HP 64770

TLCS-9000 Emulator
Softkey Interface

User’s Guide
Notice

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchant ability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

© Copyright 1995, Hewlett-Packard Company.

This document contains proprietary information, which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another language without the prior written consent of Hewlett-Packard Company. The information contained in this document is subject to change without notice.

HP is a trademark of Hewlett-Packard Company.

UNIX is a registered trademark in the United States and other countries, licenced exclusively through X/Open Company Limited.

TLCS-9000™ is a trademark of Toshiba Electronics Inc.

Hewlett-Packard Company
P.O. Box 2197
1900 Garden of the Gods Road
Colorado Springs, CO 80901-2197, U.S.A.

RESTRICTED RIGHTS LEGEND Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013. Hewlett-Packard Company, 3000 Hanover Street, Palo Alto, CA 94304 U.S.A. Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).
New editions are complete revisions of the manual. The date on the title page changes only when a new edition is published.

A software code may be printed before the date; this indicates the version level of the software product at the time the manual was issued. Many product updates and fixes do not require manual changes and, manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual revisions.

Edition 1 64770-97001, June 1995
Using this Manual

This manual shows you how to use the following emulators with the Softkey Interface.

- HP 64770A/B TLCS-9000 emulator

This manual:

- Shows you how to use emulation commands by executing them on a sample program and describing their results.
- Shows you how to use the emulator in-circuit (connected to a target system).
- Shows you how to configure the emulator for your development needs. Topics include: restricting the emulator to real-time execution.

This manual does not:

- Show you how to use every Softkey Interface command and option; the Softkey Interface is described in the Softkey Interface Reference manual.

For the most part, the HP 64770A and HP 64770B emulators all operate the same way. Differences of between the emulators are described where they exist. Both the HP 64770A and HP 64770B emulators will be referred to as the "HP 64770A/B TLCS-9000 emulator" or "TLCS-9000 emulator". In the specific instances where HP 64770B emulator differs from HP 64770A emulator, it will be described as "HP 64770A emulator".
Organization

Chapter 1  **Introduction to the TLCS-9000 Emulator.** This chapter briefly introduces you to the concept of emulation and lists the basic features of the TLCS-9000 emulator.

Chapter 2  **Getting Started.** This chapter shows you how to use emulation commands by executing them on a sample program. This chapter describes the sample program and how to: load programs into the emulator, map memory, display and modify memory, display registers, step through program, run programs, set software breakpoints, search memory for data, and use the analyzer.

Chapter 3  **"In-Circuit" Emulation.** This chapter shows you how to install the emulator probe into a demo board and target system and how to use "in-circuit" emulation features.

Chapter 4  **Configuring the Emulator.** This chapter shows you how to: restrict the emulator to real-time execution, allow the target system to insert wait states, and select foreground or background monitor.

Chapter 5  **Using the Emulator.** This chapter describes emulation topics which are not covered in the "Getting Started" chapter.
Conventions

Example commands throughout the manual use the following conventions:

- **bold** Commands, options, and parts of command syntax.
- **bold italic** Commands, options, and parts of command syntax which may be entered by pressing softkey.
- **normal** User specified parts of a command.
- **$** Represents the HP-UX prompt. Commands which follow the "$" are entered at the HP-UX prompt.
- **<RETURN>** The carriage return key.
Contents

1 Introduction to the TLCS-9000 Emulator
   Introduction .................................................. 1-1
   Purpose of the Emulator ................................... 1-1
   Features of the TLCS-9000 Emulator ...................... 1-3
      Supported Microprocessors ............................... 1-3
      Clock Speeds ............................................. 1-4
      Emulation memory ....................................... 1-4
      Analysis .................................................. 1-5
      Registers ................................................ 1-5
      Emulation Monitor ...................................... 1-5
      Single-Step ............................................. 1-5
      Breakpoints ............................................. 1-5
      Reset Support .......................................... 1-5
      Real-Time Operation ................................... 1-5
      Coverage ............................................... 1-6
      Easy Products Upgrades ................................. 1-6
   Limitations, Restrictions ................................ 1-7
      Reset While in Monitor ................................ 1-7
      User Interrupts While in Monitor ...................... 1-7
      While Executing Step Command ......................... 1-7
      Watch Dog Timer (HP 64770A Only) .................... 1-7
      Vector Area ............................................. 1-7
      Register Bank .......................................... 1-8
      Unbreaking into the Monitor ........................... 1-8
      Emulation Memory ...................................... 1-8
      Evaluation Chip ......................................... 1-8

2 Getting Started
   Introduction .................................................. 2-1
   Before You Begin .......................................... 2-2
   Prerequisites ............................................... 2-2
   A Look at the Demo Program .............................. 2-2
   Assembling/Compiling the Demo Program ................. 2-3
   Linking the Demo Program ............................... 2-3
3 In-Circuit Emulation Topics

Introduction .................................................. 3-1
Prerequisites .................................................... 3-1
Installing the Emulation Probe Cable ...................... 3-2
Installing the Emulation Memory Module .................. 3-5
Installing into the Demo Target Board .................... 3-7
Installing into a Target System ............................. 3-9
   Installing into a QFP-PGA Adaptor ...................... 3-11
In-Circuit configuration Options ......................... 3-12
Running the Emulation from Target Reset ................. 3-12
Pin State in Background ..................................... 3-14
Target System Interface .................................... 3-15

4 Configuring the Emulator

Introduction .................................................. 4-1
General Emulator Configuration ............................ 4-4
   Restrict to Real-Time Runs? ............................. 4-4
   Processor type? (HP 64770A) .......................... 4-5
   Processor type? (HP 64770B) .......................... 4-5
   Processor operation mode? ............................. 4-6
   Monitor base address? ................................... 4-7
   Enable emulation VBP? .................................... 4-7
   Vector base address? ..................................... 4-7
   Initial CBP value? ........................................ 4-8
Memory Configuration ........................................ 4-9
   Mapping Memory .......................................... 4-9
Emulator Pod Configuration .................................. 4-11
   Enable watch dog timer? (HP 64770A Only) .......... 4-11
   Respond to BUSRQ from target system? .............. 4-11
   Respond to RESET from target system? ............... 4-11
   Respond to interrupts ? .................................. 4-12
   Target memory access size ............................. 4-12
Debug/Trace Configuration .................................. 4-13
   Break Processor on Write to ROM? ..................... 4-13
   Trace monitor or user program operation? ............. 4-14
   Emulation analyzer speed? (HP 64770A Only) .......... 4-14
Simulated I/O Configuration ............................... 4-15
Interactive Measurement Configuration ................... 4-15

Contents-3
5 Using the Emulator

- Introduction .......................................................... 5-1
- REGISTER CLASS and NAME ........................................ 5-2
- Hardware Breakpoints .................................................. 5-15
  - Single Chip Mode ......................................................... 5-15
  - External Bus Mode ...................................................... 5-15
- Vector Area Setting ..................................................... 5-15
- Analyzer Topics ............................................................ 5-17
  - Specifying Address and Status for Trigger or Store Condition 5-17
  - Specifying Data for Trigger or Store Condition ...................... 5-17
  - Specifying Execute Address for Trigger or Store Condition ........ 5-18
- Features Available via Pod Commands .................................. 5-19
- Storing Memory Contents to an Absolute File ......................... 5-20
- Coordinated Measurements .................................................. 5-20

Illustrations

Figure 1-1 HP 64770A/B Emulator for TLCS-9000 .................. 1-2
Figure 2-1 Linker Command File ........................................ 2-3
Figure 2-2 Softkey Interface Display .................................... 2-4
Figure 3-1 Installing cables to the control board .................... 3-2
Figure 3-2 Installing cables into cable sockets .................... 3-3
Figure 3-3 Installing cables to the emulation probe .................. 3-4
Figure 3-4 Opening the emulation probe cover ....................... 3-5
Figure 3-5 Installing the memory module .............................. 3-6
Figure 3-6 Installing the demo target board ....................... 3-7
Figure 3-7 Installing the power cable .................................. 3-8
Figure 3-8 Installing into a target system board .................... 3-11
Tables

Table 1-1 Supported Microprocessors (HP 64770A) . . . . . . . . . . . . 1-3
Table 1-2 Supported Microprocessors (HP 64770B) . . . . . . . . . . . . 1-4
Introduction to the TLCS-9000 Emulator

Introduction

The topics in this chapter include:

- Purpose of the emulator
- Features of the emulator
- Limitations and Restrictions of the emulator

Purpose of the Emulator

The TLCS-9000 emulator is designed to replace the TLCS-9000 microprocessor series in your target system to help you debug/integrate target system software and hardware. The emulator performs just like the processor which it replaces, but at the same time, it gives you information about the bus cycle operation of the processor. The emulator gives you control over target system execution and allows you to view or modify the contents of processor registers, target system memory, and I/O resources. Refer to "Memory Mapping" section in the "Using the Emulator" chapter.
Figure 1-1 HP 64770A/B Emulator for TLCS-9000

1-2 Introduction
Features of the TLCS-9000 Emulator

This section introduces you to the features of the emulator. The chapters which follow show you how to use these features.

Supported Microprocessors

The HP 64770A emulator supports the microprocessors listed in Table 1-1. The HP 64770B emulator supports the microprocessors listed in Table 1-2.

Table 1-1 Supported Microprocessors (HP 64770A)

<table>
<thead>
<tr>
<th>Supported Microprocessors</th>
<th>Internal ROM size</th>
<th>Internal RAM size</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMP97C241F</td>
<td>0</td>
<td>2K byte</td>
</tr>
<tr>
<td>TMP97PS40F</td>
<td>64K byte</td>
<td>2K byte</td>
</tr>
<tr>
<td>TMP97CS40F</td>
<td>64K byte</td>
<td>2K byte</td>
</tr>
<tr>
<td>TMP97CM40F</td>
<td>32K byte</td>
<td>1K byte</td>
</tr>
<tr>
<td>TMP97PW40F</td>
<td>128K byte</td>
<td>4K byte</td>
</tr>
<tr>
<td>TMP97CW40F</td>
<td>128K byte</td>
<td>4K byte</td>
</tr>
</tbody>
</table>
### Table 1-2 Supported Microprocessors (HP 64770B)

<table>
<thead>
<tr>
<th>Supported Microprocessors</th>
<th>Internal ROM size</th>
<th>Internal RAM size</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMP97CS42</td>
<td>64K byte</td>
<td>3.5K byte</td>
</tr>
<tr>
<td>TMP97PU42</td>
<td>64K byte</td>
<td>3.5K byte</td>
</tr>
<tr>
<td></td>
<td>96K byte</td>
<td>5.25K byte</td>
</tr>
<tr>
<td>TMP97CU42</td>
<td>96K byte</td>
<td>5.25K byte</td>
</tr>
<tr>
<td>TMP97PW42</td>
<td>128K byte</td>
<td>5.25K byte</td>
</tr>
<tr>
<td>TMP97CW42</td>
<td>128K byte</td>
<td>5.25K byte</td>
</tr>
</tbody>
</table>

**Clock Speeds**

The HP 64770A emulator runs with a target system clock from 4 to 20 MHz. The HP 64770B emulator runs with a target system clock from 4 to 16 MHz.

**Emulation memory**

The HP TLCS-9000 emulator can be used with one of the following Emulation Memory Modules.

- HP 64171A 256K byte Emulation Memory Module (35 ns)
- HP 64171B 1M byte Emulation Memory Module (35 ns)
- HP 64172A 256K byte Emulation Memory Module (20 ns)
- HP 64172B 1M byte Emulation Memory Module (20 ns)
- HP 64173A 4M byte Emulation Memory Module (25 ns)

You can define up to 7 memory ranges. You can characterize memory ranges as emulation RAM, emulation ROM, target system RAM, target system ROM, or guarded memory. The emulator generates an error message when accesses are made to guarded memory locations. You can also configure the emulator so that writes to memory defined as ROM cause emulator execution to break out of target program execution. Refer to the "Memory Mapping" section in the "Using the emulator" chapter.

**Analysis**

The HP 64770A emulator is used with one of the following analyzers which allows you to trace code execution and processor activity.

---

1-4 Introduction
The Emulation Bus Analyzer monitors the emulation processor using an internal analysis bus.

**Registers**
You can display or modify the TLCS-9000 internal register contents.

**Emulation Monitor**
The emulation monitor is a program that is executed by the emulation processor. It allows the emulation controller to access target system resources, and emulation memory. For example, when you display target system memory, it is monitor program that executes TLCS-9000 instructions which read the target memory locations and send their contents to the emulation controller.

The emulation monitor takes up 64K bytes of processor’s address space.

**Single-Step**
You can direct the emulation processor to execute a single instruction or a specified number of instructions.

**Breakpoints**
You can set up the emulator/analyzer interaction so that when the analyzer finds a specific state, emulator execution will break to the emulation monitor.

You can also define software breakpoints in your program. The emulator uses the undefined instruction(7F9Fh) to provide software breakpoint. When you define a software breakpoint, the emulator places a this undefined instruction at the specified address; after the undefined instruction causes emulator execution to break out of your program, the emulator replaces undefined instruction with the original opcode.

**Reset Support**
The emulator can be reset from the emulation system under your control, or your target system can reset the emulation processor.

**Real-Time Operation**
Real-time operation signifies continuous execution of your program without interference from the emulator. (Such interference occurs when the emulator temporarily breaks to the monitor so that it can access register contents or memory.)
You can restrict the emulator to real-time execution. When the emulator is executing your program under the real-time restriction, commands which display/modify registers, display/modify memory are not allowed.

**Coverage**
The TLCS-9000 emulator does not support coverage test.

**Easy Products Upgrades**
Because the HP 64700 Series development tools (emulator, analyzer, LAN board) contain programmable parts, it is possible to reprogram the firmware and some of the hardware without disassembling the HP 64700B Card Cage. This means that you’ll be able to update product firmware, if desired, without having to call an HP field representative to your site.
Limitations,
Restrictions

Reset While in
Monitor
If monitor program is running, \texttt{RESET} signal from target system is ignored while in monitor.

User Interrupts While
in Monitor
If the monitor is running, \texttt{NMI}, \texttt{INT0-7} (edge sense) for HP 64770A, \texttt{IREQ} for HP 64770B signals from target system are suspended until the emulator goes into user program operation. Other interrupts are ignored.

While Executing Step
Command
While stepping user program, interrupts are ignored. While single stepping, \texttt{BUSRQ} from target system is always ignored even if \texttt{BUSRQ} from target system is enabled.

Note
You should not use step command in case the interrupt handler’s punctuality is critical.

Watch Dog Timer
(HP 64770A Only)
When the HP 64770A emulator breaks into the monitor, the watched dog timer is resets, and disabled until the emulator goes into user program operation.
You must display/modify MDMOD register by "reg" command instead of "m" command.

Vector Area
You need to configure vector entry for the emulator to realize the following features.

- Break
- Single-Step
- Software Break Point

Refer to the "Vector Area Setting" section in the "Using the Emulator" Chapter in this manual.
Register Bank  When the emulator breaks into the monitor, the PC and PSW are stored at register bank of “CBP-1” in the same way as the emulator accepts interrupts.

