

GTM Debugger and Trace

TRACE32 Online Help

TRACE32 Directory

TRACE32 Index

TRACE32 Documents	
ICD In-Circuit Debugger	
Processor Architecture Manuals	
GTM	
GTM Debugger and Trace	1
History	7
Introduction	8
Brief Overview of Documents for New Users	8
Demo and Start-up Scripts	9
GTM Debugger and Trace	9
Warning	10
Target Design Requirement/Recommendations	11
General	11
For MPC57xx	11
Quick Start GTM Debugger	12
AURIX Architecture - Quick Start	13
MPC57xx/SPC58xx/SPC57xx Architecture - Quick Start	14
RH850 Architecture - Quick Start	15
S32 and SR6 Architecture - Quick Start	16
Troubleshooting	17
FAQ	17
Configuration	18
System Overview	18
GTM Operating Modes	18
Debugging the GTM	18
Breakpoints and Watchpoints	20
Software Breakpoints	20
On-chip Breakpoints/Watchpoints (only MPC57xx)	20
On-chip Breakpoints/Watchpoints (only Tricore Emulation Device)	20
Access Classes	21
Address Spaces and Addressing Modes	21

CPU specific SYStem Commands	22
SYStem.CONFIG.state	Display target configuration 22
SYStem.CONFIG	Configure debugger according to target topology 23
SYStem.CONFIG.CORE	Assign core to TRACE32 instance 23
SYStem.CONFIG.DEBUGPORTTYPE	Set debug cable interface mode 24
SYStem.CONFIG.MCSModule	Select the MCS module 24
SYStem.CONFIG.PortSHaRing	Control sharing of debug port with other tool 24
SYStem.CPU	Select the CPU type 25
SYStem.JtagClock	Select the debug clock frequency 26
SYStem.LOCK	Lock and tristate the debug port 26
SYStem.MemAccess	Select run-time memory access method 26
SYStem.Mode	Select operation mode 27
SYStem.Option.DUALPORT	Implicitly use run-time memory access 28
SYStem.Option.IMASKASM	Disable interrupts while single stepping 28
SYStem.Option.IMASKHLL	Disable interrupts while HLL single stepping 28
SYStem.CONFIG.DEBUGPORT	Select target interface 29
NEXUS Commands	30
NEXUS.ARU	Control for ARU trace messages 30
NEXUS.ARUAccessX	ARU debugging address 30
NEXUS.FTM	Control for fetch trace messages 30
NEXUS.FTCE	Fetch trace enable per channel 31
NEXUS.DPLL	DPLL data trace messages 31
NEXUS.DPLLMemory	RAM module selection 32
NEXUS.DTM	Control for data trace messages 32
NEXUS.DTC	Data trace channel select 32
NEXUS.DTCE	Data trace enable per channel 33
NEXUS.OFF	Switch the NEXUS trace port off 33
NEXUS.ON	Switch the NEXUS trace port on 33
NEXUS.RESet	Reset NEXUS trace port settings 34
NEXUS.RefClock	Enable Aurora reference clock 34
NEXUS.PortMode	Set NEXUS trace port frequency 34
NEXUS.PortSize	Set trace port width 35
NEXUS.state	Display Nexus configuration window 35
NEXUS.SYNC	Address-sync trace messaging enable 35
NEXUS.TimeStamps	Control for timestamp trace messages 36
NEXUS.TimeStampSel	Select TBU channel for timestamp trace messages 36
General TrOnchip Commands	37
TrOnchip.state	Display onchip trigger window 37
TrOnchip.CONVert	Adjust range breakpoint in on-chip resource 38
TrOnchip.RESet	Reset on-chip trigger settings 38
TrOnchip.VarCONVert	Adjust complex breakpoint in on-chip resource 38
TriCore specific TrOnchip Commands	39

TrOnchip.ARU	ARU settings	39
TrOnchip.ARU.ACCESS	ARU debugging address	39
TrOnchip.MCS	MCS setting	40
TrOnchip.MCS.Channel	Select the MCS channel	40
TrOnchip.MCS.Module	Select the MCS module	40
TrOnchip.OTGBx	OTGB0 and OTGB1 settings	41
TrOnchip.OTGBx.SELect	Select trace source	41
TrOnchip.OTGBx.LowBMType	Select IOS module for low byte	42
TrOnchip.OTGBx.HighBMType	Select IOS module for high byte	43
TrOnchip.OTGBx.LowBMInst	Low byte module instance	43
TrOnchip.OTGBx.HighBMInst	High byte module instance	43
TrOnchip.OTGBx.SENSitivNeg	Bit sensitive trace selection	44
TrOnchip.OTGBx.SENSitivPos	Bit sensitive trace selection	44
TrOnchip.OTGB2	OTGB2 setting	45
TrOnchip.OTGB2.SELect	Select trace source	45
TrOnchip.OTGBM0	OTGBM0 setting	46
TrOnchip.OTGBM0.SELect	Select trace source	46
TrOnchip.OTGBM1	OTGBM1 setting	47
TrOnchip.OTGBM1.SELect	Select trace source	47
CPU specific TrOnchip Commands		48
TrOnchip.ARUx.Address	ARU address compare	48
TrOnchip.ARUx.DataHigh	ARU data low value compare	48
TrOnchip.ARUx.DataLow	ARU data low value compare	48
TrOnchip.ARUx.HALT	ARU access halt enable	49
TrOnchip.ARUx.Watchpoint	ARU access watchpoint enable	49
TrOnchip.ATOMWPCx	ATOM watchpoint settings	50
TrOnchip.ATOMWPCx.Channel	ATOM channel selection	50
TrOnchip.ATOMWPCx.HALT	ATOM halt enable	50
TrOnchip.ATOMWPCx.Module	ATOM sub-module selection	50
TrOnchip.ATOMWPCx.TIMING	ATOM watchpoint enable	51
TrOnchip.ATOMWPCx.Transition	ATOM channel slope selection	51
TrOnchip.ATOMWPCx.Watchpoint	ATOM watchpoint enable	51
TrOnchip.DPLLWPC1	DPLL watchpoint settings	52
TrOnchip.DPLLWPC1.Event	DPLL source selection	52
TrOnchip.DPLLWPC1.HALT	DPLL TASI/SASI halt enable	52
TrOnchip.DPLLWPC1.Transition	DPLL TASI/SASI slope selection	52
TrOnchip.DPLLWPC1.Watchpoint	DPLL TASI/SASI watchpoint enable	52
TrOnchip.DPLLWPC2	DPLL RAM watchpoint settings	53
TrOnchip.DPLLWPC2.Address	DPLL RAM address compare	53
TrOnchip.DPLLWPC2.ACCESS	DPLL RAM read/write control	53
TrOnchip.DPLLWPC2.Data	DPLL RAM data compare	53
TrOnchip.DPLLWPC2.HALT	DPLL RAM access halt enable	53
TrOnchip.DPLLWPC2.Module	DPLL RAM module selection	54

TrOnchip.DPLLWPC2.Watchpoint	DPLL RAM access watchpoint enable	54
TrOnchip.EVTOx	Select EVTOx output	54
TrOnchip.SPEx	SPEx	55
TrOnchip.SPEx.DIR	SPEx DIR watchpoint settings	55
TrOnchip.SPEx.DIR.HALT	SPEx DIR halt enable	55
TrOnchip.SPEx.DIR.TIMING	SPEx DIR watchpoint enable	55
TrOnchip.SPEx.DIR.Transition	SPEx DIR slope selection	55
TrOnchip.SPEx.DIR.Watchpoint	SPEx DIR watchpoint enable	56
TrOnchip.SPEx.NIPD	SPEx NIPD watchpoint settings	57
TrOnchip.SPEx.NIPD.HALT	SPEx NIPD halt enable	57
TrOnchip.SPEx.NIPD.TIMING	SPEx NIPD watchpoint enable	57
TrOnchip.SPEx.NIPD.Transition	SPEx NIPD slope selection	57
TrOnchip.SPEx.NIPD.Watchpoint	SPEx NIPD watchpoint enable	58
TrOnchip.TBU	TBU watchpoint settings	59
TrOnchip.TBUx.Data	TBU data value compare	59
TrOnchip.TBUx.HALT	TBU access halt enable	59
TrOnchip.TBUx.Watchpoint	TBU access watchpoint enable	59
TrOnchip.TBU0.SELect	TBU0 type selection	60
TrOnchip.TIMWPC	TIM watchpoint settings	61
TrOnchip.TIMWPCx.Channel	TIM channel selection	61
TrOnchip.TIMWPCx.HALT	TIM halt enable	61
TrOnchip.TIMWPCx.Module	TIM sub-module selection	61
TrOnchip.TIMWPCx.TIMING	TIM watchpoint enable	62
TrOnchip.TIMWPCx.Transition	TIM channel slope selection	62
TrOnchip.TIMWPCx.Watchpoint	TIM watchpoint enable	62
TrOnchip.TOMWPC	TOM watchpoint settings	63
TrOnchip.TOMWPCx.Channel	TOM channel selection	63
TrOnchip.TOMWPCx.HALT	TOM halt enable	63
TrOnchip.TOMWPCx.Module	TOM sub-module selection	63
TrOnchip.TOMWPCx.TIMING	TOM watchpoint enable	64
TrOnchip.TOMWPCx.Transition	TOM channel slope selection	64
TrOnchip.TOMWPCx.Watchpoint	TOM watchpoint enable	64
TrOnchip.WPCE	Breakpoint enable per channel	65
Arm specific TrOnchip Commands		66
TrOnchip.TIOINWPCx	TIO input watchpoint settings	66
TrOnchip.TIOINWPCx.Channel	TIO input channel selection	66
TrOnchip.TIOINWPCx.HALT	TIO input halt enable	66
TrOnchip.TIOINWPCx.Module	TIO input sub-module selection	66
TrOnchip.TIOINWPCx.TIMING	TIO input watchpoint enable	67
TrOnchip.TIOINWPCx.Transition	TIO input channel slope selection	67
TrOnchip.TIOINWPCx.Watchpoint	TIO input watchpoint enable	67
TrOnchip.TIOOUTWPCx	TIO output watchpoint settings	68
TrOnchip.TIOOUTWPCx.Channel	TIO output channel selection	68

