



# Simulator for H8/300, H8/300H and H8S

TRACE32 Online Help

TRACE32 Directory

TRACE32 Index

TRACE32 Documents .....		
TRACE32 Instruction Set Simulators .....		
<b>Simulator for H8/300, H8/300H and H8S</b> .....		<b>1</b>
<b>TRACE32 Simulator License</b> .....		<b>4</b>
<b>Quick Start of the Simulator</b> .....		<b>5</b>
<b>Peripheral Simulation</b> .....		<b>7</b>
<b>Troubleshooting</b> .....		<b>7</b>
<b>FAQ</b> .....		<b>7</b>
<b>Specific SYStem Commands</b> .....		<b>8</b>
SYStem.CONFIG	Configure debugger according to target topology	8
SYStem.CPU	CPU type	8
SYStem.LOCK	Lock and tristate the debug port	8
SYStem.MemAccess	Select run-time memory access method	9
SYStem.Mode	Establish the communication with the simulator	9
SYStem.Option.Advanced	Advanced addressing mode	10
SYStem.Option.EXR	EXR mode setting	10
SYStem.Option.IMASKASM	Disable interrupts while single stepping	10
SYStem.Option.IMASKHLL	Disable interrupts while HLL single stepping	11
<b>CPU specific TrOnchip Commands</b> .....		<b>12</b>
TrOnchip	Onchip triggers	12
<b>Memory Classes</b> .....		<b>13</b>

The screenshot displays the TRACE32 PowerView for H8 simulator interface. The main window shows assembly code for the file 'iarh83.c'. The code includes instructions like 'cmp.b', 'subx.b', 'bgt', 'mov.w', 'add.s.w', 'mov.b', and 'bra'. A C code snippet is visible, showing a 'for' loop and an 'if' statement. Below the assembly view, the 'B::Frame/Locals/Caller' window shows the call stack with 'sieve()' and 'main()' frames. The 'B::Var.Watch' window displays the current state of variables, including 'flags' and 'ast'. The status bar at the bottom indicates the current instruction is 'P:001146' and the processor is 'stopped'.

Step	Over	Diverge	Return	Up	Go	Break	Mode	Find:	iarh83.c
P:001140	AE12								
P:001142	B600								
P:001144	4E0C								
P:001146	0D46								
P:001148	0B04								
P:00114A	FD01								
P:00114C	6EEDC248								
P:001150	40EC								
692									
P:001152	1944								
P:001154	0D46								
P:001156	AE12								
P:001158	B600								
P:00115A	4E2A								
694									
P:00115C	6E4EC248								

```
for ( i = 0 ; i <= SIZE ; i++ )
{
    sub.w r4,r4
    mov.w r4,r6
    cmp.b #0x12,r6l ; #18,r6l
    subx.b #0x0,r6h ; #0,r6h
    bgt 0x1186
    if ( flags[ i ] )
        mov.b @(0xC248,r4),r6l ; @(flags,r4),r6l
}
```

B::Frame/Locals/Caller

```
-000| sieve()
    | i = 5
    | primz = 4582
    | k = 24910
    | anzah1 = 0
-001| main()
    | return = -251
    | j = 24910
    | p = 0x0
676 | { sieve();
```

B::Var.Watch

```
@ flags = (1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0)
@ ast = (word = 0x0, count = 12346, left = 0xC0CA, right = 0x0, field1 = 1, fiel
```

B::: 50

components trace Data Var List PERF SYSTEM Step Go Break sYmbol other previous

P:001146 \\iarh83t\iarh83\sieve+0x16 stopped MIX UP

All general commands are described in the [“PowerView Command Reference”](#) (ide\_ref.pdf) and [“General Commands Reference”](#).

# TRACE32 Simulator License

---

[build 68859 - DVD 02/2016]

The extensive use of the TRACE32 Instruction Set Simulator requires a *TRACE32 Simulator License*.

For more information, see [www.lauterbach.com/sim\\_license.html](http://www.lauterbach.com/sim_license.html).