Unbreaking into the Monitor  The emulator can not break into the monitor when the emulation processor is the following states.

- Standby Mode by HALT instruction
- Power Save state(Hardware standby mode) by PS signal
- Hold Mode by BUSRQ signal
- Reset state by RESET signal from target

Emulation Memory  When you use the emulator in single chip mode, you need the emulation memory because the emulator maps internal ROM/RAM area as emulation memory.

If you use the emulator in single chip mode or the emulation processor does burst fetch, the emulation memory module is restricted by clock speed as following.

HP 64770A  If clock speed is equal to 18MHz or greater 18MHz, you need HP64712A/B emulation memory module. If clock speed is less than 18MHz, you can use HP64712A/B and HP64713A emulation memory modules. If clock speed is less than 15MHz, you can use HP64171A/B, HP64172A/B and HP64713A emulation memory module.

HP 64770B  If clock speed is equal to 16MHz or less than 16MHz, you can use HP64712A/B and HP64713A emulation memory modules. If clock speed is less than 15MHz, you can use HP64171A/B, HP64172A/B and HP64713A emulation memory module.

Evaluation Chip  Hewlett-Packard makes no warranty of the problem caused by the TLCS-9000 Evaluation chip in the emulator.

1-8 Introduction
Getting Started

Introduction

This chapter will lead you through a basic, step by step tutorial that shows how to use the HP 64770A/B emulator (for the TLCS-9000 microprocessor) with the Softkey Interface.

This chapter will:

- Tell you what must be done before you can use the emulator as shown in the tutorial examples.

- Describe the demo program used for this chapter’s examples.

This chapter will show you how to:

- Start up the Softkey Interface.

- Load programs into emulation and target system memory.

- Enter emulation commands to view execution of the demo program.
Before You Begin

Prerequisites
Before beginning the tutorial presented in this chapter, you must have completed the following tasks:

1. Connected the emulator to your computer. The HP 64700 Series Installation/Service manual show you how to do this.

2. Installed the Softkey Interface software on your computer. Refer to the HP 64700 Series Installation/Service manual for instructions on installing software.

3. In addition, you should read and understand the concepts of emulation presented in the Concepts of Emulation and Analysis manual. The Installation/Service manual also covers HP 64700 system architecture. A brief understanding of these concepts may help avoid questions later.

You should read the Softkey Interface Reference manual to learn how to use the Softkey Interface in general. For the most part, this manual contains information specific to the TLCS-9000 emulator.

A Look at the Demo Program
The demo program is spmt_demo consisting of source program spmt_demo.c and init.s.

Where is the spmt_demo Software?
The demo program is shipped with the Softkey Interface and may be copied from the following directory.

/usr/hp64000/demo/emul/hp64770
Assembling/Compiling
the Demo Program
The demo program is written for and compiled/linked with the MICROTEC RESEARCH Inc. MCCT9K C Compiler Package. The demo program was assembled/compiled with the following commands.

```
$ mcct9k -c -g spmt_demo.c <RETURN>
$ asmt9k -f debug,casemcct9k -l init.s > init.lis <RETURN>
```

Linking the Demo Program
The following command was used to generate the absolute file. The "spmt_demo.cmd" linker command file is shown in figure 2-1.

```
$ lnkt9k -c spmt_demo.cmd -M<RETURN>
```

![Figure 2-1 Linker Command File](image)

Entering the Softkey Interface
If you have installed your emulator and Softkey Interface software as directed in the HP 64700 Series Emulators Softkey Interface Installation Notice, you are ready to enter the interface. The Softkey Interface can be entered from the HP-UX shell.

From the HP-UX Shell
If /usr/hp64000/bin is specified in your PATH environment variable, you can also enter the Softkey Interface with the following command.

```
$ emul700 <emul_name> <RETURN>
```

The "emul_name" in the command above is the logical emulator name given in the HP 64700 emulator device table (/usr/hp64000/etc/64700tab.net).

```
# Channel Logical Processor Remainder of Information for the Channel
# Type Name Type (IP address for LAN connections)
#---------------------------------------------------------------
lan: tlcs t9k40 21.17.9.143
```

If this command is successful, you will see a display similar to figure 2-2. The status message shows that the default configuration file has

Getting Started 2-3
been loaded. If the command is not successful, you will be given an error message and returned to the HP-UX prompt. Error messages are described in the *Softkey Interface Reference* manual.

---

**Figure 2-2 Softkey Interface Display**

To do operations described in this chapter (loading absolute program into emulation memory, displaying memory contents, etc), you need to configure the emulator as below. For detailed description of each configuration option (question), refer to the "Configuring the Emulator" chapter.

To get into the configuration session of the emulator, enter the following command.

```
modify configuration <RETURN>
```

Answer to the series of questions as below.

Restrict to real-time runs? no <RETURN>

Processor type? 97CM40 <RETURN>

When you use HP 64770A emulator, answer this question as shown.

Processor type? 97CU42 <RETURN>

When you use HP 64770B emulator, answer this question as shown.

---

2-4 Getting Started
Processor operation mode?  \textit{external\_bus} \texttt{<RETURN>}

Monitor base address?  \texttt{0F0000H} \texttt{<RETURN>}

Enable emulation VBP?  \texttt{yes} \texttt{<RETURN>}

When you use HP 64770A emulator, answer this question as shown.

Vector base address (97PS/CM40)?  \texttt{OFF0000H} \texttt{<RETURN>}

Vector base address (97CU42)?  \texttt{0FE7800H} \texttt{<RETURN>}

Initial CBP value?  \texttt{01H} \texttt{<RETURN>}

Modify memory configuration?  \texttt{yes} \texttt{<RETURN>}

Now you should be facing memory mapping screen. If you use HP 64770A emulator, two mapper terms must be specified for the demo program. Enter the following lines to map the program code area as emulation ROM, data area as emulation RAM. If you use HP 64770B emulator, you do not need to map because mapper terms for the demo program are specified automatically.

\texttt{400h tho 9ffh emulation ram} \texttt{<RETURN>}

\texttt{0a00h tho 0ffh emulation rom} \texttt{<RETURN>}

\texttt{end} \texttt{<RETURN>}

Modify emulator pod configuration?  \texttt{no} \texttt{<RETURN>}

Modify debug/trace options?  \texttt{no} \texttt{<RETURN>}

Modify simulated I/O configuration?  \texttt{no} \texttt{<RETURN>}

Modify interactive measurement specification?  \texttt{no} \texttt{<RETURN>}

If you wish to save the configuration specified above, answer this question as shown.

Configuration file name?  \texttt{spmt\_demo} \texttt{<RETURN>}

Now you are ready to go ahead. Above configuration is used throughout this chapter.

\textbf{Getting Started 2-5}
On-Line Help

There are two ways to access on-line help in the Softkey Interface. The first is by using the Softkey Interface help facility. The second method allows you to access the firmware resident Terminal Interface on-line help information.

Softkey Driven Help

To access the Softkey Interface on-line help information, type either "help" or "?" on the command line; you will notice a new set of softkeys. By pressing one of these softkeys and <RETURN>, you can cause information on that topic to be displayed on your screen. For example, you can enter the following command to access "system command" help information.

```
? system_commands <RETURN>
```

---SYSTEM COMMANDS & COMMAND FILES---

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>displays the possible help files</td>
</tr>
<tr>
<td>help</td>
<td>displays the possible help files</td>
</tr>
<tr>
<td>!</td>
<td>fork a shell (specified by shell variable SH)</td>
</tr>
<tr>
<td>!&lt;shell command&gt;</td>
<td>fork a shell and execute a shell command</td>
</tr>
<tr>
<td>pwd</td>
<td>print the working directory</td>
</tr>
<tr>
<td>cd &lt;directory&gt;</td>
<td>change the working directory</td>
</tr>
<tr>
<td>pws</td>
<td>print the default symbol scope</td>
</tr>
<tr>
<td>cws &lt;SYMB&gt;</td>
<td>change the working symbol - the working symbol also gets updated when displaying local symbols and displaying memory mnemonic</td>
</tr>
<tr>
<td>forward &lt;UI&gt; &quot;command&quot;</td>
<td>send the command in the quoted string from this user interface to another one. Replace &lt;UI&gt; with the name of the other user interface as shown on the softkeys:</td>
</tr>
</tbody>
</table>

The help information is scrolled on to the screen. If there is more than a screenful of information, you will have to press the space bar to see the next screenful, or the <RETURN> key to see the next line, just as you do with the HP-UX more command. After all the information on the particular topic has been displayed (or after you press "q" to quit scrolling through information), you are prompted to press <RETURN> to return to the Softkey Interface.

2-6 Getting Started
Pod Command Help

To access the emulator’s firmware resident Terminal Interface help information, you can use the following commands.

```
display pod_command <RETURN>
pod_command 'help cf' <RETURN>
```

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display pod_command</td>
<td>Display pod_command help information.</td>
</tr>
<tr>
<td>pod_command 'help cf'</td>
<td>Display long help for specified item.</td>
</tr>
</tbody>
</table>

--- VALID CONFIGURATION ITEM NAMES ---
- breq - en/dis /BUSRQ input from target system
- cbp - CBP value on break from reset state
- emvbp - en/dis emulation VBP
- int - en/dis interrupts
- loc - specify monitor location
- mode - select operation mode
- proc - select processor type
- rrt - en/dis restriction to real time runs
- trst - en/dis /RESET input from target system
- vector - specify vector address
- wdt - en/dis watch dog timer on break from reset state

STATUS: T9R40--Emulation reset________________________________________...R....
pod_command 'help cf'

The command enclosed in string delimiters (", ", or ^) is any Terminal Interface command, and the output of that command is seen in the pod_command display. The Terminal Interface help (or ?) command may be used to provide information on any Terminal Interface command or any of the emulator configuration options (as the example command above shows).

Note

If you want to use the Terminal Interface command by entering from keyboard directly, you can do it after entering the following command.

```
pod_command keyboard
```

Getting Started 2-7
The "load" command allows you to load absolute files into emulation or target system memory. You can load absolute files in the following formats:

- IEEE-695
- HP absolute (No symbols)

The "load" command has no special options for loading different absolute file formats; instead, the contents of the file are examined to determine the format being used. If you wish to load only that portion of the absolute file that resides in memory mapped as emulation RAM or ROM, use the "load emul_mem" syntax. If you wish to load only the portion of the absolute file that resides in memory mapped as target RAM, use the "load user_mem" syntax. If you want both emulation and target memory to be loaded, do not specify "emul_mem" or "user_mem". For example:

```
load spmt_demo <RETURN>
```

---

**Note**

When you use HP 64770B emulator, you must enter "break" command before you load a program. Enter the following command.

```
break <RETURN>
```
When loading a program if the status line shows

"ERROR: No absolute file, No database: spmt_demo"

you may NOT be in the directory that your program is in. To find out what directory you are in, enter:

! pwd <RETURN>

The "!" allows you to use an HP-UX shell command. To move into the correct directory, enter:

   cd <directory path> <RETURN>

You can also specify the pathname where your program resides. For example, you could enter:

   load /usr/hp64000/demo/emul/hp64770/spmt_demo <RETURN>
Displaying Symbols

When you load an absolute file into memory (unless you use the "nosymbols" syntax), symbol information is also loaded. Both global symbols and symbols that are local to a source file can be displayed.

Global

To display global symbols, enter the following command.

display global_symbols <RETURN>

Listed are address ranges associated with a symbol, the segment that the symbol is associated with, and the offset of that symbol within the segment.

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>Address range</th>
<th>Segment</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>apply_controller</td>
<td>000DBC - 000E0B</td>
<td>code</td>
<td>0396</td>
</tr>
<tr>
<td>apply_productions</td>
<td>000CC0 - 000D13</td>
<td>code</td>
<td>029A</td>
</tr>
<tr>
<td>calculate_answer</td>
<td>000E0C - 000E53</td>
<td>code</td>
<td>03E6</td>
</tr>
<tr>
<td>clear_buffer</td>
<td>000BEC - 000C1B</td>
<td>code</td>
<td>01C6</td>
</tr>
<tr>
<td>endcommand</td>
<td>000EFA - 000F05</td>
<td>code</td>
<td>04D4</td>
</tr>
<tr>
<td>format_result</td>
<td>000D14 - 000D43</td>
<td>code</td>
<td>02EE</td>
</tr>
<tr>
<td>get_next_token</td>
<td>000D80 - 000DDB</td>
<td>code</td>
<td>035A</td>
</tr>
<tr>
<td>initialize</td>
<td>000D44 - 000D7F</td>
<td>code</td>
<td>031E</td>
</tr>
<tr>
<td>input_line</td>
<td>000A26 - 000A57</td>
<td>code</td>
<td>0000</td>
</tr>
<tr>
<td>lookup_token</td>
<td>000C1C - 000C55</td>
<td>code</td>
<td>01F6</td>
</tr>
<tr>
<td>main</td>
<td>000F06 - 000F6B</td>
<td>code</td>
<td>04E0</td>
</tr>
<tr>
<td>math_library</td>
<td>000B38 - 000BAD</td>
<td>code</td>
<td>0112</td>
</tr>
<tr>
<td>move_byte</td>
<td>000A58 - 000A83</td>
<td>code</td>
<td>0032</td>
</tr>
<tr>
<td>outputline</td>
<td>000BAE - 000EBB</td>
<td>code</td>
<td>018B</td>
</tr>
<tr>
<td>parse_command</td>
<td>000E84 - 000EBD</td>
<td>code</td>
<td>045E</td>
</tr>
</tbody>
</table>

STATUS:  T9K40--Emulation reset....................................................................................................R....
display global_symbols

run  trace  step  display  modify  break  end  ---ETC--
When displaying local symbols, you must include the name of the source file in which the symbols are defined. For example,

```
  display local_symbols_in spmt_demo.c:
  <RETURN>
```

As you can see, the procedure symbols and static symbols in "spmt_demo.c" are displayed.

To list the next symbols, press the <PGDN> or <Next> key. The source reference symbols in "spmt_demo.c" will be displayed.

Listed are: address ranges associated with a symbol, the segment that the symbol is associated with, and the offset of that symbol within the segment.