TrOnchip.TIOOUTWPCx.HALT	TIO output halt enable	68
TrOnchip.TIOOUTWPCx.Module	TIO output sub-module selection	68
TrOnchip.TIOOUTWPCx.Polarity	TIO output polarity selection	69
TrOnchip.TIOOUTWPCx.TIMING	TIO output watchpoint enable	69
TrOnchip.TIOOUTWPCx.Transition	TIO output channel slope selection	69
TrOnchip.TIOOUTWPCx.Watchpoint	TIO output watchpoint enable	69
RH850 specific TrOnchip Commands		70
TrOnchip.BreakChannel	Select the channel for breakpoints	70
TrOnchip.ATOMSlotx	Select the ATOM module for trace	70
TrOnchip.TIMSlotx	Select the TIM module for trace	70
JTAG Connector		71
Mechanical Description		71

History

- 05-Dec-22 Support for S32 and SR6 CPUs.
New subchapter '[S32 and SR6 Architecture - Quick Start](#)'.
New '[Arm Specific TrOnchip Commands](#)', for S32 only.

Introduction

This document describes the processor specific settings and features for TRACE32-ICD for the GTM core.

TRACE32 supports the GTM for the following processor architectures:

- AURIX from Infineon
- MPC57xx from Freescale
- SPC58xx SPC57xx from STMicroelectronics
- RH850 from Renesas
- S32E2, S32Z2 from NXP
- Stellar SR6P6/P7/G7 from STMicroelectronics

Please keep in mind that only the [Processor Architecture Manual](#) (the document you are reading at the moment) is CPU specific, while all other parts of the online help are generic for all CPUs supported by Lauterbach. So if there are questions related to the CPU, the Processor Architecture Manual should be your first choice.

If some of the described functions, options, signals or connections in this Processor Architecture Manual are only valid for a single CPU or for specific families, the name(s) of the family(ies) is added in brackets.

Brief Overview of Documents for New Users

Architecture-independent information:

- [“Training Basic Debugging”](#) (training_debugger.pdf): Get familiar with the basic features of a TRACE32 debugger.
- [“T32Start”](#) (app_t32start.pdf): T32Start assists you in starting TRACE32 PowerView instances for different configurations of the debugger. T32Start is only available for Windows.
- [“General Commands”](#) (general_ref_<x>.pdf): Alphabetic list of debug commands.

Architecture-specific information:

- [“Processor Architecture Manuals”](#): These manuals describe commands that are specific for the processor architecture supported by your Debug Cable. To access the manual for your processor architecture, proceed as follows:
 - Choose **Help** menu > **Processor Architecture Manual**.
- [“OS Awareness Manuals”](#) (rtos_<os>.pdf): TRACE32 PowerView can be extended for operating system-aware debugging. The appropriate OS Awareness manual informs you how to enable the OS-aware debugging.

Demo and Start-up Scripts

Lauterbach provides ready-to-run start-up scripts for known GTM based hardware.

To search for PRACTICE scripts, do one of the following in TRACE32 PowerView:

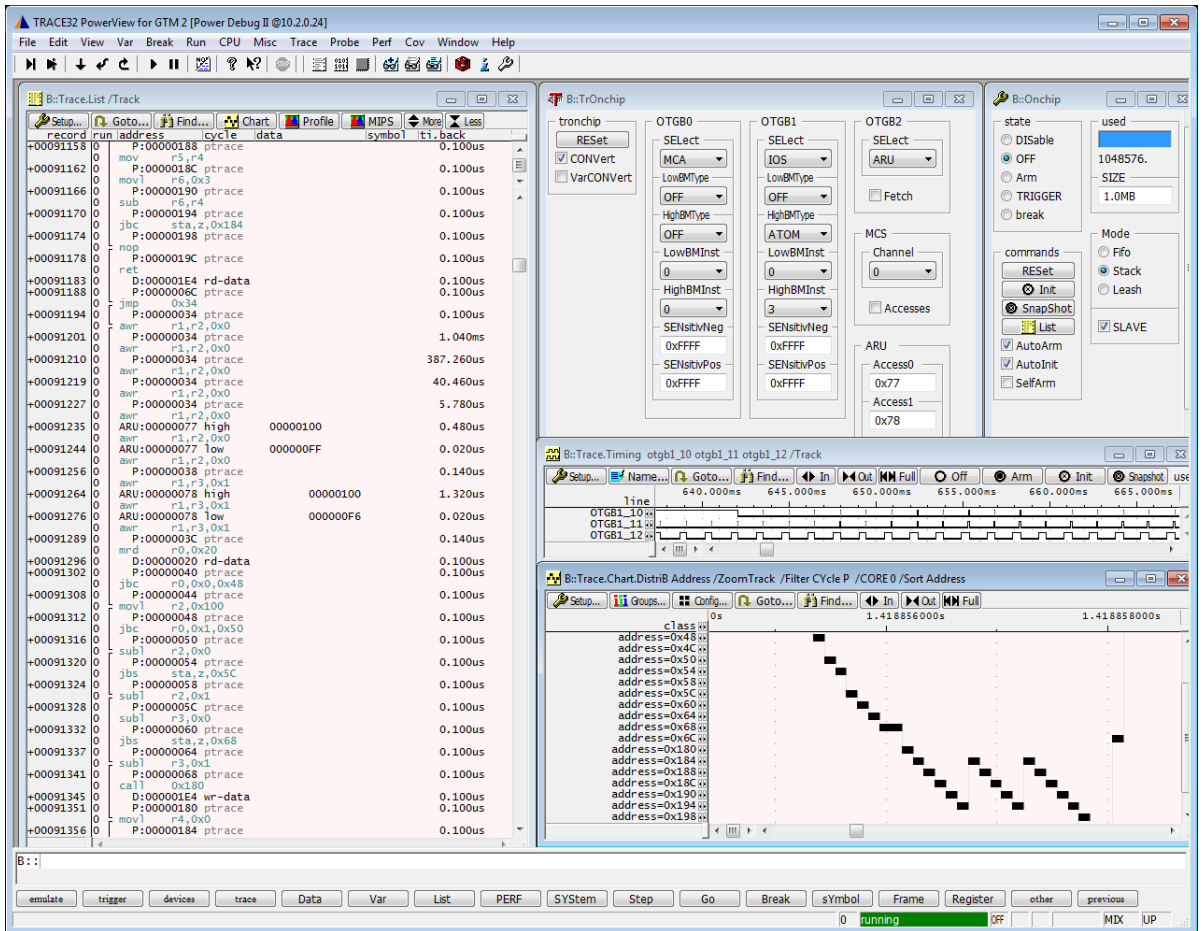
- Type at the command line: **WELCOME.SCRIPTS**
- or choose **File** menu > **Search for Script**.

You can now search the demo folder and its subdirectories for PRACTICE start-up scripts (*.cmm) and other demo software.

You can also manually navigate in the `~/demo/gtm/` subfolder of the system directory of TRACE32.

GTM Debugger and Trace

A debug session might look like this:



This is an example with an AURIX device.

Warning

Signal Level

MPC57XX	The debugger drives the output pins of the JTAG/OnCE connector with the same level as detected on the VCCS pin. If the IO pins of the processor are 3.3 V compatible then the VCCS should be connected to 3.3 V.
---------	--

ESD Protection

WARNING:	To prevent debugger and target from damage it is recommended to connect or disconnect the Debug Cable only while the target power is OFF. Recommendation for the software start: 1. Disconnect the Debug Cable from the target while the target power is off. 2. Connect the host system, the TRACE32 hardware and the Debug Cable. 3. Power ON the TRACE32 hardware. 4. Start the TRACE32 software to load the debugger firmware. 5. Connect the Debug Cable to the target. 6. Switch the target power ON. 7. Configure your debugger e.g. via a start-up script. Power down: 1. Switch off the target power. 2. Disconnect the Debug Cable from the target. 3. Close the TRACE32 software. 4. Power OFF the TRACE32 hardware.
----------	--

Target Design Requirement/Recommendations

General

- Locate the **JTAG/OnCE/Nexus connector** as close as possible to the processor to minimize the capacitive influence of the trace length and cross coupling of noise onto the JTAG signals.