# Quick Start of the Simulator

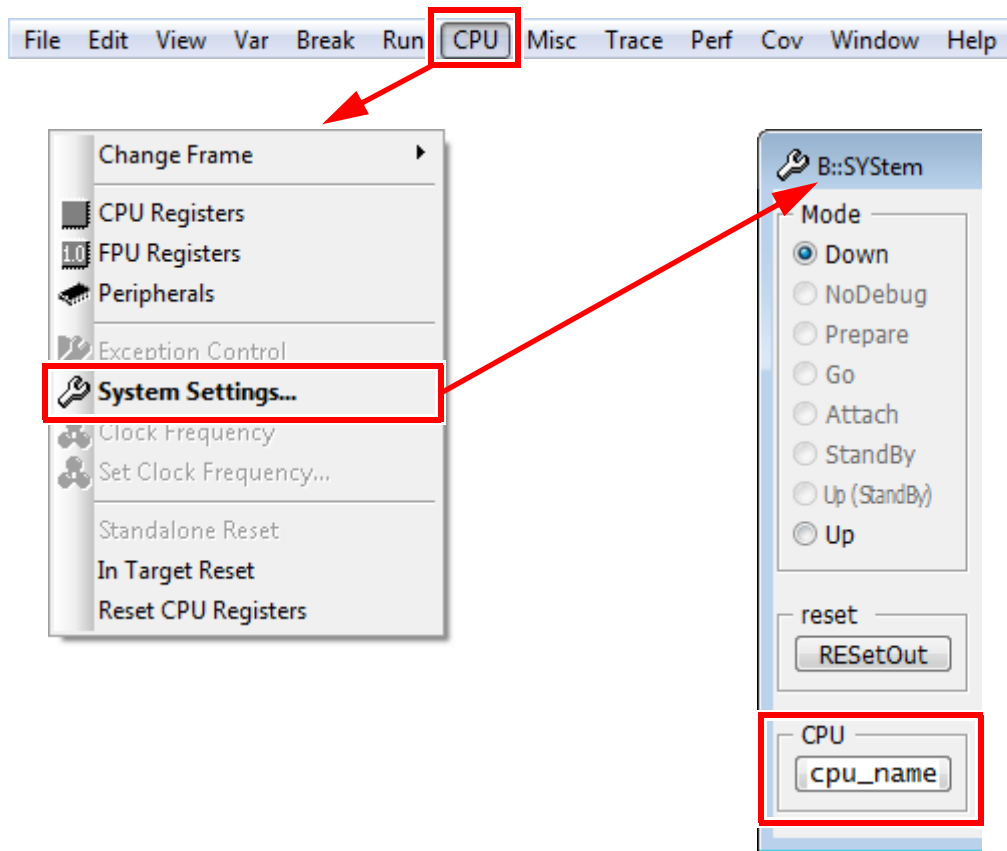
To start the simulator, proceed as follows:

1. Select the device prompt for the Simulator and reset the system.

```
B : :  
  
RESet
```

The device prompt `B : :` is normally already selected in the [TRACE32 command line](#). If this is not the case, enter `B : :` to set the correct device prompt. The `RESet` command is only necessary if you do not start directly after booting TRACE32.

2. Specify the CPU specific settings.



```
SYStem.CPU <cpu_name>
```

The default values of all other options are set in such a way that it should be possible to work without modification. Please consider that this is probably not the best configuration for your target.

### 3. Enter debug mode.

```
SYStem.Up
```

This command resets the CPU and enters debug mode. After this command is executed it is possible to access memory and registers.

### 4. Load the program.

```
Data.LOAD.<file_format> <file> ; load program and symbols
```

See the [Data.LOAD](#) command reference for a list of supported file formats. If uncertain about the required format, try [Data.LOAD.auto](#).

A detailed description of the [Data.LOAD](#) command and all available options is given in the reference guide.