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>Address range</th>
<th>Segment</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>apply_controller</td>
<td>000DBC - 000E0B</td>
<td>code</td>
<td>0396</td>
</tr>
<tr>
<td>apply_productions</td>
<td>000C0 - 000D13</td>
<td>code</td>
<td>029A</td>
</tr>
<tr>
<td>calculate_answer</td>
<td>000E0C - 000E53</td>
<td>code</td>
<td>03E6</td>
</tr>
<tr>
<td>clear_buffer</td>
<td>000BEC - 000C1B</td>
<td>code</td>
<td>01C6</td>
</tr>
<tr>
<td>endcommand</td>
<td>000EFA - 000F05</td>
<td>code</td>
<td>04D4</td>
</tr>
<tr>
<td>format_result</td>
<td>000D14 - 000D43</td>
<td>code</td>
<td>02EE</td>
</tr>
<tr>
<td>get_next_token</td>
<td>000D80 - 000DBB</td>
<td>code</td>
<td>035A</td>
</tr>
<tr>
<td>initialize</td>
<td>000D44 - 000D7F</td>
<td>code</td>
<td>031E</td>
</tr>
<tr>
<td>input_line</td>
<td>000A26 - 000A57</td>
<td>code</td>
<td>0000</td>
</tr>
<tr>
<td>lookup_token</td>
<td>000C1C - 000C55</td>
<td>code</td>
<td>01F6</td>
</tr>
<tr>
<td>main</td>
<td>000F06 - 000F6B</td>
<td>code</td>
<td>04E0</td>
</tr>
<tr>
<td>math_library</td>
<td>000B38 - 000BAD</td>
<td>code</td>
<td>0112</td>
</tr>
<tr>
<td>move_byte</td>
<td>000A58 - 000A83</td>
<td>code</td>
<td>0032</td>
</tr>
<tr>
<td>outputline</td>
<td>000B4E - 000BEB</td>
<td>code</td>
<td>018B</td>
</tr>
<tr>
<td>parse_command</td>
<td>000E84 - 000EBD</td>
<td>code</td>
<td>045E</td>
</tr>
</tbody>
</table>

STATUS:  cws: spmt_demo__________________________...R....
display local_symbols_in spmt_demo.c:

```
run  trace  step  display  modify  break  end  ---ETC--
```
To display the address ranges associated with the program’s source file, you must display the local symbols in the file. For example:

```
  display local_symbols_in spmt_demo.c:
  <RETURN>
```

And scroll the information down on the display with up arrow, or <Next> key.

```
Symbols in spmt_demo(module)."spmt_demo.c":
Source reference symbols
Line range _____________ Address range __ Segment _____________ Offset
#1-#35                         000A26 - 000A27   code                     0000
#36-#37                         000A28 - 000A29   code                     0002
#37-#37                         000A30 - 000A31   code                     0002
#37-#37                         000A32 - 000A33   code                     0002
#38-#39                         000A34 - 000A35   code                     0004
#39-#40                         000A36 - 000A37   code                     0004
#40-#40                         000A38 - 000A39   code                     0004
#41-#41                         000A3A - 000A3B   code                     0004
#42-#42                         000A3C - 000A3D   code                     0004
#43-#43                         000A3E - 000A3F   code                     0004
#44-#44                         000A40 - 000A41   code                     0004
#45-#46                         000A42 - 000A43   code                     0004
#46-#47                         000A44 - 000A45   code                     0004
#47-#48                         000A46 - 000A47   code                     0004
#48-#49                         000A48 - 000A49   code                     0004
#49-#50                         000A4A - 000A4B   code                     0004
#50-#51                         000A4C - 000A4D   code                     0004
#51-#51                         000A4E - 000A4F   code                     0004

STATUS:   cws: spmt_demo."spmt_demo.c":_________________________________...R....
display local_symbols_in spmt_demo.c:

run   trace   step   display   modify   break   end   ---ETC--

2-12 Getting Started
Displaying Memory in Mnemonic Format

You can display, in mnemonic format, the absolute code in memory. For example to display the memory of the demo program,

\[
\text{display memory main mnemonic}<\text{RETURN}>
\]

Notice that you can use symbols when specifying expressions. The global symbol \text{main} is used in the command above to specify the starting address of the memory to be displayed.
Display Memory with Symbols

If you want to see symbol information with displaying memory in mnemonic format, the emulator Softkey Interface provides "set symbols" command. To see symbol information, enter the following command.

```
set symbols on <RETURN>
```

<table>
<thead>
<tr>
<th>Memory</th>
<th>:mnemonic :file = spmt_demo(module).&quot;spmt_demo.c&quot;:</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>000F06</td>
</tr>
<tr>
<td>000F06</td>
<td>spmt_de.main</td>
</tr>
<tr>
<td>000F08</td>
<td>518E</td>
</tr>
<tr>
<td>000F0A</td>
<td>00D060AB00</td>
</tr>
<tr>
<td>000F10</td>
<td>00D05420</td>
</tr>
<tr>
<td>000F14</td>
<td>413F</td>
</tr>
<tr>
<td>000F16</td>
<td>6F3F</td>
</tr>
<tr>
<td>000F18</td>
<td>00D08AB900</td>
</tr>
<tr>
<td>000F1E</td>
<td>0AF3</td>
</tr>
<tr>
<td>000F20</td>
<td>05870AAF</td>
</tr>
<tr>
<td>000F24</td>
<td>BB88</td>
</tr>
<tr>
<td>000F26</td>
<td>A186</td>
</tr>
<tr>
<td>000F28</td>
<td>00D80AB992</td>
</tr>
<tr>
<td>000F2E</td>
<td>00D802BF92</td>
</tr>
<tr>
<td>000F34</td>
<td>00D00A18</td>
</tr>
<tr>
<td>000F38</td>
<td>00D803BF92</td>
</tr>
<tr>
<td>000F3E</td>
<td>00D060A192</td>
</tr>
</tbody>
</table>

As you can see, the memory display shows symbol information.

2-14 Getting Started
**Display Memory with Source Code**

If you want to reference the source line information with displaying memory in mnemonic format, the emulator Softkey Interface provides "set source" command. To reference the source line information in inverse video, enter the following command:

```plaintext
set source on inverse_video on <RETURN>
```

<table>
<thead>
<tr>
<th>Address</th>
<th>Label</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>371</td>
<td>/<em>-------------------------- main program --------------------------</em>/</td>
<td></td>
</tr>
<tr>
<td>372</td>
<td>main()</td>
<td></td>
</tr>
<tr>
<td>373</td>
<td>int dummyv;</td>
<td></td>
</tr>
<tr>
<td>374</td>
<td>dummyv = 1;</td>
<td></td>
</tr>
<tr>
<td>375</td>
<td>tasknumber = 0;</td>
<td></td>
</tr>
<tr>
<td>376</td>
<td>request_command();</td>
<td></td>
</tr>
<tr>
<td>377</td>
<td>parse_command();</td>
<td></td>
</tr>
<tr>
<td>378</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To see the memory without source line referencing, enter the following command:

```plaintext
set source off <RETURN>
```

Getting Started 2-15
Running the Program

The "run" command lets you execute a program in memory. Entering the "run" command by itself causes the emulator to begin executing at the current program counter address. The "run from" command allows you to specify an address at which execution is to start.

From Transfer Address

The "run from transfer_address" command specifies that the emulator start executing at a previously defined "start address". Transfer addresses are defined in assembly language source files with the END assembler directive (i.e., pseudo instruction). Enter:

`run from transfer_address <RETURN>`

From Reset

The "run from reset" command specifies that the emulator begin executing from reset vector as actual microprocessor does.

(See "Running From Reset" section in the "In-Circuit Emulation" chapter).

Displaying Memory

The demo program "spmt_demo.c" alters memory.

Using Symbolic Addresses

In the following display, the memory range is displayed using symbolic addresses `data`.

The memory display window is periodically updated. For example, enter the following command:

`display memory data thru +7fh blocked bytes <RETURN>`

This command string is used to specify the range of memory from `data` to `data+7fh`.
### Modifying Memory

You can use the `modify memory` command to send commands to the sample program. Memory locations `stackarea` and `stackarea+10h` correspond to memory address 804 hex and 814 hex respectively. For example, to enter the '10h' at address 804 and enter 'A' at address 814:

- Use the following commands.

```plaintext
display memory stackarea <RETURN>
modify memory stackarea to 10h <RETURN>
modify memory stackarea+10h string to 'A' <RETURN>
```

After the memory location are modified, the memory display shows the following.
Breaking into the Monitor

The "break" command allows you to divert emulator execution from the user program to the monitor. You can continue user program execution with the "run" command. To break emulator execution from the demo program to the monitor, enter the following command.

**break** <RETURN>

Notice that the current address is pointed out with inverse video in displaying memory when the execution breaks to the monitor.
Using Software Breakpoints

Software breakpoints are handled by the TLCS-9000 undefined instruction (breakpoint interrupt instruction:7F9Fh). When you define or enable a software breakpoint, the emulator will replace the opcode at the software breakpoint address with a breakpoint interrupt instruction.

Caution

Software breakpoints should not be set, cleared, enabled, or disabled while the emulator is running user code. If any of these commands are entered while the emulator is running user code and the emulator is executing code in the area where the breakpoint is being modified, program execution may be unreliable.

Note

You must only set software breakpoints at memory locations which contain instruction opcodes (not operands or data). If a software breakpoint is set at a memory location which is not an instruction opcode, the software breakpoint instruction will never be executed. Further, your program won’t work correctly.

Note

NMI will be ignored, when software breakpoint and NMI occur at the same time.

Note

Because software breakpoints are implemented by replacing opcodes with the breakpoint interrupt instruction, you cannot define software breakpoints in target ROM. Then you can use software breakpoints.
When software breakpoints are enabled and the emulator detects the breakpoint interrupt instruction, it generates a break into the monitor. Since the system controller knows the locations of defined software breakpoints, it can determine whether the breakpoint instruction in your target program.

If the breakpoint interrupt was generated by a software breakpoint, execution breaks to the monitor, and the breakpoint interrupt instruction is replaced by original opcode. A subsequent run or step command will execute from this address.

If the breakpoint interrupt was generated by a undefined instruction (7F9Fh) in the target program, execution still breaks to the monitor, and an "undefined breakpoint" status message is displayed. To continue program execution, you must run or step from the target program’s breakpoint interrupt vector address.

Enabling/Disabling Software Breakpoints

When you initially enter the Softkey Interface, software breakpoints are disabled. To enable the software breakpoints feature, enter the following command.

```
modify software_breakpoints enable <RETURN>
```

When software breakpoints are enabled and you set a software breakpoint, the TLCS-9000 breakpoint interrupt instruction (7F9Fh) will be placed at the address specified. When the breakpoint interrupt instruction is executed, program execution will break into the monitor.

Setting a Software Breakpoint

To set a software breakpoint at line 80 of "spmt_demo.c", enter the following command.

```
modify software_breakpoints set line 80 <RETURN>
```

To see the address where the software breakpoint has been set, enter the following command:

```
display memory line 80 mnemonic <RETURN>
set source on inverse_video on <RETURN>
```
The asterisk (*) in left side of the address lists points out that the software breakpoint has been set. The opcode at the software breakpoint address was replaced to the software breakpoint instruction.

### Displaying Software Breakpoints

To display software breakpoints, enter the following command.

```
display software_breakpoints <RETURN>
```
The software breakpoints display shows that the breakpoint is pending. When breakpoints are hit they become inactivated. To reactivate the breakpoint so that it is "pending", you must re-enter the "modify software_breakpoints set" command.

After the software breakpoint has been set, enter the following command to cause the emulator to continue executing the demo program.

```
run <RETURN>
```

A message on the status line shows that the software breakpoint has been hit. The status line also shows that the emulator is now executing in the monitor.

The software breakpoint address is pointed out with inverse video in displaying memory in mnemonic format. To see the software breakpoint with memory, enter the following command.

```
display memory line 80 mnemonic <RETURN>
```

Notice that the original opcode was replaced at the address that the software breakpoint has been set.

2-22 Getting Started
Clearing a Software Breakpoint

To remove software breakpoint defined above, enter the following command.

```
modify software_breakpoints clear line 80
<RETURN>
```

The breakpoint is removed from the list, and the original opcode is restored if the breakpoint was pending.

To clear all software breakpoints, you can enter the following command.

```
modify software_breakpoints clear <RETURN>
```

Displaying Registers

Enter the following command to display registers. You can display the basic registers, or an individual register. Refer to "REGISTER CLASS and NAME" section in "Using the Emulator" chapter.

```
display registers <RETURN>
```

```
<table>
<thead>
<tr>
<th>Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next_PC 000ABC</td>
</tr>
<tr>
<td>PC 000ABC CBP 01 PBP 00 PSW 00000808</td>
</tr>
<tr>
<td>RW0 0000 RW1 0000 RW2 0000 RW3 0000 RW4 0000 RW5 0000</td>
</tr>
<tr>
<td>RW6 0000 RW7 0000 RW8 0000 RW9 0000 RW10 0002 RW11 0001</td>
</tr>
<tr>
<td>RW12 0003 RW13 0000 RW14 0000 RW15 0000</td>
</tr>
<tr>
<td>ISP 000007E8 USP 00000000 FP 00000000</td>
</tr>
</tbody>
</table>
```

```
STATUS: T9K40--Running in monitor Software break: 0000abc_________R....
display registers
run trace step display modify break end ---ETC--
```
Stepping Through the Program

The step command allows you to step through program execution an instruction or a number of instructions at a time. Also, you can step from the current program counter or from a specific address. To step through the example program from the address of the software breakpoint set earlier, enter the following command.

`step <RETURN>, <RETURN>, <RETURN>, ...`

You will see the inverse-video moves according to the step execution. You can continue to step through the program just by pressing the <RETURN> key.

---

You can step program execution by source lines, enter:

`step source <RETURN>`

Source line stepping is implemented by single stepping assembly instructions until the next PC is outside of the address range of the current source line. When source line stepping is attempted on assembly code, stepping will complete when a source line is found. To terminate stepping type <Ctrl>-C.

---

2-24 Getting Started
Using the Analyzer

HP 64700 emulators contain an emulation analyzer. The emulation analyzer monitors the internal emulation lines (address, data, and status).

Source Line Referencing

A trace may be taken and displayed using source line referencing. Also, lines of the source program can be displayed with the trace list where the trace occurred.

To display the trace with source code in inverse video, enter the following command:

```
set source on inverse_video on <RETURN>
```

Specifying a Simple Trigger

Suppose you want you trace program execution after the point at address `semantic_check`. The following command make this trace specification.

```
trace after semantic_check <RETURN>
```

The STATUS message shows "Emulation trace started."

Enter the following command to cause sample program execution to continue from the current program counter.

```
run <RETURN>
```

The STATUS message shows "Emulation trace complete."
Display the Trace

The trace listings which following are of program execution on the TLCS-9000 emulator. To see the trace list, enter the following command:

```
display trace <RETURN>
```

<table>
<thead>
<tr>
<th>Trace List</th>
<th>Depth=8192</th>
<th>Offset=0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label:</td>
<td>Address</td>
<td>Data</td>
</tr>
<tr>
<td>Base:</td>
<td>symbols</td>
<td>hex</td>
</tr>
<tr>
<td></td>
<td>mnemonic w/symbols</td>
<td>relative</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td>#spmt_demo.c - line 200 brave 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>after</td>
<td>=syntax_ch+00002C</td>
<td>A104</td>
</tr>
<tr>
<td>+001</td>
<td>s.semantic_check</td>
<td>A104 A104</td>
</tr>
<tr>
<td>+002</td>
<td>st</td>
<td>init.s+0003E8</td>
</tr>
<tr>
<td>+003</td>
<td>=syntax_ch+00002E</td>
<td>AB04</td>
</tr>
<tr>
<td>+004</td>
<td>semantic_+000002</td>
<td>AB04</td>
</tr>
<tr>
<td>+005</td>
<td>st</td>
<td>init.s+0003EA</td>
</tr>
<tr>
<td>+006</td>
<td>st</td>
<td>init.s+0003EC</td>
</tr>
<tr>
<td>+007</td>
<td>apply_con+00001A</td>
<td>8641 8641</td>
</tr>
</tbody>
</table>

| #spmt_demo.c - line 292 brave 0                      |
| for (i = 0; i 1 * 3; i++) |
| +008 | -apply_con+00001C | 8E43 ADD.W:S RW4,1 |
| +009 | apply_con+00001C | 8E43 8E43 fetch |

### spmt_demo.c - line 292 brave 0                      |

STATUS: T9K40--Running user program Emulation trace complete...R....

display trace

The trace list shows the trace after line (semantic_check()).

To list the next lines of the trace, press the <PGDN> or <NEXT> key.
Displaying Trace with No Symbol

The trace listing shown above has symbol information because of the "set symbols on" setting before in this chapter. To see the trace listing with no symbol information, enter the following command.