For MPC57xx

- Ensure that the debugger signal ($\overline{\text{HRESET}}$) is connected directly to the $\overline{\text{HRESET}}$ of the processor. This will provide the ability for the debugger to drive and sense the status of $\overline{\text{HRESET}}$. The target design should only drive the $\overline{\text{HRESET}}$ with open collector, open drain. $\overline{\text{HRESET}}$ should not be tied to $\overline{\text{PORESET}}$, because the debugger drives the $\overline{\text{HRESET}}$ and DSCK to enable JTAG operation.

Quick Start GTM Debugger

This chapter describes how to start up the debugger for the following architectures:

- [“AURIX Architecture - Quick Start”](#), page 13.
- [“MPC57xx/SPC58xx/SPC57xx Architecture - Quick Start”](#), page 14.
- [“RH850 Architecture - Quick Start”](#), page 15.
- [“S32 and SR6 Architecture - Quick Start”](#), page 16.

Every MCS module must be started as one TRACE32 instance (AMP), Channels are configured as SMP.

In your TRACE32 installation directory, there is the subdirectory `~/demo/gtm/hardware` where you will find example scripts and demo software.

Multiple Instances of the GTM Debugger

A demo script example for advanced users is included in your TRACE32 installation, demonstrating how to start multiple instances of the GTM debugger. The example applies to AURIX.

To access the script, run this command:

```
B: :CD.PSTEP ~/demo/gtm/hardware/triboard-tc2xx/tc27x_tc29x_demo.cmm
```

AURIX Architecture - Quick Start

1. Set the CPU type to load the CPU specific settings.:

```
SYStem.CPU TC277TE
```

2. Configure select target core. See [SYStem.CONFIG.CORE](#) for details.

```
;               <core_index>   <chip>  
SYStem.CONFIG.CORE      6.      1.
```

3. Select the MCS module.

```
SYStem.CONFIG.MCSModule MCS2      ; MCS2 is a module name
```

4. Start debug session by attaching to the GTM:

```
SYStem.Mode.Attach
```

5. Load the debug symbols. The program code will be usually loaded by the master core (TriCore).

```
Data.LOAD Elf app.elf /NoCODE /NoRegister
```

6. Enable the Program Trace):

```
TrOnchip.OTGB0 SElect MCA
```

7. Select the MCS2 Module for tracing:

```
TrOnchip.MCS Module MCS2
```

8. Define the channel or all channels to be used for breakpoints:

```
TrOnchip.MCS Channel ALL
```

9. Set up breakpoint(s) and run master CPU afterwards:

```
Break.Set func_increment /Onchip
```

MPC57xx/SPC58xx/SPC57xx Architecture - Quick Start

1. Set the CPU type to load the CPU specific settings:

```
SYStem.CPU MPC5746M
```

2. Configure select target core. See [SYStem.CONFIG.CORE](#) for details.

```
;          <core_index>    <chip>  
SYStem.CONFIG.CORE      6.      1.
```

3. Select the MCS module.

```
SYStem.CONFIG.MCSModule MCS2      ; MCS2 is a module name
```

4. Start debug session by attaching to the GTM:

```
SYStem.Mode.Attach
```

5. Load the debug symbols. The program code will be usually loaded by the master core (PowerPC).

```
Data.LOAD.Elf app.elf /NoCODE /NoRegister
```

6. Define the channel or channels to be used for breakpoints:

```
TrOnchip.WPCE 0xFF
```

7. Set up breakpoint(s) and run master CPU afterwards:

```
Break.Set func_increment /Onchip
```

1. Set the CPU type to load the CPU specific settings.:

```
SYStem.CPU R7F701326
```

2. Configure select target core. See [SYStem.CONFIG.CORE](#) for details.

```
;          <core_index>    <chip>  
SYStem.CONFIG.CORE      4.      1.
```

3. Select the MCS module

```
SYStem.CONFIG.MCSModule MCS0      ; MCS0 is a module name
```

4. Start debug session by attaching to the GTM:

```
SYStem.Mode.Attach
```

5. Load the debug symbols. The program code will be usually loaded by the master core (Rh850):

```
Data.LOAD.Elf app.elf /NoCODE /NoRegister
```

6. Define the channel to be used for breakpoints:

```
TrOnchip.BreakChannel 3
```

7. Set up breakpoint(s) and run master CPU afterwards:

```
Break.Set func_increment /Onchip
```

S32 and SR6 Architecture - Quick Start

1. Set the CPU type to load the CPU specific settings.:

```
SYStem.CPU S32Z270
```

2. Configure select target core. See [SYStem.CONFIG.CORE](#) for details.

```
;          <core_index>    <chip>  
SYStem.CONFIG.CORE      8.      1.
```

3. Select the same Tap-State as the Arm core

```
SYStem.CONFIG.TapState SElectDRscan
```

4. Select the MCS module

```
SYStem.CONFIG.MCSModule MCS2      ; MCS2 is a module name
```

5. Start debug session by attaching to the GTM:

```
SYStem.Mode.Attach
```

6. Load the debug symbols. The program code will be usually loaded by the master core:

```
Data.LOAD.Elf app.elf /NoCODE /NoRegister
```

7. Define the channel or channels to be used for breakpoints:

```
TrOnchip.WPCE 0xFF
```

8. Set up breakpoint(s) and run master CPU afterwards:

```
Break.Set func_increment /Onchip
```

Troubleshooting

Only GTM with Tricore (TC275 A-Step): It is not recommended to break the GTM core because accesses from the debugger for register or RAM image generation lead to a misbehavior of the GTM.

FAQ

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

System Overview

- In case of AURIX devices, see section [“System Overview”](#) (debugger_tricore.pdf).
- In case of MPC57xx or SPC58xx, please refer to [“Qorivva MPC5xxx/SPC5xx Debugger and NEXUS Trace”](#) (debugger_mpc5500.pdf):
 - [JTAG Debugger](#)
 - [Aurora Nexus Debugger and Trace](#)
- In case of RH850, see section [“System Overview”](#) (debugger_rh850.pdf).

GTM Operating Modes

The Channels of the MCS modules are from case to case active. When all Channels of the MCS Module are disabled, the debugger will display IDLE in the status line. The IDLE state is only for the MCS Module which selected with SYStem.CONFIG.MCSModule. You don't see if the Channels of the other available MCS Modules are also disabled.

Debugging the GTM

Before the GTM debug session can be started, the main core (MPC, Tricore or RH850) has to initialize the GTM i.e. load the program and initiate a host service request.

This is the recommended method to start an GTM debug session:

1. Program main application to FLASH using main core (if not already programmed)
2. Reset processor and begin debug session on main core ([SYStem.Up](#))
3. Begin GTM debug session using [SYStem.Mode.Attach](#).
4. Load debug symbols for all main and MCS Module ([Data.LOAD.Elf](#)).

NOTE:

Every MCS Module of the GTM have their own debug symbols. The source file of the main core does *not* include debug symbols of the GTM.

5. To debug a service request or function, set a Breakpoint or enable a debug event on a service request. When another Channel of the MCS Module uses the same program it is recommended to enable only this channel which you want to debug. See [TrOnchip.WPCE](#).
6. Run application on main core (Go).
7. The main core will initialize the GTM. The GTM should halt for the debugger when the breakpoint occurs.

Breakpoints and Watchpoints

There are only on-chip breakpoints (HW-BP) available:

Software Breakpoints

Software breakpoints can not be used!

On-chip Breakpoints/Watchpoints (only MPC57xx)

An GTM has up to four on-chip break-/watchpoints. They can be used to

- generate a debug event (core halts for debugger)
- generate a watchpoint hit trace message
- enable/disable trace message generation when the event occurs.

The on-chip break-/watchpoints can be configured for

- instruction address comparison (instruction break/watchpoint)
- data address comparison (optional with data value comparison)

In addition, the break/watchpoints can be enabled for one channel, all channels or a certain set of channels. See [TrOnchip.WPCE](#) for details.

On-chip Breakpoints/Watchpoints (only Tricore Emulation Device)

An GTM has up to four on-chip breakpoints. They can be used to generate a debug event (core halts for debugger)

The on-chip breakpoints can be configured for instruction address comparison (instruction breakpoint)

The [TrOnchip.OTGB0.SELect](#) must be select to MCA. In addition, the breakpoints can be enabled for one channel or for all channels. See [TrOnchip.MCS.Channel](#) for details.

Access Classes

The following [access classes](#) are available:

Access Class	Description
P	Program Memory
D	Data Memory
FPI	Allows access to the data memory space of the main core (HOST); For example, this access class is used by peripheral files.

Address Spaces and Addressing Modes

The GTM cores have an address space which is independent of the main core (Aurix, MPC, SPC or RH850). Every MCS module have their own address space which starts at 0x0.