### 5. Start-up example

A typical start sequence is shown below. This sequence can be written to a PRACTICE script file (\*.cmm, ASCII format) and executed with the command **DO** *<file>*.

```
B:: ; Select the ICD device prompt
WinCLEAR ; Clear all windows
SYStem.CPU <cpu_name> ; Select CPU type
SYStem.Up ; Reset the target and enter
; debug mode
Data.LOAD.<file_format> <file> ; Load the application
Register.Set pc main ; Set the PC to function main
PER.view ; Show clearly arranged
; peripherals in window *)
List.Mix ; Open source code window *)
Register.view /SpotLight ; Open register window *)
Frame.view /Locals /Caller ; Open the stack frame with
; local variables *)
Var.Watch %Spotlight flags ast ; Open watch window for
; variables *)
```

\*) These commands open windows on the screen. The window position can be specified with the [WinPOS](#) command.

# Peripheral Simulation

---

For more information, see “[API for TRACE32 Instruction Set Simulator](#)” (simulator\_api.pdf).

## Troubleshooting

---

No information available.

## FAQ

---

Please refer to <https://support.lauterbach.com/kb>.

# Specific SYStem Commands

---

## SYStem.CONFIG

Configure debugger according to target topology

---

The **SYStem.CONFIG** commands have no effect on the simulator. They are only provided to allow the user to run PRACTICE scripts written for the debugger within the simulator without modifications.

## SYStem.CPU

CPU type

---

Format:           **SYStem.CPU** <type>

<mode>:           **H83001** | **H83002** ...

Selects the processor type. The ROM monitor software requires also a modification in the configuration table for different processor types.

## SYStem.LOCK

Lock and tristate the debug port

---

Format:           **SYStem.LOCK** [ON | OFF]

The command has no effect for the simulator.



Format: **SYSystem.MemAccess Enable | StopAndGo | Denied**  
**SYSystem.ACCESS** (deprecated)

**Enable** Real-time memory access during program execution to target is enabled.  
**CPU** (deprecated)

**Denied** Real-time memory access during program execution to target is disabled.

**StopAndGo** Temporarily halts the core(s) to perform the memory access. Each stop takes some time depending on the speed of the JTAG port, the number of the assigned cores, and the operations that should be performed.

## SYSystem.Mode

## Establish the communication with the simulator

Format: **SYSystem.Mode <mode>**  
**SYSystem.Down** (alias for **SYSystem.Mode Down**)  
**SYSystem.Up** (alias for **SYSystem.Mode Up**)

<mode>:  
**Down**  
**NoDebug**  
**Go**  
**Up**

Default: Down.

Selects the target operating mode.

**Down** The CPU is in reset. Debug mode is not active. Default state and state after fatal errors.

**NoDebug** The CPU is running. Debug mode is not active. Debug port is tristate. In this mode the target should behave as if the debugger is not connected.

**Go** The CPU is running. Debug mode is active. After this command the CPU can be stopped with the break command or if any break condition occurs.

**Up** The CPU is not in reset but halted. Debug mode is active. In this mode the CPU can be started and stopped. This is the most typical way to activate debugging.

If the mode **Go** is selected, this mode will be entered, but the control button in the **SYStem.state** window jumps to the mode **Up**.

## SYStem.Option.Advanced

## Advanced addressing mode

Format: **SYStem.Option.Advanced** [ON | OF]

Defines the address mode of the CPU.

**off** Normal address mode (64K).

**on** Advanced address mode (16M).

## SYStem.Option.EXR

## EXR mode setting

Format: **SYStem.Option.EXR** [ON | OFF]

TBD.

## SYStem.Option.IMASKASM

## Disable interrupts while single stepping

Format: **SYStem.Option.IMASKASM** [ON | OFF]

Default: OFF.

If enabled, the interrupt mask bits of the CPU will be set during assembler single-step operations. The interrupt routine is not executed during single-step operations. After single step the interrupt mask bits are restored to the value before the step.

Format: **SYStem.Option.IMASKHLL [ON | OFF]**

Default: OFF.

If enabled, the interrupt mask bits of the CPU will be set during HLL single-step operations. The interrupt routine is not executed during single-step operations. After single step the interrupt mask bits are restored to the value before the step.

This command group has no effect on the TRACE32 Instruction Set Simulator.

# Memory Classes

---

Memory Class	Description
D	Data
P	Program