```
set symbols off <RETURN>
```

As you can see, the analysis trace display shows the trace list without symbol information.

```
Trace List  Depth=8192  Offset=0
Label:  Address  Data  Opcode or Status w/ Source Lines  time count
Base:   hex  hex  mnemonic  relative

after = 000C82  A104  INSTRUCTION--opcode unavailable
+001  000C86  A104  A104 fetch
+002  0007E8  0000  0000 read mem word
+003  000C84  AB04  INSTRUCTION--opcode unavailable
+004  000C88  AB04  AB04 fetch
+005  0007EA  0DD6  0DD6 read mem word
+006  0007EC  0000  0000 read mem word
+007  000DD6  8641  8641 fetch

for (i = 0; i * 3; i++)
+008  000DD6  8E43  ADD.W:S RW4,1
+009  000DD8  8E43  8E43 fetch

After
```

Getting Started 2-27
Displaying Trace with Compress Mode

If you want to see more executed instructions on a display, the TLCS-9000 emulator Softkey Interface provides **compress mode** for analysis display. To see trace display with compress mode, enter the following command:

```
display trace compress on <RETURN>
```

<table>
<thead>
<tr>
<th>Trace List</th>
<th>Depth=8192</th>
<th>Offset=0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label: Address</td>
<td>Data</td>
<td>Opcode or Status w/ Source Lines</td>
</tr>
<tr>
<td>Base: hex</td>
<td>hex</td>
<td>mnemonic</td>
</tr>
<tr>
<td>000C82</td>
<td>A104</td>
<td>INSTRUCTION--opcode unavailable</td>
</tr>
<tr>
<td>0007E8</td>
<td>0000</td>
<td>0000 read mem word</td>
</tr>
<tr>
<td>000C84</td>
<td>AB04</td>
<td>INSTRUCTION--opcode unavailable</td>
</tr>
<tr>
<td>0007EA</td>
<td>00D6</td>
<td>00D6 read mem word</td>
</tr>
<tr>
<td>0007EC</td>
<td>0000</td>
<td>0000 read mem word</td>
</tr>
<tr>
<td>000DD6</td>
<td>8E43</td>
<td>ADD.W:S RW4,1</td>
</tr>
<tr>
<td>000DD8</td>
<td>1BFA</td>
<td>CP.W:S RW4,3</td>
</tr>
<tr>
<td>000DDA</td>
<td>AB04</td>
<td>JRC LT,000DD4</td>
</tr>
</tbody>
</table>

As you can see, the analysis trace display shows the analysis trace lists without fetch cycles. With this command you can examine program execution easily.

If you want to see all of cycles including fetch cycles, enter following command:

```
display trace compress off <RETURN>
```

The trace display shows you all of the cycles the emulation analyzer have captured.

2-28 Getting Started
Trigger the Analyzer at an Instruction Execution State

The emulator analyzer can capture states of instruction execution. If you want to trigger the analyzer when an instruction at a desired address is executed, you should not set up the analyzer trigger condition to detect the address. If you do so, the analyzer will be also triggered in case that the address is accessed to fetch the instruction, or read the data from address. You should use the execution address(eaddr) qualifier. Suppose that you want to trace the states of the execution after the instruction at `clear_buffer` of the `spmt_demo.c` file, enter the following command.

```
trace after eaddr clear_buffer <RETURN>
```

The message "Emulation trace started" will appear on the status line, and the status line now shows "Emulation trace complete".

```
Trace List    Depth=8192    Offset=0
Label:    Address Data    Opcode or Status w/ Source Lines    time count
Base:    hex    hex                    mnemonic                    relative

############spmt_demo.c - line 153 thru 157 ############################
/******************** level three ********************/
clear_buffer()
{
    after = 000BEC   D000  INSTRUCTION--opcode unavailable           ------------
+002    0007E8     0000    0000  write mem word                460      nS
############spmt_demo.c - line 158 thru 159 ############################
    int i;
    for (i = 0; i < 3; i++)
        data = 0;

STATUS:   T9K40--Running user program     Emulation trace complete______...R....
trace after eaddr clear_buffer
```

The emulator has disassemble capability in trace listing. When the emulator disassembles instructions in stored trace information, the fetch cycles of each instruction are required. When you displayed the results of analyzer trace, some lines which include "INSTRUCTION--opcode unavailable" message may be displayed. Each line is instruction execution cycle at the address in the left side of the displayed because the fetch states for the instructions were not stored by the analyzer.

```
run    trace    step    display    modify    break    end    ---ETC---
```

Getting Started 2-29
To display complete disassembles in the trace listing, you should modify location of trigger state in trace list, referred to as the "trigger position", to "about" instead of "after".

Displaying trace option

You can specify whether the emulator display only bus cycles, or only execution cycles, or both cycles. To specify, the TLCS-9000 emulator Softkey Interface provides display trace option. To display only bus cycles, enter the following command:

```
display trace mnemonic option bus_cycles_only <RETURN>
```

If you want to display only execution cycles, enter the following command:

```
display trace mnemonic option exec_cycles_only <RETURN>
```

If you want to display bus cycles and execution cycles, enter the following command:

```
display trace mnemonic option both_cycles <RETURN>
```

Emulator Analysis Status Qualifiers

The following analysis status qualifiers may also be used with the TLCS-9000 emulator.

<table>
<thead>
<tr>
<th>Qualifier</th>
<th>Status bits</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bus</td>
<td>0x00xxxxxxxy</td>
<td>bus cycle</td>
</tr>
<tr>
<td>byte</td>
<td>0x010xxxxxxxy</td>
<td>byte memory cycle</td>
</tr>
<tr>
<td>exec</td>
<td>0x001xxxxxxxy</td>
<td>execute instruction</td>
</tr>
<tr>
<td>fetch</td>
<td>0x010xxxxxxxy</td>
<td>program fetch</td>
</tr>
<tr>
<td>halt</td>
<td>0x001xxxxxxxy</td>
<td>halt</td>
</tr>
<tr>
<td>intack</td>
<td>0x000xxxxxxxy</td>
<td>interrupt acknowledge</td>
</tr>
<tr>
<td>monitor</td>
<td>0x000xxxxxxxy</td>
<td>monitor cycle</td>
</tr>
<tr>
<td>read</td>
<td>0x010xxxxxxxy</td>
<td>read</td>
</tr>
<tr>
<td>user</td>
<td>0x001xxxxxxxy</td>
<td>user program cycle</td>
</tr>
<tr>
<td>word</td>
<td>0x010xxxxxxxy</td>
<td>word memory cycle</td>
</tr>
<tr>
<td>write</td>
<td>0x0100xxxxxxxy</td>
<td>write</td>
</tr>
</tbody>
</table>

For a Complete Description

For a complete description of using the HP 64700 Series analyzer with the Softkey Interface, refer to the Analyzer Softkey Interface User’s Guide.
Resetting the Emulator

To reset the emulator, enter the following command.

```
reset <RETURN>
```

Exiting the Softkey Interface

There are several options available when exiting the Softkey Interface: exiting and releasing the emulation system, exiting with the intent of re-entering (continuing), exiting locked from multiple emulation windows, and exiting (locked) and selecting the measurement system display or another module.

End Release System

To exit the Softkey Interface, releasing the emulator so that other users may use the emulator, enter the following command.

```
end release_system <RETURN>
```

Ending to Continue Later

You may also exit the Softkey Interface without specifying any options; this causes the emulator to be locked. When the emulator is locked, other users are prevented from using it and the emulator configuration is saved so that it can be restored the next time you enter (continue) the Softkey Interface.

```
end <RETURN>
```

Ending Locked from All Windows

When using the Softkey Interface from within window systems, the "end" command with no options causes an exit only in that window. To end locked from all windows, enter the following command.

```
end locked <RETURN>
```

This option only appears when you enter the Softkey Interface via the `emu700` command.

Refer to the *Softkey Interface Reference* manual for more information on using the Softkey Interface with window systems.
In-Circuit Emulation Topics

Introduction

Many of the topics described in this chapter involve the installation, and the commands which relate to using the emulator in-circuit, that is, connected to a target system or demo target board.

This chapter will:

- Show you how to install the emulation probe cable
- Show you how to install the emulation memory module.
- Show you how to install the emulation probe to demo target board.
- Describe the issues concerning the installation of the emulation probe into target systems.
- Describe how to execute program from target reset. This topics is related to program execution in general.
- Describe how to use software breakpoints with ROMed code, and how to test patches to ROMed code. These topics relate to the debugging of target system ROM.

Prerequisites

Before performing the tasks described in this chapter, you should be familiar with how the emulator operates in general. Refer to the Concepts of Emulation and Analysis manual and the "Getting Started" chapter of this manual.
Installing the Emulation Probe Cable

The probe cables consist of three ribbon cables. The longest cable connects to J3 of the emulation control card, and to J3 of the probe. The shortest cable connects to J1 of the emulation control card and J1 of the probe. The ribbon cables are held in place on the emulation control card by a cable clamp attached with two screws. No clamp holds the ribbon cables in the probe.

1. Secure the cable on the emulation control card with cable clamp and two screws.

Figure 3-1 Installing cables to the control board

3-2 In-Circuit Emulation Topics
2. When insert the ribbon cables into the appropriate sockets, press inward on the connector clips so that they into the sockets as shown.

Figure 3-2 Installing cables into cable sockets
3. Connect the other ends of the cables to the emulation probe.

Figure 3-3 Installing cables to the emulation probe

3-4 In-Circuit Emulation Topics
Installing the Emulation Memory Module

There are four types of emulation memory modules that can be inserted into sockets on the probe.

1. Remove plastic rivets that secure the plastic cover on the top of the emulation probe, and remove the cover. The bottom cover is only removed when you need to replace a defective active probe on the exchange program.

Figure 3-4 Opening the emulation probe cover
2. Insert emulation memory module on the emulation probe. There is a cutout on one side of the memory modules so that they can only be installed one way.

To install memory modules, place the memory module into the socket groove at an angle. Firmly press the memory module into the socket to make sure it is completely seated. Once the memory module is seated in the connector groove, pull the memory module forward so that the notches on the socket fit into the holes on the memory module. There are two latches on the sides of the socket that hold the memory module in place.

![Figure 3-5 Installing the memory module](image)

3. Replace the plastic cover, and insert new plastic rivets to secure the cover.
Installing into the Demo Target Board

To connect the microprocessor connector to the demo target board, proceeded with the following instructions:

1. Remove front bezel and connect the power cable to the connector of the HP 64700A front panel. Refer to the HP 64700 Series Installation/Service manual.

2. With HP 64700A power OFF, connect the emulation probe to the demo target board. When you install the emulation probe into the demo target board, be careful not to bend any of the pins.

After connection the probe to the demo target board, set the TEST/TARGET MODE and SINGLE CHIP/EXTERNAL BUS MODE switches. Use TEST MODE position when you run performance verification test, and use TARGET MODE position when you run the emulator in "out-of-circuit" mode. You must set SINGLE CHIP/EXTERNAL BUS switch according to "Processor operation mode?" configuration.

Figure 3-6 Installing the demo target board
3. Connect the power cable supply wires from the emulator to demo target board. When attaching the wire cable to the demo target board, make sure the connector is aligned properly so that all three pins are connected.

Figure 3-7 Installing the power cable
### Installing into a Target System

The TLCS-9000 emulation probe has a 135-pin PGA connector; the emulation probe is also provided with a conductive pin protector to protect the delicate gold-plated pins of the probe connector from damage due to impact.

**Caution**

**Protect against electrostatic discharge.** The emulator probe contains devices that are susceptible to damage by electrostatic discharge. Therefore, precautionary measures should be taken before handling the microprocessor connector attached to the end of the probe cable to avoid damaging the internal components of the probe by electrostatic electricity.

**Caution**

**Make sure target system power is OFF.** Do not install the emulation probe into the target system microprocessor socket with power applied to the target system. The emulator may be damaged if target system power is not removed before probe installation.

**Caution**

**Make sure pin 1 of probe connector is aligned with pin 1 of the socket.** When installing the emulation probe, be sure that probe is inserted into the processor socket so that pin 1 of the connector aligns with pin 1 of the socket. Damage to the emulator probe will result if the probe is incorrectly installed.

**Caution**

**DO NOT use the microprocessor connector without using a pin protector.** The pin protector prevents damage to the probe when inserting and removing the probe from the flexible adapter.
Caution

Compatibility of VOLTAGE/CURRENCY. Please be sure to check that the voltage/currency of the emulator and target system being connected are compatible. If there is a discrepancy, damage may result.

Caution

Do not apply strong force to PGA-QFP probe, as that might damage the probe.

Caution

Turn ON power. When you start to use the 64770A/B emulator which is plugged into a target system, you must turn HP 64770A/B power ON at first, then turn target system power ON.

Caution

Turn OFF power. Do not turn HP 64770A/B power OFF while the emulator is plugged into a target system whose power is ON.

3-10 In-Circuit Emulation Topics
## Installing into a QFP-PGA Adaptor

To connect the microprocessor connector to the target system, proceed with the following instructions.

1. Attach the QFP socket/adapter (YAMAICHI IC149-120K13207-0B) on your target system.

2. Connect the PGA-QFP probe (64770-61602) to the emulation probe through PGA connector (1200-1840).

3. Install the PGA-QFP probe to the QFP socket/adaptor on your target system.

![Diagram of Installing into a target system board]

Figure 3-8 Installing into a target system board
The TLCS-9000 emulation provides configuration options for the following in-circuit emulation issues. Refer to the "Configuring the Emulator" chapter for more information on these configuration options.

**In-Circuit configuration Options**

### Enabling BUSRQ, NMI, RESET and INT0-7 (for HP 64770A), IREQ (for HP 64770B) Input from the Target System

You can configure whether the emulator should accept or ignore the BUSRQ, NMI, RESET and INT0-7 (for HP 64770A), IREQ (for HP 64770B) signals from the target system.

### Running the Emulation from Target Reset

You can specify that the emulator begins execution from target system reset. When the target system RESET line becomes active and then inactive, the emulator will start reset sequence (option) as actual microprocessor.

At first, you must specify the emulator responds to RESET signal by the target system (see the "Enable RESET inputs from target system?" configuration in "Configuring the Emulator" chapter on this manual).

To specify a run from target system reset, enter the following command:

```
run from reset <RETURN>
```

The status now shows that the emulator is "Awaiting target reset". After the target system is reset, the status line message will change to show the appropriate emulator status.
In the "Awaiting target reset" status, you cannot break into the monitor. If you enter "run from reset" in the configuration that emulator ignores target system reset, you must reset the emulator.

After you turn on the emulator, you must enter "reset" command and then "break" command to set the emulation stack pointer.

The TLCS-9000 emulator supports power on reset. If you want your program to be executed by power on reset, execute the following process.

1) Enter "reset"

2) Enter "break"

3) Enter "run from reset"

4) Turn OFF your target system

4-1) If you see the p> system prompt, enter "run from reset" again.

5) Turn ON your target system
## Pin State in Background

While the emulator is running in the monitor, the probe pins of the emulator are in the following state.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Bus</td>
<td>Same as running user’s program.</td>
</tr>
<tr>
<td>Data Bus</td>
<td>Same as running user’s program.</td>
</tr>
<tr>
<td>BS R/W</td>
<td>Same as running user’s program.</td>
</tr>
<tr>
<td>UB/WEH</td>
<td>Same as running user’s program except accessing monitor area. When accessing monitor area, High level.</td>
</tr>
<tr>
<td>LB/WEL</td>
<td></td>
</tr>
<tr>
<td>CAS/OE</td>
<td></td>
</tr>
<tr>
<td>RAS0/CE0</td>
<td></td>
</tr>
<tr>
<td>RAS1/CE1</td>
<td></td>
</tr>
<tr>
<td>RAS2/CE2</td>
<td></td>
</tr>
<tr>
<td>RAS3/CE3</td>
<td></td>
</tr>
<tr>
<td>RFSH/CE</td>
<td>Same as running user’s program except accessing monitor area. When accessing monitor area, Low level.</td>
</tr>
</tbody>
</table>

---

3-14 In-Circuit Emulation Topics
Target System Interface

**RESET, PS, NMI**
These signals are connected to 74HC14 through 10K ohm pull-up register.

**EA**
These signals are connected to 74ABT16244 through 10K ohm pull-up register.

**Other signals**
These signals are connected to TLCS-9000 emulation processor.

In-Circuit Emulation Topics 3-15
3-16 In-Circuit Emulation Topics
Configuring the Emulator

Introduction

Your TLCS-9000 emulator can be used in all stages of target system development. For instance, you can run the emulator out-of-circuit when developing target system software, or you can use the emulator in-circuit when integrating software with target system hardware. Emulation memory can be used in place of, or along with, target system memory. You can use the emulator’s internal clock or the target system clock. You can execute target programs in real-time or allow emulator execution to be diverted into the monitor when commands request access of target system resources (target system memory, register contents, etc.)

The emulator is a flexible instrument and it may be configured to suit your needs at any stage of the development process. This chapter describes the options available when configuring the TLCS-9000 emulator.

The configuration options are accessed with the following command.

`modify configuration <RETURN>`

After entering the command above, you will be asked questions regarding the emulator configuration. The configuration questions are listed below and grouped into the following classes.
General Emulator Configuration:
  – Restricting to real-time execution.
  – Selecting processor type.
  – Specifying processor operation mode.
  – Specifying value of the monitor base address.
  – Enabling emulation Vector Base Pointer.
  – Specifying value of Vector base address.
  – Specifying initial Current Bank Pointer value.

Memory Configuration:
  – Mapping memory.

Emulator Pod Configuration:
  – Enabling watch dog timer.
  – Enabling BUSRQ input from target system.
  – Enabling RESET input from target system.
  – Enabling interrupt requests.
  – Selecting target memory access size.

Debug/Trace Configuration:
  – Enabling breaks on writes to ROM.
  – Specifying tracing of user program/emulation monitor cycles.
  – Selecting emulation analyzer speed.
**Simulated I/O Configuration:** Simulated I/O is described in the *Simulated I/O* reference manual.

**Interactive Measurement Configuration:** See the chapter on coordinated measurements in the *Softkey Interface Reference* manual.
General Emulator Configuration

The configuration questions described in this section involve general emulator operation.

Restrict to Real-Time Runs?

This configuration allows you to specify whether program execution should take place in real-time or whether commands should be allowed to cause breaks to the monitor during program execution.

**no**

- All commands, regardless of whether or not they require a break to the emulation monitor, are accepted by the emulator.

**yes**

- When runs are restricted to real-time and the emulator is running the user program, all commands that cause a break (except "reset", "break", "run", and "step") are refused. For example, the following commands are not allowed when runs are restricted to real-time:
  - Display/modify registers.
  - Display/modify memory.

Caution

If your target system circuitry is dependent on constant execution of program code, you should restrict the emulator to real-time runs. This will help insure that target system damage does not occur. However, remember that you can still execute the "run", "reset", "break", and "step" commands; you should use caution in executing these commands.

4-4 Configuring the Emulator
Processor type? (HP 64770A)

This question allows you to select which microprocessor to be emulated.

97PS40
The TMP97PS40F, TMP97CS40F, and TMP97C241F microprocessors are emulated. When you emulate TMP97C241F microprocessor, you must specify "Processor operation mode? external_bus".

97CM40
The TMP97CM40F microprocessor is emulated.

97PW40
The TMP97PW40F, TMP97CW40F microprocessors are emulated.

NONE
no valid processor is selected. This is a power up default and can not break into monitor from reset until valid processor is selected.

Processor type? (HP 64770B)

This question allows you to select which microprocessor to be emulated.

97CS42
The TMP97CS42, TMP97PU42(64K mode) microprocessors is emulated.

97CU42
The TMP97CU42, TMP97PU42(96K mode) microprocessor are emulated.

97CW42
The TMP97PW42, TMP97CW42 microprocessors are emulated.

NONE
no valid processor is selected. This is a power up default and can not break into monitor from reset until valid processor is selected.

Configuring the Emualtor 4-5
You must specify processor type before operation the emulator, Otherwise, you can not operate the emulator correctly.

Changing this configuration setting will drive the emulator into a reset state and will reset the memory mapping. Monitor address and vector address configurations will be set to their default.

This configuration allows to you specify whether operation mode is single chip mode or external bus mode.

**single** The emulator will operate in single chip mode.

**external_bus** The emulator will operate in external bus mode.

TLCS-9000 emulator operates in accordance with this configuration instead of EA signal from target system.

But when the emulator breaks into the monitor from reset state, EA signal must accord with this configuration.

Changing this configuration setting will drive the emulator into a reset state and will reset the memory mapping. Monitor address and vector address configurations will be set to their default.

4-6 Configuring the Emulator
Monitor base address?
This configuration allows you to specify the range of addresses that the monitor uses. The emulation monitor occupies 64K byte address space and the address of the monitor must be located on a 64K boundary. Valid address range is from 10000H through 0EF0000H.

Note
Changing this configuration setting will drive the emulator into a reset state and will reset the memory mapping. The vector address configuration will set to its default.

Enable emulation VBP?
This configuration allows you to specify whether or not the emulation VBP is used.

yes
The emulator supplies VBP value which determines the base address of the vector address. The emulator automatically initializes necessary vector entry to perform emulation tasks, emulation break, single stepping, and software breakpoint breaks.

no
VBP value is read from target system. The emulator does not do initializations for the vector entries to perform emulation tasks.

Note
Changing this configuration setting will drive the emulator into a reset state.

Vector base address?
This configuration allows you to specify the value for the VBP (Vector Base Pointer) to be calculated. Because this configuration is used whenever the emulator breaks into the monitor regardless "Enable emulation VBP?" configuration, you must specify address which accord with vector address.

Configuring the Emualtor 4-7
Note

Changing this configuration setting will drive the emulator into a reset state and will reset the memory mapping.

Initial CBP value?

This configuration allows you to specify the value of CBP (Current Bank Pointer) when the emulator breaks into the monitor from reset state. When emulation VBP is enabled and first 256 byte of vector area is mapped as emulation ROM/RAM, this configuration is ignored and CBP is initialized along with the vector entry.
Memory Configuration

The memory configuration questions allow you to select the monitor type, to select the location of the monitor, and to map memory. To access the memory configuration questions, you must answer "yes" to the following question.

Modify memory configuration?

Mapping Memory

The emulation memory consists of 256k, 1M, or 4Mbytes. You can define up to 7 memory ranges.

The memory mapper allows you to characterize memory locations. It allows you to specify whether a certain range of memory is present in the target system or whether you will be using emulation memory for that address range. You can also specify whether the target system memory is ROM or RAM, and you can specify that emulation memory be treated as ROM or RAM.

The internal RAM area (if you select the single chip mode, also internal ROM area) and emulation monitor area are mapped automatically. And you cannot delete these map terms. External I/O area cannot be mapped as emulation memory.

When you characterize memory ranges as emulation memory, note the following.

- When you characterize memory range which does not override 64K byte boundary as emulation memory, 64K byte is used.
  For example, when you characterize memory range (1000h thru 010ffh), 64K byte of emulation memory is used.

- When you characterize memory range which override N block of 64K byte as emulation memory, 64K x 2^M (2^M-1 < N <= 2^M) byte is used.
  For example, when you characterize memory range (0ff00h thru 200ffh) which overrides 3 block of 64K byte as emulation RAM, the 64K x 2^2 (2^1 < 3 <= 2^2:M=2) byte of emulation memory is used.
  For example, when 192K byte emulation memory is remained you cannot characterize memory range (80000h thru 0afffh), which is 192K byte and override 3 block of 64K byte, as

Configuring the Emulator 4-9
emulatoin RAM by one mapper term because the emulator needs 256K byte to map memory range(80000h thru 0afffh). In this case, you can characterize that memory range by two mapper term, the one is 128K byte(80000h thru 9ffffh) mapper term, the another is 64K byte(0a0000h thru 0afffh) mapper term.

---

**Note**

Target system accesses to emulation memory are not allowed. Target system devices that take control of the bus (for example, DMA controllers) cannot access emulation memory.

---

Blocks of memory can also be characterized as guarded memory. Guarded memory accesses will generate "break to monitor" requests. Writes to ROM will generate "break to monitor" requests if the "Enable breaks on writes to ROM?" configuration item is enabled (see the "Debug/Trace Configuration" section which follows).

**Determining the Locations to be Mapped**

Typically, assemblers generate relocatable files and linkers combine relocatable files to form the absolute file. The linker load map listing will show what locations your program will occupy in memory.
Emulator Pod Configuration

To access the emulator pod configuration questions, you must answer "yes" to the following question.

**Modify emulator pod configuration?**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>The emulator enables watch dog timer.</td>
</tr>
<tr>
<td>no</td>
<td>The emulator disables watch dog timer.</td>
</tr>
</tbody>
</table>

**Enable watch dog timer?**

(HP 64770A Only)

This question allows you to specify whether the watch dog timer is enabled or disabled when user's program running.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>The emulator enables watch dog timer.</td>
</tr>
<tr>
<td>no</td>
<td>The emulator disables watch dog timer.</td>
</tr>
</tbody>
</table>

**Respond to BUSRQ from target system?**

This configuration allows you to specify whether or not the emulator accepts BUSRQ (Bus Request) signal generated by the target system.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>The emulator accepts BUSRQ signal. When the BUSRQ is accepted, the emulator will respond as actual microprocessor.</td>
</tr>
<tr>
<td>no</td>
<td>The emulator ignore BUSRQ signal from target system completely.</td>
</tr>
</tbody>
</table>

**Respond to RESET from target system?**

The TLCS-9000 emulator can respond or ignore target system reset while running in user program or waiting for target system reset (refer to "run from reset" command in the Softkey Interface Reference manual). While running in monitor, the TLCS-9000 emulator ignores target system reset completely independent on this setting.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>Specify that, this is a default configuration, make the emulator to respond to reset from target system. In this configuration, emulator will accept reset and execute from reset vector as same manner as actual microprocessor after reset is inactivated.</td>
</tr>
</tbody>
</table>
The emulator ignores reset signal from target system completely, even while in foreground (executing user program).

**Note**
Changing this configuration option will drive the emulator into a reset state.

**Respond to interrupts?**
This question allows you to specify whether or not the emulation processor accepts interrupts.

- **yes**
  The emulator will respond to interrupt requests from target system (NMI and INT0-3 for HP 64770A, NMI and IREQ for 64770B) and an internal peripheral during user program execution.

- **no**
  The emulator will always ignore interrupt requests.

**Note**
When target interrupts signal is enabled, it is in effect while the emulator is running the target program. While the emulator is running monitor, NMI, INT0-7 (edge sense) for HP 64770A, IREQ for HP 64770B will be suspended until the emulator goes into user’s program.

**Target memory access size**
This configuration specifies the type of microprocessor cycles that are used by the monitor program to access target memory or I/O locations. When a command requests the monitor to read or write to target system memory or I/O, the monitor program will look at the access mode setting to determine whether byte or word instructions should be used.

- **bytes**
  Selecting the byte access mode specifies that the emulator will access target memory using byte cycles (one byte at a time).

4-12 Configuring the Emulator
Selecting the word access mode specifies that the emulator will access target memory using word cycles (one word at a time).

Selecting the any access mode specifies that the emulator will access target memory using a display/modify target memory command option. If option "words" is specified, access size will be set to "words". Other target memory commands such as "load" and "store" will use an access size of "bytes".

---

**Debug/Trace Configuration**

The debug/trace configuration questions allows you to specify breaks on writes to ROM, enable/disable the software breakpoints feature, and specify that the analyzer trace foreground/background execution. To access the debug/trace configuration questions, you must answer "yes" to the following question.

**Modify debug/trace options?**

This question allows you to specify that the emulator break to the monitor upon attempts to write to memory space mapped as ROM. The emulator will prevent the processor from actually writing to memory mapped as emulation ROM; however, they cannot prevent writes to target system RAM locations which are mapped as ROM, even though the write to ROM break is enabled.

**Break Processor on Write to ROM?**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>yes</strong></td>
<td>Causes the emulator to break into the emulation monitor whenever the user program attempts to write to a memory region mapped as ROM.</td>
</tr>
<tr>
<td><strong>no</strong></td>
<td>The emulator will not break to the monitor upon a write to ROM. The emulator will not modify the memory location if it is in emulation ROM.</td>
</tr>
</tbody>
</table>

Configuring the Emulator 4-13
The *wrrom* trace command status option allows you to use "write to ROM" cycles as trigger and storage qualifiers. For example, you could use the following command to trace about a write to ROM:

```
trace about status wrrom <RETURN>
```

**Trace monitor or user program operation?**

This question allows you to specify whether the analyzer trace only user program emulation processor cycles, only monitor cycles, or both monitor or user program cycles.

**user**

Specifies that the analyzer trace only user program cycles. This option is specified by the default emulator configuration.

**monitor**

Specifies that the analyzer trace only emulation monitor cycles. (This is rarely a useful setting.)

**both**

Specifies that the analyzer trace both user program and emulation monitor cycles. You may wish to specify this option so that all emulation processor cycles may be viewed in the trace display.

**Emulation analyzer speed? (HP 64770A Only)**

This question allows you to specify the emulation processor clock speed. The analyzer capabilities of time and state count are affected by the processor clock speed. If you use 64794A/C/D Deep emulation analyzer, the trace state and time counter qualifiers can be used regardless of clock speed. You must answer this question, when you use HP 64770A emulator with HP 64704A emulation bus analyzer.

**slow**

Specifies the processor clock speed is less than or equal to 16MHz. Both state and time counting are available.

**fast**

Specifies the processor clock speed is greater than 16MHz. Only state counting are available.

4-14 Configuring the Emulator
Simulated I/O Configuration

The simulated I/O feature and configuration options are described in the Simulated I/O reference manual.

Interactive Measurement Configuration

The interactive measurement configuration questions are described in the chapter on coordinated measurements in the Softkey Interface Reference manual. Examples of coordinated measurements that can be performed between the emulator and the emulation analyzer are found in the "Using the Emulator" chapter.

Saving a Configuration

The last configuration question allows you to save the previous configuration specifications in a file which can be loaded back into the emulator at a later time.

Configuration file name? <FILE>

The name of the last configuration file is shown, or no filename is shown if you are modifying the default emulator configuration.

If you press <RETURN> without specifying a filename, the configuration is saved to a temporary file. This file is deleted when you exit the Softkey Interface with the "end release_system" command.

When you specify a filename, the configuration will be saved to two files; the filename specified with extensions of ".EA" and ".EB". The file with the ".EA" extension is the "source" copy of the file, and the file with the ".EB" extension is the "binary" or loadable copy of the file.

Ending out of emulation (with the "end" command) saves the current configuration, including the name of the most recently loaded configuration file, into a "continue" file. The continue file is not normally accessed.
Loading a Configuration

Configuration files which have been previously saved may be loaded with the following Softkey Interface command.