GTM Address	Main Core Address
MCS0 P:0x0000	MPC57XX: A:0xF0138000 TriCore: A:0xFFD38000 RH850: A:0xFFE38000
MCS1 P:0x0000	MPC57XX: A:0xF0148000 TriCore: A:0xFFD48000 RH850: A:0xFFE48000
MCS2 P:0x0000	MPC57XX: A:0xF0158000 TriCore: A:0xFFD58000
MCS3 P:0x0000	MPC57XX: A:0xF0168000 TriCore: A:0xFFD68000
MCS4 P:0x0000	MPC57XX: A:0xF0178000 TriCore: A:0xFFD78000
MCS5 P:0x0000	MPC57XX: A:0xF0188000 TriCore: A:0xFFD88000
MCS6 P:0x0000	MPC57XX: A:0xF0198000 TriCore: A:0xFFD98000

SYStem.CONFIG.state

Display target configuration

Format:	SYStem.CONFIG.state [/<tab>]
<tab>:	DebugPort Jtag MOdule XCP

Opens the **SYStem.CONFIG.state** window, where you can view and modify most of the target configuration settings. The configuration settings tell the debugger how to communicate with the chip on the target board and how to access the on-chip debug and trace facilities in order to accomplish the debugger’s operations.

Alternatively, you can modify the target configuration settings via the [TRACE32 command line](#) with the **SYStem.CONFIG** commands. Note that the command line provides *additional* **SYStem.CONFIG** commands for settings that are *not* included in the **SYStem.CONFIG.state** window.

<tab>	Opens the SYStem.CONFIG.state window on the specified tab. For tab descriptions, see below.
DebugPort	Lets you configure the electrical properties of the debug connection, such as the communication protocol or the used pinout.
Jtag	Informs the debugger about the position of the Test Access Ports (TAP) in the JTAG chain which the debugger needs to talk to in order to access the debug and trace facilities on the chip.
MOdule	Allows you to select an MCS module.
XCP	Lets you configure the XCP connection to your target. For descriptions of the commands on the XCP tab, see “XCP Debug Back-End” (backend_xcp.pdf).

Format:

SYStem.CONFIG <mode>

SYStem.MultiCore <mode> (deprecated)

For the description of **SYStem.CONFIG** commands, refer to the debugger manual for the main core:

- [“Qorivva MPC5xxx/SPC5xx Debugger and NEXUS Trace”](#) (debugger_mpc5500.pdf)
- [“TriCore Debugger and Trace”](#) (debugger_tricore.pdf)
- [“RH850 Debugger and Trace”](#) (debugger_rh850.pdf)

This setting is only available for CPUs with JTAG as debug port (not available for BDM).

SYStem.CONFIG.CORE

Assign core to TRACE32 instance

Format:

SYStem.CONFIG CORE <core_index> [<chip_index>]

SYStem.MutiCore.Core <core_index> (deprecated)

This command is used to assign a specific core to a TRACE32 instance. Please make sure that the host debugger’s CPU selection is appropriate before this command is called. If this command is called while a CPU without GTM is selected, the command will fail. The valid parameters for <core-id> are given by debugger implementation:

Architecture / GTM	Core-ID
MPC5744K MPC5743K SPC574K7x	5 (MCS0), 6 (MCS1), 7 (MCS2)
MPC5777M SPC57HM90	5 (MCS0), 6 (MCS1), 7 (MCS2), 8 (MCS3) 9 (MCS4), 10 (MCS5), 11 (MCS6)
MPC5746M SPC57EM80	5 (MCS0), 6 (MCS1), 7 (MCS2), 8 (MCS3)
TriCore	5 (MCS0), 6 (MCS1), 7 (MCS2), 8 (MCS3) 9 (MCS4), 10 (MCS5), 11 (MCS6)
RH850	5 (MCS0), 6 (MCS1)

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:	SYStem.CONFIG.DEBUGPORTTYPE [JTAG DAP2 CJTAG LPD4 LPD1] SYStem.CONFIG.Interface [JTAG DAP2 ...] (deprecated)
---------	--

Default: JTAG.

This command is used to configure the interface mode used by the debugger.

SYStem.CONFIG.MCSModule

Select the MCS module

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:	SYStem.CONFIG.MCSModule <module>
<module>:	MCS0 MCS1 MCS2 ...

This command is used to select the MCS Module.

SYStem.CONFIG.PortSHaRing

Control sharing of debug port with other tool

AURIX, RH850

Format:	SYStem.CONFIG.PortSHaRing [ON OFF Auto] SYStem.Option.ETK [ON AUTO OFF] (deprecated)
---------	---

Configure if the debug port is shared with another tool, e.g., an ETAS ETK.

ON	Request for access to the debug port and wait until the access is granted before communicating with the target.
OFF	Communicate with the target without sending requests.
Auto	Automatically detect a connected tool on next SYStem.Mode Up , SYStem.Mode Attach or SYStem.Mode Go . If a tool is detected switch to mode ON else switch to mode OFF .

The current setting can be obtained by the **PORTSHARING()** function, immediate detection can be performed using **SYStem.DETECT PortSHaRing**.

GTM Debugger and Trace
| 25

GTM is a generic core. Note that special features such as on-chip trace or synchronization with the main core are not supported by the generic cores.

SYStem.JtagClock

Select the debug clock frequency

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:

SYStem.JtagClock *<frequency>*

SYStem.BdmClock *<frequency>* (deprecated)

<frequency>:

1 000 000. ... 50 000 000.

Default: 10 MHz

NOTE:

- If possible, use the same JTAG clock frequency for all cores debugged with the same debug interface.
- MPC57XX: the max. allowed JTAG clock frequency is 1/4th of the core frequency.

SYStem.LOCK

Lock and tristate the debug port

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:

SYStem.LOCK [ON | OFF]

Default: OFF.

If the system is locked, no access to the debug port will be performed by the debugger. While locked, the debug connector of the debugger is tristated. The main intention of the **SYStem.LOCK** command is to give debug access to another tool.

SYStem.MemAccess

Select run-time memory access method

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:

SYStem.MemAccess *<mode>*

<mode>:

Denied | Enable | StopAndGo

This option declares if and how a non-intrusive memory access can take place while the CPU is executing code. Although the CPU is not halted, run-time memory access creates an additional load on the processor's internal data bus.

The run-time memory access has to be activated for each window by using the memory class E: (e.g. **Data.dump** E:0x100) or by using the format option %E (e.g. **Var.View** %E var1). It is also possible to activate this non-intrusive memory access for all memory ranges displayed on the TRACE32 screen by setting **SYStem.Option.DUALPORT ON**.

Denied	Memory access is disabled while the CPU is executing code.
Enable CPU (deprecated)	The debugger performs memory accesses via a dedicated CPU interface.
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.

SYStem.Mode

Select operation mode

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:	SYStem.Mode <i><mode></i> SYStem.Attach (alias for SYStem.Mode Attach) SYStem.Down (alias for SYStem.Mode Down)
<i><mode></i> :	Down Attach

Selects the target reset mode.

Down	Disables the Debugger. The state of the CPU remains unchanged.
Attach	Establishes connection to the GTM.
NoDebug Go StandBy Up	Not applicable for GTM.

Format:

SYStem.Option.DUALPORT [ON | OFF]

Forces all list, dump and view windows to use the memory class E: (e.g. **Data.dump E:0x100**) or to use the format option %E (e.g. **Var.View %E var1**) without being specified. Use this option if you want all windows to be updated while the processor is executing code. This setting has no effect if **SYStem.Option.MemAccess** is disabled.

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 a single step, the interrupt mask bits are restored to the value before the step.

SYStem.Option.IMASKHLL

Disable interrupts while HLL single stepping

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 a single step, the interrupt mask bits are restored to the value before the step.

Format:	SYStem.CONFIG.DEBUGPORT <i><port></i>
<i><port></i> :	DebugCable0 DebugCableA InfineonDAS0 XCP0 Unknown

Selects the interface to the target. The available options depend on whether TRACE32 uses a hardware debugger or runs in HostMCI mode (without TRACE32 hardware).

With TRACE32 hardware

DebugCable0	Uses the debug cable directly connected to a PowerDebug hardware module.
DebugCableA	Uses the whisker connected to a CombiProbe.

HostMCI mode

InfineonDAS0 Aurix only	Selects the Infineon DAS backend as interface. For a detailed description and examples, see “Debugging via Infineon DAS Server” (backend_das.pdf).
XCP0	Selects the XCP backend as interface. For a detailed description and examples, see “XCP Debug Back-End” (backend_xcp.pdf).
Unknown	No backend is selected. Debugging is not possible.

NEXUS.ARU

Control for ARU trace messages

MPC57xx, SPC58xx, RH850, S32, SR6

Format: NEXUS.ARU [ON | OFF]

Control for the NEXUS ARU trace messages.

NEXUS.ARUAccessX

ARU debugging address

MPC57xx, SPC58xx, RH850, S32, SR6

Format: NEXUS.ARUAccess0 <address>
NEXUS.ARUAccess1 <address>

<address>: 0x0...0x1FE

Defines the ARU debugging address of the two independent debug channels for the ARU trace messages.

