter .....	69
Timer .....	2
Timer/Counter Controls .....	14
Timers .....	69
Tool Bar(DT63K) .....	88
Tool Bar(DTS63K) .....	97
Tool Bar(LCDTOOL) .....	103
TPR Command .....	118
Trace Data	
Saving .....	52
Searching .....	53
Trace Full(DT63K) .....	32
Trace Full(DTS63K) .....	37
Trace Memory	
Clear .....	48
Size .....	48
Trace Pointer .....	48
Trace Window .....	48
Tracing .....	48
Overview .....	48
Tracing Options .....	51
TRSAV Command .....	116
<b>U</b>	
Uninstalling .....	10
URST Command .....	113
User application circuit .....	6
User interface .....	1
User Reset Pin .....	14
<b>V</b>	
Value Lists .....	62
VER Command .....	106
VERSION Command .....	121
Version Information .....	65
VPR Command .....	118
<b>W</b>	
Wait Between Instructions .....	67
WAIT Command .....	118
Wait Setting .....	20
Watch	
Adding Items .....	46
Candidates .....	45
Changing Radix .....	47
Deleting Items .....	47
Disassembly Window .....	46
Reordering Items .....	47
SFR Window .....	46
Source Window .....	46
Symbol List .....	47
Watch List	
Loading .....	47
Modifying .....	47
Saving .....	47
Watch window .....	42
Watch Window .....	46
Watch Window Usage .....	45
Watchdog timer .....	14
Watchdog Timer .....	70
WDT .....	14
Windows 95 .....	1
Windows NT .....	1
Word .....	5
<b>X</b>	
XP .....	50
XY .....	50
XY Register .....	41
<b>Z</b>	
Zero Flag .....	41
Zoom .....	82