```
load configuration <FILE> <RETURN>
```

This feature is especially useful after you have exited the Softkey Interface with the "end release_system" command; it saves you from having to modify the default configuration and answer all the questions again. To reload the current configuration, you can enter the following command.

```
load configuration <RETURN>
```
Using the Emulator

Introduction

The "Getting Started" chapter shows you how to use the basic
This chapter discuss:

- Register names and classes
- Hardware breakpoint
- Vector area setting
- Analyzer topics
  - Specifying address and status for trigger or store condition
  - Specifying data for trigger or store condition
  - Specifying execute address for trigger or store condition
- Features available via "pod_command"

This chapter shows you how to:

- Store the contents of memory into absolute files
- Make coordinated measurements
### REGISTER CLASS and NAME

**Summary** 70732 register designator. All available register class names and register names are listed below.

<table>
<thead>
<tr>
<th>&lt;REG_CLASS&gt;</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>(All basic registers)</em></td>
<td></td>
</tr>
<tr>
<td>PC</td>
<td>BASIC registers.</td>
</tr>
<tr>
<td>RW0</td>
<td></td>
</tr>
<tr>
<td>RW1</td>
<td></td>
</tr>
<tr>
<td>RW2</td>
<td></td>
</tr>
<tr>
<td>RW3</td>
<td></td>
</tr>
<tr>
<td>RW4</td>
<td></td>
</tr>
<tr>
<td>RW5</td>
<td></td>
</tr>
<tr>
<td>RW6</td>
<td></td>
</tr>
<tr>
<td>RW7</td>
<td></td>
</tr>
<tr>
<td>RW8</td>
<td></td>
</tr>
<tr>
<td>RW9</td>
<td></td>
</tr>
<tr>
<td>RW10</td>
<td></td>
</tr>
<tr>
<td>RW11</td>
<td></td>
</tr>
<tr>
<td>RW12</td>
<td></td>
</tr>
<tr>
<td>RW13</td>
<td></td>
</tr>
<tr>
<td>RW14</td>
<td></td>
</tr>
<tr>
<td>RW15</td>
<td></td>
</tr>
<tr>
<td>ISP</td>
<td></td>
</tr>
<tr>
<td>USP</td>
<td></td>
</tr>
<tr>
<td>FP</td>
<td></td>
</tr>
<tr>
<td>CBP</td>
<td></td>
</tr>
<tr>
<td>PBP</td>
<td></td>
</tr>
<tr>
<td>PSW</td>
<td></td>
</tr>
</tbody>
</table>

5-2 Using the Emulator
### PBANK (Previous bank registers)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPC</td>
<td>Saved PC</td>
</tr>
<tr>
<td>PPSW</td>
<td>Saved PSW</td>
</tr>
<tr>
<td>PPBP</td>
<td>Saved PBP</td>
</tr>
<tr>
<td>PR0</td>
<td>pw0 on previous bank</td>
</tr>
<tr>
<td>PR1</td>
<td>pw1 on previous bank</td>
</tr>
<tr>
<td>PR2</td>
<td>pw2 on previous bank</td>
</tr>
<tr>
<td>PR3</td>
<td>pw3 on previous bank</td>
</tr>
<tr>
<td>PR4</td>
<td>pw4 on previous bank</td>
</tr>
<tr>
<td>PR5</td>
<td>pw5 on previous bank</td>
</tr>
<tr>
<td>PR6</td>
<td>pw6 on previous bank</td>
</tr>
<tr>
<td>PR7</td>
<td>pw7 on previous bank</td>
</tr>
<tr>
<td>PR8</td>
<td>pw8 on previous bank</td>
</tr>
<tr>
<td>PR9</td>
<td>pw9 on previous bank</td>
</tr>
<tr>
<td>PR10</td>
<td>pw10 on previous bank</td>
</tr>
<tr>
<td>PR11</td>
<td>pw11 on previous bank</td>
</tr>
<tr>
<td>PR12</td>
<td>pw12 on previous bank</td>
</tr>
<tr>
<td>PR13</td>
<td>pw13 on previous bank</td>
</tr>
<tr>
<td>PR14</td>
<td>pw14 on previous bank</td>
</tr>
<tr>
<td>PR15</td>
<td>pw15 on previous bank</td>
</tr>
</tbody>
</table>

### SYS (System control registers) (HP 64770A Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WDMOD</td>
<td>Watch dog timer mode</td>
</tr>
<tr>
<td>WDCR</td>
<td>Watch dog timer control (Write Only)</td>
</tr>
<tr>
<td>CH0CR</td>
<td>Memory controller channel 0</td>
</tr>
<tr>
<td>CH1CR</td>
<td>Memory controller channel 1</td>
</tr>
<tr>
<td>CH2CR</td>
<td>Memory controller channel 2</td>
</tr>
<tr>
<td>CH3CR</td>
<td>Memory controller channel 3</td>
</tr>
<tr>
<td>REFHREG</td>
<td>Refresh control</td>
</tr>
</tbody>
</table>

### SYS (System control registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMR</td>
<td>Operation mode</td>
</tr>
<tr>
<td>PDMR</td>
<td>Power down mode</td>
</tr>
<tr>
<td>STBYMD</td>
<td>Stand-by mode</td>
</tr>
<tr>
<td>CH0CR</td>
<td>Memory controller channel 0</td>
</tr>
<tr>
<td>CH1CR</td>
<td>Memory controller channel 1</td>
</tr>
<tr>
<td>CH2CR</td>
<td>Memory controller channel 2</td>
</tr>
<tr>
<td>CH3CR</td>
<td>Memory controller channel 3</td>
</tr>
<tr>
<td>REFHREG</td>
<td>Refresh control</td>
</tr>
</tbody>
</table>

**Using the Emulator 5-3**
### TMR (Timer registers) (HP 64770A Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUN0</td>
<td>Timer control (TRUN0123)</td>
<td></td>
</tr>
<tr>
<td>TRUN4</td>
<td>Timer control (TRUN4567)</td>
<td></td>
</tr>
<tr>
<td>TRDC0</td>
<td>Double buffer control (TRDC0123)</td>
<td></td>
</tr>
<tr>
<td>TRDC4</td>
<td>Double buffer control (TRDC4567)</td>
<td></td>
</tr>
<tr>
<td>TFFCR0</td>
<td>Timer flip-flop control (TFFCR0123)</td>
<td></td>
</tr>
<tr>
<td>TFFCR4</td>
<td>Timer flip-flop control (TFFCR4567)</td>
<td></td>
</tr>
<tr>
<td>T01MOD</td>
<td>Timer source clk and mode</td>
<td>Write Only</td>
</tr>
<tr>
<td>T23MOD</td>
<td>Timer source clk and mode</td>
<td>Write Only</td>
</tr>
<tr>
<td>T45MOD</td>
<td>Timer source clk and mode</td>
<td>Write Only</td>
</tr>
<tr>
<td>T67MOD</td>
<td>Timer source clk and mode</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG0</td>
<td>Timer register 0</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG1</td>
<td>Timer register 1</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG2</td>
<td>Timer register 2</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG3</td>
<td>Timer register 3</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG4</td>
<td>Timer register 4</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG5</td>
<td>Timer register 5</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG6</td>
<td>Timer register 6</td>
<td>Write Only</td>
</tr>
<tr>
<td>TREG7</td>
<td>Timer register 7</td>
<td>Write Only</td>
</tr>
<tr>
<td>TT0RUN</td>
<td>Timer control 0</td>
<td></td>
</tr>
<tr>
<td>TT1RUN</td>
<td>Timer control 1</td>
<td></td>
</tr>
<tr>
<td>TT0MOD</td>
<td>Timer source clk and mode</td>
<td></td>
</tr>
<tr>
<td>TT1MOD</td>
<td>Timer source clk and mode</td>
<td></td>
</tr>
<tr>
<td>TT0FFCR</td>
<td>Timer flip-flop control</td>
<td></td>
</tr>
<tr>
<td>TT1FFCR</td>
<td>Timer flip-flop control</td>
<td></td>
</tr>
<tr>
<td>TTREG0</td>
<td>Timer register 0</td>
<td>Write Only</td>
</tr>
<tr>
<td>TTREG1</td>
<td>Timer register 1</td>
<td>Write Only</td>
</tr>
<tr>
<td>TTREG2</td>
<td>Timer register 2</td>
<td>Write Only</td>
</tr>
<tr>
<td>TTREG3</td>
<td>Timer register 3</td>
<td>Write Only</td>
</tr>
<tr>
<td>CAP1</td>
<td>Capture register 1</td>
<td>Read Only</td>
</tr>
<tr>
<td>CAP2</td>
<td>Capture register 2</td>
<td>Read Only</td>
</tr>
<tr>
<td>CAP3</td>
<td>Capture register 3</td>
<td>Read Only</td>
</tr>
<tr>
<td>CAP4</td>
<td>Capture register 4</td>
<td>Read Only</td>
</tr>
</tbody>
</table>

5-4 Using the Emulator
<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTR</td>
<td>General timer</td>
</tr>
<tr>
<td>CPRS0</td>
<td>Compare reg for &quot;Set ch0&quot;</td>
</tr>
<tr>
<td>CPRS1</td>
<td>Compare reg for &quot;Set ch1&quot;</td>
</tr>
<tr>
<td>CPRS2</td>
<td>Compare reg for &quot;Set ch2&quot;</td>
</tr>
<tr>
<td>CPRS3</td>
<td>Compare reg for &quot;Set ch3&quot;</td>
</tr>
<tr>
<td>CPRS4</td>
<td>Compare reg for &quot;Set ch4&quot;</td>
</tr>
<tr>
<td>CPRS5</td>
<td>Compare reg for &quot;Set ch5&quot;</td>
</tr>
<tr>
<td>CPRS6</td>
<td>Compare reg for &quot;Set ch6&quot;</td>
</tr>
<tr>
<td>CPRS7</td>
<td>Compare reg for &quot;Set ch7&quot;</td>
</tr>
<tr>
<td>CPRR0</td>
<td>Compare reg for &quot;Reset ch0&quot;</td>
</tr>
<tr>
<td>CPRR1</td>
<td>Compare reg for &quot;Reset ch1&quot;</td>
</tr>
<tr>
<td>CPRR2</td>
<td>Compare reg for &quot;Reset ch2&quot;</td>
</tr>
<tr>
<td>CPRR3</td>
<td>Compare reg for &quot;Reset ch3&quot;</td>
</tr>
<tr>
<td>CPRR4</td>
<td>Compare reg for &quot;Reset ch4&quot;</td>
</tr>
<tr>
<td>CPRR5</td>
<td>Compare reg for &quot;Reset ch5&quot;</td>
</tr>
<tr>
<td>CPRR6</td>
<td>Compare reg for &quot;Reset ch6&quot;</td>
</tr>
<tr>
<td>CPRR7</td>
<td>Compare reg for &quot;Reset ch7&quot;</td>
</tr>
<tr>
<td>DOMR1</td>
<td>Digital output mode</td>
</tr>
<tr>
<td>DOCR</td>
<td>Digital output control</td>
</tr>
<tr>
<td>DOR1</td>
<td>Digital out</td>
</tr>
<tr>
<td>LGTO</td>
<td>Output level of GTO</td>
</tr>
<tr>
<td>GTOEN</td>
<td>GTO enable</td>
</tr>
</tbody>
</table>

Using the Emulator 5-5
### GTI (General input timer registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPCL0</td>
<td>Pulse counter latch 0</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPCL1</td>
<td>Pulse counter latch 1</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPCL2</td>
<td>Pulse counter latch 2</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPCL3</td>
<td>Pulse counter latch 3</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA0P</td>
<td>GTIA positive edge 0</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA1P</td>
<td>GTIA positive edge 1</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA2P</td>
<td>GTIA positive edge 2</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA3P</td>
<td>GTIA positive edge 3</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA0N</td>
<td>GTIA negative edge 0</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA1N</td>
<td>GTIA negative edge 1</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA2N</td>
<td>GTIA negative edge 2</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTA3N</td>
<td>GTIA negative edge 3</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTB0</td>
<td>GTIB edge 0</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTB1</td>
<td>GTIB edge 1</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTB2</td>
<td>GTIB edge 2</td>
<td>Read Only</td>
</tr>
<tr>
<td>GTB3</td>
<td>GTIB edge 3</td>
<td>Read Only</td>
</tr>
</tbody>
</table>

### POUT (Pulse timer output registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIOC</td>
<td>TIO control</td>
<td>Read Only</td>
</tr>
<tr>
<td>LPOUT</td>
<td>Output level of POUT</td>
<td>Read Only</td>
</tr>
<tr>
<td>DOMR2</td>
<td>Digital output mode</td>
<td>Read Only</td>
</tr>
<tr>
<td>DOR2</td>
<td>Digital out</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD0</td>
<td>Compare register for Pout 0</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD1</td>
<td>Compare register for Pout 1</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD2</td>
<td>Compare register for Pout 2</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD3</td>
<td>Compare register for Pout 3</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD4</td>
<td>Compare register for Pout 4</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD5</td>
<td>Compare register for Pout 5</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD6</td>
<td>Compare register for Pout 6</td>
<td>Read Only</td>
</tr>
<tr>
<td>CPRD7</td>
<td>Compare register for Pout 7</td>
<td>Read Only</td>
</tr>
</tbody>
</table>

5-6 Using the Emulator
POC (Pulse output down-counter registers) (HP 64770B Only)

CPOC0: Pulse output counter of ch0
CPOC1: Pulse output counter of ch1
CPOC2: Pulse output counter of ch2
CPOC3: Pulse output counter of ch3
CPOC4: Pulse output counter of ch4
CPOC5: Pulse output counter of ch5
CPOC6: Pulse output counter of ch6
CPOC7: Pulse output counter of ch7

SC (Serial communication registers) (HP 64770A Only)