NEXUS.FTM

Control for fetch trace messages

MPC57xx, SPC58xx, RH850, S32, SR6

Format: NEXUS.FTM [ON | OFF]

Control for the NEXUS fetch trace messages.

Format:	NEXUS.FTCE <channel>
<channel>:	0x00 0x01 0x02 ... 0x41 0xFF

The FTCE is a one hot bit field which defines the channel or channels to be monitored for fetch traces.

0x00	No channel selected for trace
0x01	Channel[0] selected
0x02	Channel[1] selected
0x41	Channel[0] and channel[6] selected
0xFF	All channels selected

Format:	NEXUS.DPLL [OFF Read Write ReadWrite]
---------	---

OFF (default)	No Digital PLL Module trace messages are output by NEXUS.
Read	NEXUS outputs Digital PLL Module trace messages for read accesses.
Write	NEXUS outputs Digital PLL Module trace messages for write accesses.
ReadWrite	NEXUS outputs Digital PLL Module trace messages for read and write accesses.

MPC57xx, SPC58xx, S32, SR6

Format:

NEXUS.DPLLMemory [RAM1A, RAM1B, RAM2]

This setting selects a DPLL memory module for Digital PLL data tracing.

MPC57xx, SPC58xx, RH850, S32, SR6

Format:

NEXUS.DTM [OFF | Read | Write | ReadWrite]

- OFF (default)

No data trace messages are output by NEXUS.
- Read

NEXUS outputs data trace messages for read accesses.
- Write

NEXUS outputs data trace messages for write accesses.
- ReadWrite

NEXUS outputs data trace messages for read and write accesses.

RH850

Format:

NEXUS.DTC <channel>

<channel>:

0 | 1 | 2 ... 7 | ALL

Selects the MCS Channel for data trace messages.

MPC57xx, SPC58xx, S32, SR6

Format:	NEXUS.DTCE <channel>
<channel>:	0x00
	0x01
	0x02
	...
	0xFF

The DTCE is a one hot bit field which defines the channel or channels to be monitored for data traces.

0x00	No channel selected for trace
0x01	Channel[0] selected
0x02	Channel[1] selected
0x2D	Channel[0], Channel[2], Channel[3] and Channel[5] selected

MPC57xx, SPC58xx, RH850, S32, SR6

Format:	NEXUS.OFF
---------	-----------

If the debugger is used stand-alone, the trace port is disabled by the debugger.

MPC57xx, SPC58xx, RH850, S32, SR6

Format:	NEXUS.ON
---------	----------

The NEXUS trace port is switched on. All trace registers are configured by debugger.

MPC57xx, SPC58xx, RH850, S32, SR6

Format:	NEXUS.RESet
---------	--------------------

Resets NEXUS trace port settings to default settings.

MPC57xx, SPC58xx, S32, SR6

Format:	NEXUS.RefClock [ON OFF]
---------	----------------------------------

Aurora NEXUS only. When set to ON, the preprocessor provides the reference clock for the Aurora NEXUS block on the processor. Only enable when the processor requires this reference clock and when no module provides the Aurora clock source for the processor.

MPC57xx, SPC58xx, RH850, S32, SR6

Format:	NEXUS.PortMode <mode>
<mode>:	625MBPS 750MBPS 850MBPS 1000MBPS 1250MBPS 1500MBPS 1700MBPS 2000MBPS 2500MBPS 3000MBPS 3125MBPS

Sets the NEXUS trace port frequency. For Aurora NEXUS, the setting is a fixed bit clock which is independent of the system frequency.

NOTE:	Set the bit clock according to the processor's data sheet. Automotive processors usually need an external reference clock for Aurora operation. The Aurora preprocessor can provide that clock signal. It is enabled using NEXUS.RefClock ON.
--------------	--

MPC57xx, SPC58xx, RH850, S32, SR6

Format:

NEXUS.PortSize <port_size>

<port_size>:

2Lane | 4Lane

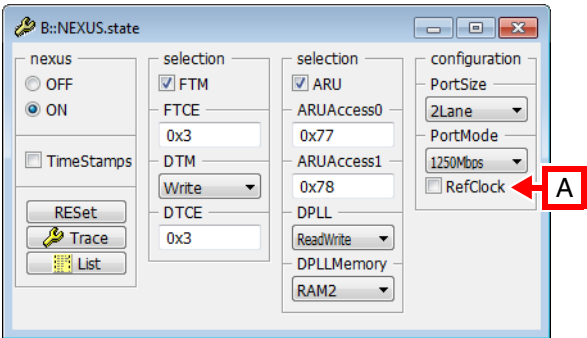
Sets the nexus Aurora lanes. The setting can only be changed if no debug session is active (SYStem.Down).

MPC57xx, SPC58xx, RH850, S32, SR6

Format:

NEXUS.state

Displays the Nexus configuration window NEXUS.state.



A For descriptions of the commands in the NEXUS.state window, please refer to the NEXUS.* commands in this chapter.

Example: For information about the RefClock check box, see NEXUS.RefClock.

RH850

Format:

NEXUS.SYNC [ON | OFF]

Forces NEXUS address-sync trace messaging on all data trace messages.

MPC57xx, SPC58xx, RH850, S32, SR6

Format:

NEXUS.TimeStamps [ON | OFF]

Forces the transmission of the Timestamp value along with the Nexus message data for all messages transmitted by the GTM.

NEXUS.TimeStampSel

Select TBU channel for timestamp trace messages

SR6 only

[build 152953 - DVD 02/2023]

Format:

NEXUS.TimeStampSel TBU0 | TBU1 | TBU2 | TBU3

This setting selects the TBU channel for timestamp value insertion in trace messages.

General TrOnchip Commands

The following **TrOnchip** commands apply to AURIX, MPC57xx, SPC58xx, RH850, S32 and SR6.

TrOnchip.state

Display onchip trigger window

AURIX, MPC57xx, SPC58xx, RH850, S32, SR6

Format:

TrOnchip.state [/<tab>]

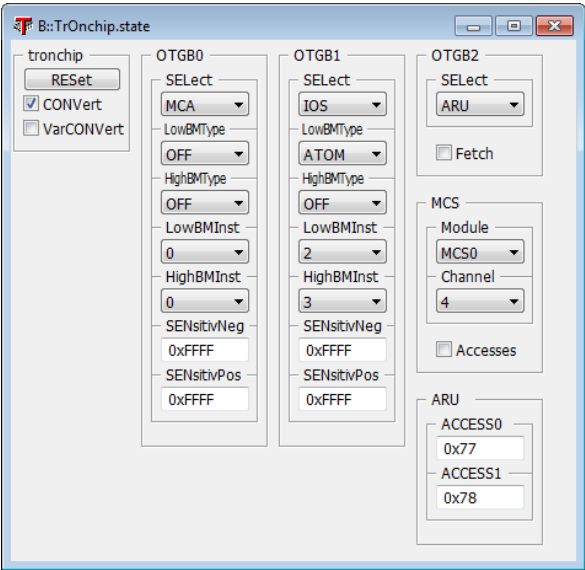
<tab>:

TIM | TOM | ATOM | SPE | DPLL | ARU | TBU

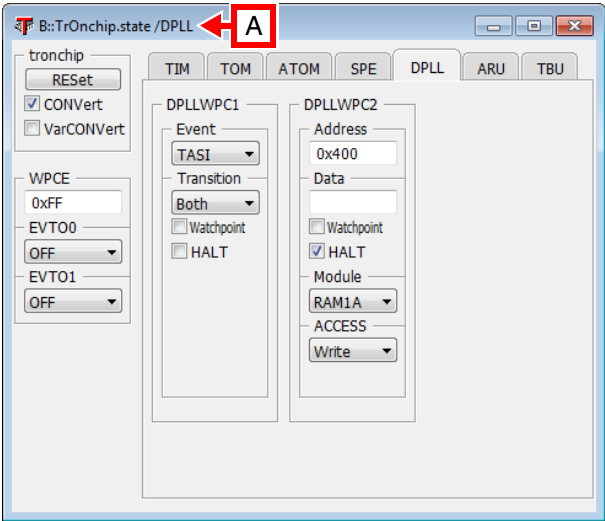
for PowerPC only

Displays the onchip trigger window **TrOnchip.state**.

TrOnchip.state window for TriCore:



TrOnchip.state window for PowerPC:



<tab>	Opens the TrOnchip.state window on the specified tab, see [A].
-------	---

Format:

TrOnchip.CONVert [ON | OFF] (deprecated)
Use Break.CONFIG.InexactAddress instead

The on-chip breakpoints can only cover specific ranges. If a range cannot be programmed into the breakpoint it will automatically be converted into a single address breakpoint when this option is active. This is the default. Otherwise an error message is generated.

```
TrOnchip.CONVert ON
Break.Set 0x1000--0x17ff /Write      ; sets breakpoint at range
Break.Set 0x1001--0x17ff /Write      ; 1000--17ff sets single breakpoint
...                                   ; at address 1001

TrOnchip.CONVert OFF
Break.Set 0x1000--0x17ff /Write      ; sets breakpoint at range
Break.Set 0x1001--0x17ff /Write      ; 1000--17ff
Break.Set 0x1001--0x17ff /Write      ; gives an error message
```

Format:

TrOnchip.RESet

Resets the trigger system to the default state.