SC0CR: Serial channel 0 control
SC0MOD: Serial channel 0 mode
BR0CR: Serial channel 0 baud rate control
SC0BUF: Serial channel 0 buffer
SC1CR: Serial channel 1 control
SC1MOD: Serial channel 1 mode
BR1CR: Serial channel 1 baud rate control
SC1BUF: Serial channel 1 buffer
SC2CR: Serial channel 2 control
SC2MOD: Serial channel 2 mode
BR2CR: Serial channel 2 baud rate control
SC2BUF: Serial channel 2 buffer
ODE: Port 8 open-drain enable

Using the Emulator 5-7
### SCI (Serial interface registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCATB</td>
<td>SCIA transmit buffer</td>
<td>Write Only</td>
</tr>
<tr>
<td>SCARB</td>
<td>SCIA receive buffer</td>
<td>Read Only</td>
</tr>
<tr>
<td>SCAMR</td>
<td>SCIA mode</td>
<td></td>
</tr>
<tr>
<td>SCASR</td>
<td>SCIA status</td>
<td>Read Only</td>
</tr>
<tr>
<td>SCACR</td>
<td>SCIA control</td>
<td></td>
</tr>
<tr>
<td>SC2TB</td>
<td>SC12 transmit buffer</td>
<td>Write Only</td>
</tr>
<tr>
<td>SC2RB</td>
<td>SC12 receive buffer</td>
<td>Read Only</td>
</tr>
<tr>
<td>SC2MR</td>
<td>SC12 mode</td>
<td></td>
</tr>
<tr>
<td>SC2SR</td>
<td>SC12 status</td>
<td>Read Only</td>
</tr>
<tr>
<td>SC2CR</td>
<td>SC12 control</td>
<td></td>
</tr>
<tr>
<td>SCB2TB</td>
<td>SCIB transmit buffer</td>
<td>Write Only</td>
</tr>
<tr>
<td>SCB2RB</td>
<td>SCIB receive buffer</td>
<td>Read Only</td>
</tr>
<tr>
<td>SCB2MR</td>
<td>SCIB mode</td>
<td></td>
</tr>
<tr>
<td>SCB2SR</td>
<td>SCIB status</td>
<td>Read Only</td>
</tr>
<tr>
<td>SCB2CR</td>
<td>SCIB control</td>
<td></td>
</tr>
</tbody>
</table>

### SEI (Expansion serial interface registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCR</td>
<td>Asynchronous mode command</td>
<td>Write Only</td>
</tr>
<tr>
<td>ASBF</td>
<td>Asynchronous mode buffer</td>
<td>Read Only</td>
</tr>
<tr>
<td>AKCR</td>
<td>Synchronous mode command</td>
<td>Write Only</td>
</tr>
<tr>
<td>SKBF</td>
<td>Synchronous mode buffer</td>
<td>Read Only</td>
</tr>
<tr>
<td>SE2CR</td>
<td>SEI2 control &amp; status</td>
<td></td>
</tr>
<tr>
<td>SE3BO</td>
<td>SEI3 buffer register out</td>
<td>Write Only</td>
</tr>
<tr>
<td>SE3BI</td>
<td>SEI3 buffer register in</td>
<td>Read Only</td>
</tr>
<tr>
<td>SE3SFO</td>
<td>SEI3 shift register out</td>
<td>Write Only</td>
</tr>
<tr>
<td>SE3SFI</td>
<td>SEI3 shift register in</td>
<td>Read Only</td>
</tr>
<tr>
<td>SE3CR</td>
<td>SEI3 control</td>
<td></td>
</tr>
<tr>
<td>SESR</td>
<td>SEI shif register</td>
<td></td>
</tr>
<tr>
<td>SECR</td>
<td>SEI control &amp; status</td>
<td></td>
</tr>
</tbody>
</table>

### SMP (Serial monitor port registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMISR</td>
<td>SMP input shift register</td>
<td>Read Only</td>
</tr>
<tr>
<td>SMOSR</td>
<td>SMP output shift register</td>
<td>Write Only</td>
</tr>
<tr>
<td>SMFULL</td>
<td>SMP input full register</td>
<td></td>
</tr>
</tbody>
</table>

5-8 Using the Emulator
AD (A/D converter registers) (HP 64770A Only)

ADMOD  A/D converter mode
ADCCS  ADC channel selector
ADREG04  AD result 04  (Read Only)
ADREG15  AD result 15  (Read Only)
ADREG26  AD result 26  (Read Only)
ADREG37  AD result 37  (Read Only)

DMA (DMA controller registers) (HP 64770B Only)

MAR0  Memory address 0
DTCR0  Data transfer count 0
MAR1  Memory address 1
DTCR1  Data transfer count 1
MAR2  Memory address 2
DTCR2  Data transfer count 2
MAR3  Memory address 3
DTCR3  Data transfer count 3
MAR4  Memory address 4
DTCR4  Data transfer count 4
MAR5  Memory address 5
DTCR5  Data transfer count 5
CHSR0  Channel status 0  (Read Only)
CHSR1  Channel status 1  (Read Only)
CHSR2  Channel status 2  (Read Only)
INT (Interrupt control registers) (HP 64770A Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTE0</td>
<td>Interrupt enable 0</td>
</tr>
<tr>
<td>INTE1</td>
<td>Interrupt enable 1</td>
</tr>
<tr>
<td>INTE2</td>
<td>Interrupt enable 2</td>
</tr>
<tr>
<td>INTE3</td>
<td>Interrupt enable 3</td>
</tr>
<tr>
<td>INTE4</td>
<td>Interrupt enable 4</td>
</tr>
<tr>
<td>INTE5</td>
<td>Interrupt enable 5</td>
</tr>
<tr>
<td>INTE6</td>
<td>Interrupt enable 6</td>
</tr>
<tr>
<td>INTE7</td>
<td>Interrupt enable 7</td>
</tr>
<tr>
<td>INTE0</td>
<td>Interrupt enable 8 bit timer 0</td>
</tr>
<tr>
<td>INTE1</td>
<td>Interrupt enable 8 bit timer 1</td>
</tr>
<tr>
<td>INTE2</td>
<td>Interrupt enable 8 bit timer 2</td>
</tr>
<tr>
<td>INTE3</td>
<td>Interrupt enable 8 bit timer 3</td>
</tr>
<tr>
<td>INTE4</td>
<td>Interrupt enable 8 bit timer 4</td>
</tr>
<tr>
<td>INTE5</td>
<td>Interrupt enable 8 bit timer 5</td>
</tr>
<tr>
<td>INTE6</td>
<td>Interrupt enable 8 bit timer 6</td>
</tr>
<tr>
<td>INTE7</td>
<td>Interrupt enable 8 bit timer 7</td>
</tr>
<tr>
<td>INTET0</td>
<td>Interrupt enable 16 bit timer TTREG0</td>
</tr>
<tr>
<td>INTET1</td>
<td>Interrupt enable 16 bit timer TTREG1</td>
</tr>
<tr>
<td>INTET2</td>
<td>Interrupt enable 16 bit timer TTREG2</td>
</tr>
<tr>
<td>INTET3</td>
<td>Interrupt enable 16 bit timer TTREG3</td>
</tr>
<tr>
<td>INTE0R</td>
<td>Interrupt enable serial 0 receive</td>
</tr>
<tr>
<td>INTE0T</td>
<td>Interrupt enable serial 0 transmit</td>
</tr>
<tr>
<td>INTE1R</td>
<td>Interrupt enable serial 1 receive</td>
</tr>
<tr>
<td>INTE1T</td>
<td>Interrupt enable serial 1 transmit</td>
</tr>
<tr>
<td>INTE2R</td>
<td>Interrupt enable serial 2 receive</td>
</tr>
<tr>
<td>INTE2T</td>
<td>Interrupt enable serial 2 transmit</td>
</tr>
<tr>
<td>INTEAD</td>
<td>Interrupt enable A/D</td>
</tr>
<tr>
<td>INTE TASK</td>
<td>Interrupt enable TASK</td>
</tr>
<tr>
<td>INMIMC</td>
<td>Interrupt NMI input mode control</td>
</tr>
</tbody>
</table>
PIC (Interrupt control registers) (HP 64770B Only)

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTICR</td>
<td>General timer interrupt control</td>
</tr>
<tr>
<td>TIICR0</td>
<td>GTI interrupt control 0</td>
</tr>
<tr>
<td>TIICR1</td>
<td>GTI interrupt control 1</td>
</tr>
<tr>
<td>TIICR2</td>
<td>GTI interrupt control 2</td>
</tr>
<tr>
<td>TOICR0</td>
<td>GTO interrupt control 0</td>
</tr>
<tr>
<td>TOICR1</td>
<td>GTO interrupt control 1</td>
</tr>
<tr>
<td>TOICR2</td>
<td>GTO interrupt control 2</td>
</tr>
<tr>
<td>TOICR3</td>
<td>GTO interrupt control 3</td>
</tr>
<tr>
<td>POICR0</td>
<td>POUT interrupt control 0</td>
</tr>
<tr>
<td>POICR1</td>
<td>POUT interrupt control 1</td>
</tr>
<tr>
<td>SIOICR0</td>
<td>SCI interrupt control</td>
</tr>
<tr>
<td>DMAICR0</td>
<td>SCI2 interrupt control</td>
</tr>
<tr>
<td>DMAICR1</td>
<td>SCI3 interrupt control</td>
</tr>
<tr>
<td>SWICR0</td>
<td>SOFTWARE interrupt control 0</td>
</tr>
<tr>
<td>SWICR1</td>
<td>SOFTWARE interrupt control 1</td>
</tr>
<tr>
<td>SWICR2</td>
<td>SOFTWARE interrupt control 2</td>
</tr>
<tr>
<td>NMIRQ</td>
<td>NMI interrupt request flag</td>
</tr>
<tr>
<td>GTIRQ</td>
<td>GT interrupt request flag</td>
</tr>
<tr>
<td>TIIRQ</td>
<td>Timer input interrupt request flag</td>
</tr>
<tr>
<td>TOISRQ</td>
<td>Timer output set interrupt request</td>
</tr>
<tr>
<td>TOIRRQ</td>
<td>Timer output reset interrupt request</td>
</tr>
<tr>
<td>POIRQ</td>
<td>Pout interrupt flag</td>
</tr>
<tr>
<td>DMAIRQ</td>
<td>DMA interrupt flag</td>
</tr>
<tr>
<td>SWIRQ</td>
<td>SWI interrupt flag</td>
</tr>
</tbody>
</table>

Using the Emulator 5-11
PRT (Port registers) (HP 64770A Only)

PT0  Port 0
PT1  Port 1
PT2  Port 2
PT3  Port 3
PT4  Port 4
PT5  Port 5
PT6  Port 6
PT7  Port 7
PT8  Port 8
PT9  Port 9
PTA  Port A
PTB  Port B
PTC  Port C
P0CR Port 0 control
P0FC Port 0 function
P1CR Port 1 control
P1FC Port 1 function
P2CR Port 2 control
P2FC Port 2 function
P3CR Port 3 control
P3FC Port 3 function
P4CR Port 4 control
P4FC Port 4 function
P5CR Port 5 control
P5FC Port 5 function
P6CR Port 6 control
P6FC Port 6 function
P7CR Port 7 control
P7FC Port 7 function
P8CR Port 8 control
P8FC Port 8 function
P9CR Port 9 control
P9FC Port 9 function
PACR Port A control
PAFC Port A function
PBCR Port B control
PBFC Port B function

5-12 Using the Emulator
PRT (Port registers) (HP 64770B Only)

| P0  | Port 0 data |
| P1  | Port 1 data |
| P2  | Port 2 data |
| P3  | Port 3 data |
| P4  | Port 4 data |
| P5  | Port 5 data |
| P6  | Port 6 data |
| P9  | Port 9 data |
| PJ  | Port J data |
| PF  | Port F data |
| PG  | Port G data |
| PM  | Port M data |
| PH  | Port H data |
| PS  | Port S data |
| P0CR | Port 0 control |
| P0FC | Port 0 function |
| P1CR | Port 1 control |
| P1FC | Port 1 function |
| P2CR | Port 2 control |
| P2FC | Port 2 function |
| P3CR | Port 3 control |
| P3FC | Port 3 function |
| P4CR | Port 4 control |
| P4FC | Port 4 function |
| P5CR | Port 5 control |
| P5FC | Port 5 function |
| P6CR | Port 6 control |
| P6FC | Port 6 function |
| P9CR | Port 9 control |
| PJCR | Port J control |
| PFCR | Port F control |
| PGCR | Port G control |
| PMCR | Port M control |
| PHCR | Port H control |
| PSCR | Port S control |

Using the Emulator 5-13
<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFSR0</td>
<td>Port function select 0</td>
</tr>
<tr>
<td>PFSR1</td>
<td>Port function select 1</td>
</tr>
<tr>
<td>PFSR2</td>
<td>Port function select 2</td>
</tr>
<tr>
<td>PFSR3</td>
<td>Port function select 3</td>
</tr>
<tr>
<td>PFSR4</td>
<td>Port function select 4 (Write Only)</td>
</tr>
<tr>
<td>PFSR5</td>
<td>Port function select 5</td>
</tr>
<tr>
<td>PFSR6</td>
<td>Port function select 6</td>
</tr>
</tbody>
</table>

**OTHER (Other registers)**

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RB0</td>
<td>RB0</td>
</tr>
<tr>
<td>RB1</td>
<td>RB1</td>
</tr>
<tr>
<td>RB2</td>
<td>RB2</td>
</tr>
<tr>
<td>RB3</td>
<td>RB3</td>
</tr>
<tr>
<td>RB4</td>
<td>RB4</td>
</tr>
<tr>
<td>RB5</td>
<td>RB5</td>
</tr>
<tr>
<td>RB6</td>
<td>RB6</td>
</tr>
<tr>
<td>RB7</td>
<td>RB7</td>
</tr>
<tr>
<td>RB8</td>
<td>RB8</td>
</tr>
<tr>
<td>RB9</td>
<td>RB9</td>
</tr>
<tr>
<td>RB10</td>
<td>RB10</td>
</tr>
<tr>
<td>RB11</td>
<td>RB11</td>
</tr>
<tr>
<td>RB12</td>
<td>RB12</td>
</tr>
<tr>
<td>RB13</td>
<td>RB13</td>
</tr>
<tr>
<td>RB14</td>
<td>RB14</td>
</tr>
<tr>
<td>RB15</td>
<td>RB15</td>
</tr>
<tr>
<td>RD0</td>
<td>RD0</td>
</tr>
<tr>
<td>RD1</td>
<td>RD1</td>
</tr>
<tr>
<td>RD2</td>
<td>RD2</td>
</tr>
<tr>
<td>RD3</td>
<td>RD3</td>
</tr>
<tr>
<td>RD4</td>
<td>RD4</td>
</tr>
<tr>
<td>RD5</td>
<td>RD5</td>
</tr>
<tr>
<td>RD6</td>
<td>RD6</td>
</tr>
<tr>
<td>RD7</td>
<td>RD7</td>
</tr>
<tr>
<td>RD8</td>
<td>RD8</td>
</tr>
<tr>
<td>RD9</td>
<td>RD9</td>
</tr>
<tr>
<td>RD10</td>
<td>RD10</td>
</tr>
<tr>
<td>RD11</td>
<td>RD11</td>
</tr>
<tr>
<td>RD12</td>
<td>RD12</td>
</tr>
<tr>
<td>RD13</td>
<td>RD13</td>
</tr>
<tr>
<td>RD14</td>
<td>RD14</td>
</tr>
<tr>
<td>USPL</td>
<td>lower 16 bits of USP</td>
</tr>
<tr>
<td>USPH</td>
<td>upper 16 bits of USP</td>
</tr>
<tr>
<td>FPL</td>
<td>lower 16 bits of FP</td>
</tr>
<tr>
<td>FPH</td>
<td>upper 16 bits of FP</td>
</tr>
</tbody>
</table>

5-14 Using the Emulator
Hardware
Breakpoints

The analyzer may generate a break request to the emulation processor. To break when the analyzer trigger condition is satisfied, use the "break_on_trigger" trace option.

Additionally, you can see the program states before the breakpoint in trace listing. Specify the trigger position at the end of trace listing by using "before" option.

When the trigger condition is found, the emulator execution will break into the emulation monitor. Then you can also see the trace listing mentioned above, enter the following commands.

```
trace before <QUALIFIER>
broke_on_trigger<RETURN>
```

Without the trigger condition, the trigger will never occur and will never break.

Vector Area
Setting

TLCS-9000 microprocessor has vector area(2k bytes). TLCS-9000 emulator uses three vector entry in vector area to realize the following emulator features.

- Break
- Single-Step
- Software Break Point

**Single Chip Mode**

If you use the TLCS-9000 emulator in single chip mode, you do not need to set the vector entry since the emulator set automatically. The values of PC, PSW, and CBP are set bye vector entry when the emulator breaks into the monitor from reset state.

**External Bus Mode**

If you use the TLCS-9000 emulator in external bus mode, the way of the emulator’s operations differ according to "Enable emulation VBP?" configuration and memory mapping.

Using the Emulator 5-15
"Enable emulation VBP? yes"

The emulator read Vector Base Pointer(VBP) from emulation VBP. When the emulator breaks into the monitor from reset state, the value of emulation VBP is specified by "Vector base address?" configuration.

If vector area are mapped as emulation memory, the emulator sets the vector entry when the emulator breaks into the monitor from reset state. When the emulator breaks into the monitor from reset state, the values of PC, PSW, and CBP are set by vector entry.

If vector area are mapped as target memory, the emulator uses copy of vector area. The emulator copies data of vector area when the emulator breaks into the monitor from reset state, and then sets the vector entry. When the emulator breaks into the monitor from reset, the value of PC, and PSW are set by vector entry and the value of CBP is specified by "Initial CBP value?" configuration.

"Enable emulation VBP? no"

In this case, the emulator does not set the vector entry. So you must set up the vector entry to realize the emulator features. If you do not set the vector entry, the emulator can not operate correctly.

Even if you specify that "Enable emulation VBP? no", the value of PC, PSW, and CBP are specified in the same way as you specify that "Enable emulation VBP? yes" when the emulator breaks into the monitor from reset state.

Set the vector area as following.

<table>
<thead>
<tr>
<th>Vector number</th>
<th>Offset</th>
<th>value</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>60H</td>
<td>0000H</td>
<td>Break</td>
</tr>
<tr>
<td></td>
<td>62H</td>
<td>0202H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>64H</td>
<td>00xxH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>66H</td>
<td>0000H</td>
<td></td>
</tr>
</tbody>
</table>

5-16 Using the Emulator
Analyzer Topics

**Specifying Address and Status for Trigger or Store Condition**

The analyzer captures the actual bus states and execute states by exclusive bus. In some case, bus state and execute state are captured simultaneously. To specify actual bus states for trigger or store condition by "address", "status", and "data", you should add "status bus" condition to trigger/store condition as following.

```
trace after address 1000h status bus<RETURN>
```

```
trace after status write and bus
```

**Specifying Data for Trigger or Store Condition**

The analyzer captures the actual bus states of the TLCS-9000 microprocessor. When you specify a data in the analyzer trigger or store condition, the ways of analyzer data specification differ according to the address and the data size.

To trigger analyzer when the TLCS-9000 microprocessor accesses the byte data 12h at address 1000h(even address), enter the following,

```
trace after address 1000h data 0x12h status bus and byte
```

To trigger analyzer when the TLCS-9000 microprocessor accesses the byte data 12h at address 1001h(odd address), enter the following,
To trigger analyzer when the TLCS-9000 microprocessor accesses the word data 1234h at address 1001h (odd address), the data bus activity of cycles will be as follows.

<table>
<thead>
<tr>
<th>Sequencer level</th>
<th>Address bus</th>
<th>Data bus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1001</td>
<td>34xx</td>
</tr>
<tr>
<td>2</td>
<td>1002</td>
<td>xx12</td>
</tr>
</tbody>
</table>

In this case, you need to use the analyzer sequential trigger capabilities. We do not describe the detail about the sequential trigger feature. Only how to trigger the analyzer at this example is described. To specify the condition, enter:

```
trace find_sequence 1001h data 34xxh status bus trigger after 1002h data 0xx12h status bus <RETURN>
```

### Specifying Execute Address for Trigger or Store Condition

To specify "eaddr" for trigger or store condition, you must specify even addresses as execute address. To trigger analyzer when TLCS-9000 microprocessor executes instruction at address 2000h, enter the following.

```
trace after eaddr 2000h <RETURN>
```
Several emulation features available in the Terminal Interface but not in the Softkey Interface may be accessed via the following emulation commands.

```
display pod_command <RETURN>
pod_command ‘<Terminal Interface command>’ <RETURN>
```

Some of the most notable Terminal Interface features not available in the Softkey Interface are:

- Searching memory for strings or numeric expressions.
- Sequencing in the analyzer.

Refer to your Terminal Interface documentation for information on how to perform these tasks.

**Note**

Be careful when using the "pod_command". The Softkey Interface, and the configuration files in particular, assume that the configuration of the HP 64700 pod is NOT changed except by the Softkey Interface. Be aware that what you see in "modify configuration" will NOT reflect the HP 64700 pod's configuration if you change the pod's configuration with this command. Also, commands which affect the communications channel should NOT be used at all. Other commands may confuse the protocol depending upon how they are used. The following commands are not recommended for use with "pod_command":

- **stty, po, xp** - Do not use, will change channel operation and hang.
- **echo, mac** - Usage may confuse the protocol in use on the channel.
- **wait** - Do not use, will tie up the pod, blocking access.
- **init, pv** - Will reset pod and force end release_system.
- **t** - Do not use, will confuse trace status polling and unload.
Storing Memory Contents to an Absolute File

The "Getting Started" chapter shows you how to load absolute files into emulation or target system memory. You can also store emulation or target system memory to an absolute file with the following command.

```
store memory 800h thru 84fh to absfile
<RETURN>
```

The command above causes the contents of memory locations 800H-84FH to be stored in the absolute file "absfile.X". Notice that the ".X" extension is appended to the specified filename.

Coordinated Measurements

For information on coordinated measurements and how to use them, refer to the "Coordinated Measurements" chapter in the Softkey Interface Reference manual.
Index

A
absolute files
    loading, 2-8
    storing, 5-20
adaptor
    installing, 3-11
address
    symbolic, 2-16
analyzer
    features of, 1-5
    sequencing, 5-19
    status qualifiers, 2-30
analyzer, using the, 2-25
assemblers, 4-10
assembling the demo program, 2-3

B
background monitor
    pin state, 3-14
breaks
    break command, 2-18
    guarded memory accesses, 4-10
    software breakpoints, 2-19
    write to ROM, 4-13
bus request
    while stepping, 1-7
BUSRQ signal, 4-11
    from target system, 3-12

C
cautions
    installing the target system probe, 3-9
characterization of memory, 4-9
Clearing software breakpoints, 2-23
compiling the demo program, 2-3
compress mode, trace display, 2-28

cautions
    installing the target system probe, 3-9
characterization of memory, 4-9
Clearing software breakpoints, 2-23
compiling the demo program, 2-3
compress mode, trace display, 2-28

caution statements
    real-time dependent target system circuitry, 4-4
    software breakpoint cmds. while running user code, 2-19

caution statements
    real-time dependent target system circuitry, 4-4
    software breakpoint cmds. while running user code, 2-19
configuration
  for running example program, 2-4
configuration option
  enable watch dog timer, 4-11
configuration options
  accept BUSRQ, 4-11
  accept interrupts, 4-12
  break processor on write to ROM, 4-13
  emulation analyzer speed, 4-14
  honor target reset, 4-11
  target memory access, 4-12
  trace monitor/user program operation, 4-14
  coordinated measurements, 4-15, 5-20

data address
  trace, 5-17
data bus
  trace, 5-17
demo program
  description, 2-2
demo target board
  installing, 3-7
device table file, 2-3
display command
  memory mnemonic, 2-13
  memory mnemonic with symbols, 2-14
  registers, 2-23
  software breakpoints, 2-21
  symbols, 2-10
  with source code, 2-15
DMA
  external, 4-10
emul700, command to enter the Softkey Interface, 2-3, 2-31
emulation analyzer, 1-5
emulation analyzer speed, 4-14
emulation configuration
  emulation analyzer speed, 4-14
emulation memory
  burst fetch, 1-8
  installing, 3-5
  loading absolute files, 2-8
note on target accesses, 4-10
RAM and ROM characterization, 4-9
single chip mode, 1-8
size of, 4-9
emulation monitor, 1-5
emulation probe
installing, 3-9
emulation probe cable
installing, 3-2
emulator
before using, 2-2
configuration, 4-1
configure the emulator for example, 2-4
device table file, 2-3
feature list, 1-3
prerequisites, 2-2
purpose of, 1-1
supported, 1-3
emulator configuration
break processor on write to ROM, 4-13
enable emulation VBP, 4-7
for example, 2-4
initial CBP value, 4-8
loading, 4-16
monitor base address, 4-7
processor operation mode, 4-6
processor type, 4-5
restrict to real-time runs, 4-4
saving, 4-15
trace monitor/user program operation, 4-14
vector base address, 4-7
Emulator features
emulation memory, 1-4
enable emulation VBP
emulator configuration, 4-7
END assembler directive (pseudo instruction), 2-16
deck command, 2-31, 4-15
evaluation chip, 1-8
execute address
trace, 5-18
execution state, 2-29
exit, Softkey Interface, 2-31

F  file extensions
   .EA and .EB, configuration files, 4-15

G  getting started, 2-1
   global symbols, 2-13
      displaying, 2-10
   guarded memory accesses, 4-10

H  hardware breakpoints, 5-15
   help
      on-line, 2-6
      pod command information, 2-7
      softkey driven information, 2-6

I  in-circuit emulation, 3-1
   initial CBP value
      emulator configuration, 4-8
   installation, 2-2
      software, 2-2
   interactive measurements, 4-15
   interrupt
      during monitor, 1-7
      from target system, 1-7, 3-12
      while stepping, 1-7

L  linkers, 4-10
   linking the demo program, 2-3
   load map, 4-10
   loading absolute files, 2-8
   loading emulator configurations, 4-16
   local symbols
      displaying, 2-11
   locked, end command option, 2-31

M  mapping memory, 4-9
   memory
      characterization, 4-9
      mapping, 4-9
      mnemonic display, 2-13
      mnemonic display with symbols, 2-14

4- Index
modifying, 2-17
searching for strings or expressions, 5-19
with source code, 2-15
mnemonic memory display, 2-13
modify command
configuration, 4-1
memory, 2-17
software breakpoints clear, 2-23
software breakpoints set, 2-20
monitor
breaking into, 2-18
selecting processor type, 4-5
monitor base address
emulator configuration, 4-7
monitor cycles
tracing, 4-14

N
no fetch cycle in trace display, 2-29
nosymbols, 2-10
note
pod command from keyboard, 2-7
status line error, 2-9
notes
pod commands that should not be executed, 5-19
software breakpoints not allowed in target ROM, 2-19
software breakpoints only at opcode addresses, 2-19
target accesses to emulation memory, 4-10
write to ROM analyzer status, 4-14

O
on-line help, 2-6
option
displaying trace, 2-30

P
PATH, HP-UX environment variable, 2-3
Pin guard
target system probe, 3-9
pod_command, 2-7
features available with, 5-19
help information, 2-7
prerequisites for using the emulator, 2-2
processor operation mode
emulator configuration, 4-6
RAM, mapping emulation or target, 4-10
real-time execution
  restricting the emulator to, 4-4
register bank
  breaking into the monitor, 1-8
register commands, 1-5
registers
  display/modify, 2-23
release_system
  end command option, 2-31, 4-15 - 4-16
relocatable files, 4-10
reset
  during monitor, 1-7
  target system, 3-1
reset (emulator)
  running from target reset, 2-16
reset (reset emulator) command, 2-31
RESET signal, 4-11
  from target system, 3-12
restrict to real-time runs
  emulator configuration, 4-4
  permissible commands, 4-4
  target system dependency, 4-4
ROM
  mapping emulation or target, 4-10
  writes to, 4-10
run command, 2-16
run from reset, 3-1
run from target reset, 4-11

saving the emulator configuration, 4-15
sequencer, analyzer, 5-19
set
  source on inverse video, 2-25
simulated I/O, 4-15
softkey driven help information, 2-6
Softkey Interface
  entering, 2-3
  exiting, 2-31
  on-line help, 2-6
software breakpoints, 2-19
  and NMI, 2-19
clearing, 2-23
displaying, 2-21
enabling/disabling, 2-20
setting, 2-20
software installation, 2-2
source line referencing, 2-25
source lines displaying, 2-12
status trace, 5-17
status qualifiers, 2-30
step command, 2-24
string delimiters, 2-7
symbolic addresses, 2-16
symbols displaying, 2-10
system overview, 2-2

T

target memory
loading absolute files, 2-8
RAM and ROM characterization, 4-10
target system
dependency on executing code, 4-4
interface, 3-15
Target system probe
pin guard, 3-9
terminal interface, 2-7
trace
no fetch cycle, 2-29
simple trigger, 2-25
trace, displaying with compress mode, 2-28
tracing monitor operation, 4-14
transfer address, running from, 2-16
trigger position, 2-30

U

user (target) memory
loading absolute files, 2-8
user program operation, tracing, 4-14

V

vector area, 1-7, 5-15
external bus mode, 5-15
single chip mode, 5-15
vector base address
emulator configuration, 4-7

W watched dog timer
during monitor, 1-7
window systems, 2-31
write to ROM break, 4-13