Format:

TrOnchip.VarCONVert [ON | OFF] (deprecated)
Use Break.CONFIG.VarConvert instead

The on-chip breakpoints can only cover specific ranges. If you want to set a marker or breakpoint to a complex variable, the on-chip break resources of the CPU may be not powerful enough to cover the whole structure. If the option **TrOnchip.VarCONVert is ON** the breakpoint will automatically be converted into a single address breakpoint. This is the default setting. Otherwise an error message is generated.

TrOnchip.ARU

ARU settings

AURIX

TrOnchip.ARU.ACCESS

ARU debugging address

AURIX

Format:	TrOnchip.ARU.ACCESS0 <value> TrOnchip.ARU.ACCESS1 <value>
<value>:	0x0 ... 0x1FE

Defines the ARU debugging address of the two independent debug channels for the ARU trace messages.

Only visible in the trace listing if OTGB2 is set to ARU [TrOnchip.OTGB2 SElect ARU](#).

TrOnchip.MCS.Channel

Select the MCS channel

AURIX

Format:

TrOnchip.MCS.Channel <channel>

<channel>:

0 | 1 | 2 ... 7 | ALL

Selects the MCS Channel. Only visible in the trace listing if OTGB0 or OTGB1 is set to MCA or OTGB2 is set to MCD.

TrOnchip.MCS.Module

Select the MCS module

AURIX

Format:

TrOnchip.MCS.Channel <module>

<module>:

MCS0 | MCS1 | MCS2 ... MCS7

Selects the MCS Module for the Trace. Only visible in the trace listing if OTGB0 or OTGB1 is set to MCA or OTGB2 is set to MCD. The number of selectable module depends on the device. If more than one GTM GUI are open only one MCS module can be selected.

TrOnchip.OTGBx.SELect

Select trace source

Format:

TrOnchip.OTGB0.SELect [OFF | IOS | MCA | DRA1A | DRA1BC | DRA2]

TrOnchip.OTGB1.SELect [OFF | IOS | MCA | DRA1A | DRA1BC | DRA2]

- OFF (default)

No data trace messages are output by MCDS.
- IOS

IO and Other Signals.
- MCA

Allows to observe the PC of all or a selected MCS Channel.
- DRA1A

DPLL RAM Access address.
- DRA1BC

DPLL RAM Access address.
- DRA2

DPLL RAM Access address.

Format:	TrOnchip.OTGB0.LowBMType [OFF TIM TOML TOMH ATOM SPE MISC]
	TrOnchip.OTGB1.LowBMType [OFF TIM TOML TOMH ATOM SPE MISC]

IO Trigger Sets consist of the most important TIM, TOM and ATOM signals in groups of 8. In addition there is a group of SPE signals and one group for miscellaneous signals. The multiplexer allows to map arbitrary and different signal groups to the high and the low byte of a 16 bit Trigger Set.

Only if OTGBx is set to IOS. For all other selection this index option is ignored.

OFF (default)	No data trace messages are output by MCDS.
TIM	All 8 input signals from the Timer Input Module.
TOML	Lower 8 output signals TOM_OUT.
TOMH	Higher 8 output signals TOM_OUT.
ATOM	All 8 output signals ATOM_CHx_OUT.
SPE	Bits 0,1: SPE_NPD0, SPE_DIR0 Bits 2,3: SPE_NPD1, SPE_DIR1 Bits 4,5: SPE_NPD2, SPE_DIR2 Bits 6,7: SPE_NPD3, SPE_DIR3.
MISC	Bit 0: DPLL TASI Bit 1: DPLL SASI Bit 4: TBU0 trigger (OTBU0T) Bit 5: TBU1 trigger (OTBU1T) Bit 6: TBU2 trigger (OTBU2T)

AURIX

Format:	TrOnchip.OTGB0.HighBMType
	TrOnchip.OTGB1.HighBMType

Independent selection with same options as for [TrOnchip.OTGBx.LowBMType](#).

AURIX

Format:	TrOnchip.OTGB0.LowBMInst [0 1 2 ... 15]
	TrOnchip.OTGB1.LowBMInst [0 1 2 ... 15]

Index of the module instance. Index starts with 0, the max. value depends on GTM configuration and module type. For SPE and MISC signals this index is ignored.

AURIX

Format:	TrOnchip.OTGB0.HighBMInst [0 1 2 ... 15]
	TrOnchip.OTGB1.HighBMInst [0 1 2 ... 15]

Index of the module instance. Index starts with 0, the max. value depends on GTM configuration and module type. For SPE and MISC signals this index is ignored.

Format:	TrOnchip.OTGB0.SENSitivNeg
	TrOnchip.OTGB1.SENSitivNeg

The edge sensitivity is controlled for each bit individually with **TrOnchip.OTGBx.SENSitivPos** for positive edges and **TrOnchip.OTGBx.SENSitivNeg** for negative edges. The underlaying concept is that the OTGB0/1 value is always valid. If for specific Traces this is not the case, one OTGB0/1 bit needs to have the valid signal functionality by changing for each OTGB0/1 sample point.

Format:	TrOnchip.OTGB0.SENSitivPos
	TrOnchip.OTGB1.SENSitivPos

The edge sensitivity is controlled for each bit individually with **TrOnchip.OTGBx.SENSitivPos** for positive edges and **TrOnchip.OTGBx.SENSitivNeg** for negative edges. The underlaying concept is that the OTGB0/1 value is always valid. If for specific Traces this is not the case, one OTGB0/1 bit needs to have the valid signal functionality by changing for each OTGB0/1 sample point.

TrOnchip.OTGB2.SELect

Select trace source

AURIX

Format:

TrOnchip.OTGB2.SELect [OFF | ARU | MCD | DRD1A | DRD1BC | DRD2 | TTB0 | TTB1 | TTB2]

OFF (default)	No data trace messages are output by MCDS.
ARU	ARU Data.
MCD	MCS Data
DRD1A/1BC/2	DPLL RAM access Data.
TTB0/1/2	Current TBU_TS0/1/2 timestamp.

TrOnchip.OTGBM0.SELect

Select trace source

AURIX2G

Format:

TrOnchip.OTGBM0.SELect [OFF | MCA | DRA1A | DRA1BC | DRA2 | ARU | TTB0 | TTB1 | TTB2 | TTB3]

- OFF (default)

No trace messages are output by MCDS.
- MCA

Allows to observe the PC of all or a selected MCS Channel.
- DRA1A/1BC/2

DPLL RAM access address.
- ARU

ARU Data.
- TTB0/1/2/3

Current TBU_TS0/1/2/3 timestamp.

TrOnchip.OTGBM1.SELect

Select trace source

AURIX2G

Format:

TrOnchip.OTGBM1.SELect [OFF | MCD | DRD1A | DRD1BC | DRD2 | ARU]

OFF (default)	No trace messages are output by MCDS.
MCD	MCS Data
DRD1A/1BC/2	DPLL RAM access Data.
ARU	ARU Data.

TrOnchip.ARUx.Address

ARU address compare

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.ARU0.Address [*<address>*]

TrOnchip.ARU1.Address [*<address>*]

Defines the address value that is compared against the address for write and read ARU accesses. This are the same addresses which are used for ARU trace messages

TrOnchip.ARUx.DataHigh

ARU data low value compare

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.ARU0.DataHigh [*<data>*]

TrOnchip.ARU1.DataHigh [*<data>*]

Defines the data high value (29 bit) that is compared against the data for write and read ARU accesses. If no data value is used the data compare is disabled.

TrOnchip.ARUx.DataLow

ARU data low value compare

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.ARU0.DataLow [*<data>*]

TrOnchip.ARU1.DataLow [*<data>*]

Defines the data low value (29 bit) that is compared against the data for write and read ARU accesses. If no data value is used the data compare is disabled.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.ARU0.HALT [ON OFF]
	TrOnchip.ARU1.HALT [ON OFF]

This setting controls if the ARU access is enabled to halt the GTM module.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.ARU0.Watchpoint [ON OFF]
	TrOnchip.ARU1.Watchpoint [ON OFF]

If this is ON, it enables the ARU selected access event to generate a watchpoint message.

TrOnchip.ATOMWPCx.Channel

ATOM channel selection

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.ATOMWPC1.Channel [CH0 | CH1 | CH2 ... | CH7]

TrOnchip.ATOMWPC2.Channel [CH0 | CH1 | CH2 ... | CH7]

Selects which channel within a selected ATOM sub-module generates watchpoints or trigger.

TrOnchip.ATOMWPCx.HALT

ATOM halt enable

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.ATOMWPC1.HALT [ON | OFF]

TrOnchip.ATOMWPC2.HALT [ON | OFF]

This setting controls if the selected ATOM Channel is used to halt the GTM module.

TrOnchip.ATOMWPCx.Module

ATOM sub-module selection

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.ATOMWPC1.Module [ATOM0 | ATOM1 | ATOM2 ... | ATOM7]

TrOnchip.ATOMWPC2.Module [ATOM0 | ATOM1 | ATOM2 ... | ATOM7]

Selects the ATOM sub-module source selection for Trigger or Watchpoint. The number of module depends on the device.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.ATOMWPC1.TIMING [ON OFF]
	TrOnchip.ATOMWPC2.TIMING [ON OFF]

If this is ON, it enables valid transitions on the selected ATOM filter output to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing ATOM1 ATOM2](#) diagram.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.ATOMWPC1.Transition [Both Positive Negative]
	TrOnchip.ATOMWPC2.Transition [Both Positive Negative]

Selects the slope for the ATOM selected channel for Trigger or Watchpoint.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.ATOMWPC1.Watchpoint [ON OFF]
	TrOnchip.ATOMWPC2.Watchpoint [ON OFF]

If this is ON, it enables valid transitions on the selected ATOM filter output to generate a watchpoint message.

MPC57xx, SPC58xx, S32, SR6

TrOnchip.DPLLWPC1.Event

DPLL source selection

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.DPLLWPC1.Event [TASI | SASI]

Selects if a watchpoint or trigger is issued based on TASI or SASI event.

TrOnchip.DPLLWPC1.HALT

DPLL TASI/SASI halt enable

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.DPLLWPC1.HALT [ON | OFF]

This setting controls if the DPLL SASI or TASI selected signal is enabled to halt the GTM module.

TrOnchip.DPLLWPC1.Transition

DPLL TASI/SASI slope selection

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.DPLLWPC1.Transition [Both | Positive | Negative]

Selects the slope for the DIR signal for Trigger or Watchpoint.

TrOnchip.DPLLWPC1.Watchpoint

DPLL TASI/SASI watchpoint enable

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.DPLLWPC1.Watchpoint [ON | OFF]

If this is ON, it enables valid transitions on DPLL SASI or TASI to generate a watchpoint message.

MPC57xx, SPC58xx, S32, SR6

TrOnchip.DPLLWPC2.Address

DPLL RAM address compare

MPC57xx, SPC58xx, S32, SR6

Format:TrOnchip.DPLLWPC2.Address [<address> | <address_mask>]

Defines the address value that is compared against the address for write and read DPLL RAM accesses.

TrOnchip.DPLLWPC2.ACCESS

DPLL RAM read/write control

MPC57xx, SPC58xx, S32, SR6

Format:TrOnchip.DPLLWPC2.ACCESS [ReadWrite | Write | Read]

This setting selects the RAM access type to mach on.

TrOnchip.DPLLWPC2.Data

DPLL RAM data compare

MPC57xx, SPC58xx, S32, SR6

Format:TrOnchip.DPLLWPC2.Data [<data>]

Defines the data value that is compared against the data for write and read DPLL RAM accesses. If no data value is used the data compare is disabled.

TrOnchip.DPLLWPC2.HALT

DPLL RAM access halt enable

MPC57xx, SPC58xx, S32, SR6

Format:TrOnchip.DPLLWPC2.HALT [ON | OFF]

This setting controls if the DPLL RAM access is enabled to halt the GTM module.

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.DPLLWPC2.Module [RAM1A | RAM1B | RAM2]

This setting selects if a watchpoint or trigger is issued based on the RAM1a, RAM1bc or RAM2 access.

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.DPLLWPC2.Watchpoint [ON | OFF]

If this is ON, it enables the DPLL RAM selected access event to generate a watchpoint message.

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.EVTO0 [OFF | HALT | TIM | TOM | ATOM | SPE0 | SPE1 | ARU | DPLL | MCSA | MCSB | TBU0 | TBU1 | TBU2]

TrOnchip.EVTO1 [OFF | HALT | TIM | TOM | ATOM | SPE0 | SPE1 | ARU | DPLL | MCSA | MCSB | TBU0 | TBU1 | TBU2]

Selects the EVTOx output.

MPC57xx, SPC58xx, S32, SR6

TrOnchip.SPEx.DIR

SPEx DIR watchpoint settings

MPC57xx, SPC58xx, S32, SR6

TrOnchip.SPEx.DIR.HALT

SPEx DIR halt enable

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.SPE0.DIR.HALT [ON | OFF]

TrOnchip.SPE1.DIR.HALT [ON | OFF]

This setting controls if the SPEA/SPEB DIR active slope event is enabled to halt the GTM module.

TrOnchip.SPEx.DIR.TIMING

SPEx DIR watchpoint enable

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.SPE0.DIR.TIMING [ON | OFF]

TrOnchip.SPE1.DIR.TIMING [ON | OFF]

If this is ON, it enables valid transitions on SPEA/SPEB DIR to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing SPE0_DIR SPE1_DIR](#) diagram.

TrOnchip.SPEx.DIR.Transition

SPEx DIR slope selection

MPC57xx, SPC58xx, S32, SR6

Format:

TrOnchip.SPE0.DIR.Transition [Both | Positive | Negative]

TrOnchip.SPE1.DIR.Transition [Both | Positive | Negative]

Selects the slope for the DIR signal for Trigger or Watchpoint.

Format: **TrOnchip.SPE0.DIR.Watchpoint** [ON | OFF]
 TrOnchip.SPE1.DIR.Watchpoint [ON | OFF]

If this is ON, it enables valid transitions on SPEA/SPEB DIR to generate a watchpoint message.

MPC57xx, SPC58xx, S32, SR6

TrOnchip.SPEx.NIPD.HALT

SPEx NIPD halt enable

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.SPE0.NIPD.HALT [ON OFF]
	TrOnchip.SPE1.NIPD.HALT [ON OFF]

This setting controls if the SPEA/SPEB NIPD active slope event is enabled to halt the GTM module.

TrOnchip.SPEx.NIPD.TIMING

SPEx NIPD watchpoint enable

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.SPE0.NIPD.TIMING [ON OFF]
	TrOnchip.SPE1.NIPD.TIMING [ON OFF]

If this is ON, it enables valid transitions on SPEA/SPEB NIPD to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing SPE0_NIPD SPE1_NIPD](#) diagram.

TrOnchip.SPEx.NIPD.Transition

SPEx NIPD slope selection

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.SPE0.NIPD.Transition [Both Positive Negative]
	TrOnchip.SPE1.NIPD.Transition [Both Positive Negative]

Selects the slope for the NIPD signal for Trigger or Watchpoint.

Format: **TrOnchip.SPE0.NIPD.Watchpoint [ON | OFF]**
TrOnchip.SPE1.NIPD.Watchpoint [ON | OFF]

If this is ON, it enables valid transitions on SPEA/SPEB NIPD to generate a watchpoint message.

TrOnchip.TBUx.Data

TBU data value compare

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TBU0.Data [<i><data></i>]
	TrOnchip.TBU1.Data [<i><data></i>]
	TrOnchip.TBU2.Data [<i><data></i>]

Defines the data value that is compared against the data for TBUx accesses. If no data value is used the data compare is disabled.

TrOnchip.TBUx.HALT

TBU access halt enable

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TBU0.HALT [ON OFF]
	TrOnchip.TBU1.HALT [ON OFF]
	TrOnchip.TBU2.HALT [ON OFF]

This setting controls if the TBUx access is enabled to halt the GTM module.

TrOnchip.TBUx.Watchpoint

TBU access watchpoint enable

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TBU0.Watchpoint [ON OFF]
	TrOnchip.TBU1.Watchpoint [ON OFF]
	TrOnchip.TBU2.Watchpoint [ON OFF]

If this is ON, it enables the TBUx selected access event to generate a watchpoint message.

Format:	TrOnchip.TBU0.SELect [24BIT 27BIT 24BITData 27BITData]
---------	--

Controls the type of comparison used for TBU0 watchpoint or trigger generation.

MPC57xx, SPC58xx, S32, SR6

TrOnchip.TIMWPCx.Channel

TIM channel selection

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TIMWPC1.Channel [CH0 CH1 CH2 ... CH7]
	TrOnchip.TIMWPC2.Channel [CH0 CH1 CH2 ... CH7]

Selects which channel within a selected TIM sub-module generates watchpoints or trigger.

TrOnchip.TIMWPCx.HALT

TIM halt enable

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TIMWPC1.HALT [ON OFF]
	TrOnchip.TIMWPC2.HALT [ON OFF]

This setting controls if the selected TIM Channel is used to halt the GTM module.

TrOnchip.TIMWPCx.Module

TIM sub-module selection

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TIMWPC1.Module [TIM0 TIM1 TIM2 ... TIM7]
	TrOnchip.TIMWPC2.Module [TIM0 TIM1 TIM2 ... TIM7]

Selects the TIM sub-module source selection for Trigger or Watchpoint. The number of module depends on the device.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TIMWPC1.TIMING [ON OFF]
	TrOnchip.TIMWPC2.TIMING [ON OFF]

If this is ON, it enables valid transitions on the selected TIM filter output to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing TIM1 TIM2](#) diagram.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TIMWPC1.Transition [Both Positive Negative]
	TrOnchip.TIMWPC2.Transition [Both Positive Negative]

Selects the slope for the TIM selected channel for Trigger or Watchpoint.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TIMWPC1.Watchpoint [ON OFF]
	TrOnchip.TIMWPC2.Watchpoint [ON OFF]

If this is ON, it enables valid transitions on the selected TIM filter output to generate a watchpoint message.

MPC57xx, SPC58xx, S32, SR6

TrOnchip.TOMWPCx.Channel

TOM channel selection

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TOMWPC1.Channel [CH0 CH1 CH2 ... CH15]
	TrOnchip.TOMWPC2.Channel [CH0 CH1 CH2 ... CH15]

Selects which channel within a selected TOM sub-module generates watchpoints or trigger.

TrOnchip.TOMWPCx.HALT

TOM halt enable

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TOMWPC1.HALT [ON OFF]
	TrOnchip.TOMWPC2.HALT [ON OFF]

This setting controls if the selected TOM Channel is used to halt the GTM module.

TrOnchip.TOMWPCx.Module

TOM sub-module selection

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TOMWPC1.Module [TOM0 TOM1 TOM2 ... TOM7]
	TrOnchip.TOMWPC2.Module [TOM0 TOM1 TOM2 ... TOM7]

Selects the TOM sub-module source selection for Trigger or Watchpoint. The number of module depends on the device.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TOMWPC1.TIMING [ON OFF]
	TrOnchip.TOMWPC2.TIMING [ON OFF]

If this is ON it enables valid transitions on the selected TOM Filter output to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing TOM1 TOM2](#) diagram.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TOMWPC1.Transition [Both Positive Negative]
	TrOnchip.TOMWPC2.Transition [Both Positive Negative]

Selects the slope for the TOM selected channel for Trigger or Watchpoint.

MPC57xx, SPC58xx, S32, SR6

Format:	TrOnchip.TOMWPC1.Watchpoint [ON OFF]
	TrOnchip.TOMWPC2.Watchpoint [ON OFF]

If this is ON it enables valid transitions on the selected TOM filter output to generate a watchpoint message.

Format:	TrOnchip.WPCE <channel>
<channel>:	0x00 0x01 0x02 ... 0xFF

The WPCE is a one hot bit field which defines the channel or channels to be used for breakpoints.

0x00	No channel selected for breakpoints
0x01	Channel[0] selected
0x02	Channel[1] selected
0xFF	All channels selected

TrOnchip.TIOINWPCx

TIO input watchpoint settings

S32 only

TrOnchip.TIOINWPCx.Channel

TIO input channel selection

S32 only

Format:

TrOnchip.TIOINWPC1.Channel [CH0 | CH1 | CH2 ... | CH7]

TrOnchip.TIOINWPC2.Channel [CH0 | CH1 | CH2 ... | CH7]

Selects which channel within a selected TIO sub-module generates watchpoints or trigger.

TrOnchip.TIOINWPCx.HALT

TIO input halt enable

S32 only

Format:

TrOnchip.TIOINWPC1.HALT [ON | OFF]

TrOnchip.TIOINWPC2.HALT [ON | OFF]

Controls if the selected TIO Channel is used to halt the GTM module.

TrOnchip.TIOINWPCx.Module

TIO input sub-module selection

S32 only

Format:

TrOnchip.TIOINWPC1.Module [TIO1_G0 | TIO1_G1 | TIO1_G2 | TIO2_G0 | TIO2_G1 | TIO2_G2 | TIO3_G0 | TIO3_G1 | TIO3_G2]

TrOnchip.TIOINWPC2.Module [TIO1_G0 | TIO1_G1 | TIO1_G2 | TIO2_G0 | TIO2_G1 | TIO2_G2 | TIO3_G0 | TIO3_G1 | TIO3_G2]

Selects the TIO sub-module source selection for Trigger or Watchpoint. The number of module depends on the device.

S32 only

Format:	TrOnchip.TIOINWPC1.TIMING [ON OFF]
	TrOnchip.TIOINWPC2.TIMING [ON OFF]

If this is ON, it enables valid transitions on the selected TIO filter input to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing TIO1 TIO2](#) diagram.

S32 only

Format:	TrOnchip.TIOINWPC1.Transition [Both Positive Negative]
	TrOnchip.TIOINWPC2.Transition [Both Positive Negative]

Selects the slope for the TIO selected channel for Trigger or Watchpoint.

S32 only

Format:	TrOnchip.TIOINWPC1.Watchpoint [ON OFF]
	TrOnchip.TIOINWPC2.Watchpoint [ON OFF]

If this is ON, it enables valid transitions on the selected TIO filter input to generate a watchpoint message.

S32 only

TrOnchip.TIOOUTWPCx.Channel

TIO output channel selection

S32 only

Format:	TrOnchip.TIOOUTWPC1.Channel [CH0 CH1 CH2 ... CH7]
	TrOnchip.TIOOUTWPC2.Channel [CH0 CH1 CH2 ... CH7]

Selects which channel within a selected TIO sub-module generates watchpoints or trigger.

TrOnchip.TIOOUTWPCx.HALT

TIO output halt enable

S32 only

Format:	TrOnchip.TIOOUTWPC1.HALT [ON OFF]
	TrOnchip.TIOOUTWPC2.HALT [ON OFF]

Controls if the selected TIO Channel is used to halt the GTM module.

TrOnchip.TIOOUTWPCx.Module

TIO output sub-module selection

S32 only

Format:	TrOnchip.TIOOUTWPC1.Module [TIO1_G0 TIO1_G1 TIO1_G2 TIO2_G0 TIO2_G1 TIO2_G2 TIO3_G0 TIO3_G1 TIO3_G2]
	TrOnchip.TIOOUTWPC2.Module [TIO1_G0 TIO1_G1 TIO1_G2 TIO2_G0 TIO2_G1 TIO2_G2 TIO3_G0 TIO3_G1 TIO3_G2]

Selects the TIO sub-module source selection for Trigger or Watchpoint. The number of module depends on the device.

S32 only

Format:	TrOnchip.TIOOUTWPC1.Polarity [Positive Negative]
	TrOnchip.TIOOUTWPC2.Polarity [Positive Negative]

Selects the polarity of corresponding GTM signal.

S32 only

Format:	TrOnchip.TIOOUTWPC1.TIMING [ON OFF]
	TrOnchip.TIOOUTWPC2.TIMING [ON OFF]

If this is ON, it enables valid transitions on the selected TIO filter output to generate a watchpoint message. These watchpoint messages are not visible in the [Trace.List](#) window. These watchpoint messages are only necessary for the [Trace.Timing TIO1 TIO2](#) diagram.

S32 only

Format:	TrOnchip.TIOOUTWPC1.Transition [Both Positive Negative]
	TrOnchip.TIOOUTWPC2.Transition [Both Positive Negative]

Selects the slope for the TIO selected channel for Trigger or Watchpoint.

S32 only

Format:	TrOnchip.TIOOUTWPC1.Watchpoint [ON OFF]
	TrOnchip.TIOOUTWPC2.Watchpoint [ON OFF]

If this is ON, it enables valid transitions on the selected TIO filter output to generate a watchpoint message.

TrOnchip.BreakChannel

Select the channel for breakpoints

Rh850

Format:	TrOnchip.BreakChannel [<i><channel></i>]
<i><channel></i> :	0 1 2 ... 9 ALL

Selects a MCS channel on which an onchip breakpoint operates.

TrOnchip.ATOMSlotx

Select the ATOM module for trace

Rh850

Format:	TrOnchip.ATOMSlot0 [<i><module></i>] TrOnchip.ATOMSlot1 [<i><module></i>] TrOnchip.ATOMSlot2 [<i><module></i>] TrOnchip.ATOMSlot3 [<i><module></i>]
<i><module></i> :	OFF ATOM0 ATOM1 ATOM2 ... ATOM8

Selects a ATOM module for tracing the output states. Every slot trace eight lines from one ATOM module.

TrOnchip.TIMSlotx

Select the TIM module for trace

Rh850

Format:	TrOnchip.TIMSlot0 [<i><module></i>] TrOnchip.TIMSlot1 [<i><module></i>] TrOnchip.TIMSlot2 [<i><module></i>] TrOnchip.TIMSlot3 [<i><module></i>]
<i><module></i> :	OFF TIM0 TIM1 TIM2 ... TIM6

Selects a TIM module for tracing the input states. Every slot trace eight lines from one TIM module.

Mechanical Description

Debug access is always performed via the debug port of the main core debugger.

- For the TriCore, this is the JTAG or DAP connector. For detailed information, see chapter [“JTAG Connector”](#) (debugger_tricore.pdf) and [“Application Note Debug Cable TriCore”](#) (app_tricore_ocds.pdf).
- For the MPC57xx or SPC58xx, this is the JTAG connector. For detailed information, see chapter [“Debug and Trace Connectors”](#) (debugger_mpc5500.pdf).
- For RH850, this is the Debug connector. For detailed information, see chapter [“Debug Connector”](#) (debugger_rh850.pdf).