LAUTERBACH
DEVELOPMENT TOOLS

General Commands Reference Guide H

MANUAL

General Commands Reference Guide H

General Commands Reference Guide H

Version 06-Jun-2024

HAnalyzer — Host analyzer

The **HAnalyzer**, one of the trace methods provided by TRACE32, is used for target program or data trace analysis. Additional external hardware is typically not necessary, and the host PC provides the memory to store the trace data.

The trace method **HAnalyzer** is only available if the target board supports a trace stream using fast protocols such as USB or Ethernet. Using the trace method **HAnalyzer**, TRACE32 will handle the trace connection via the configured protocol and use the host PC's memory for the trace data.

The trace features of the **HAnalyzer** can be configured and controlled with the command group of the same name: **HAnalyzer**.

Aside from the command group **HAnalyzer**, the more general command group **Trace** can be used to set up and handle the information provided by the **HAnalyzer**. Precondition is that the trace method **HAnalyzer** is selected with the **Trace.METHOD HAnalyzer** command, as shown in this example:

```
HAnalyzer.SIZE 1000000000.            ;specify a size
Trace.METHOD HAnalyzer
;trace data is recorded using the commands Go, WAIT, Break

Trace.List              ;Display the trace data recorded with HAnalyzer
                        ;as a trace listing.
HAnalyzer.List          ;This is the equivalent and explicit command.
```

See also

- Trace.METHOD

- 'Generic HAnalyzer Trace Commands' in 'General Commands Reference Guide H'
- 'HAnalyzer-specific Trace Command' in 'General Commands Reference Guide H'

HAnalyzer.Mode — Set the trace operation mode

Format: **HAnalyzer.Mode** [*<mode>*]

<mode>:
Fifo
Stack
Pipe

Selects the trace operation mode.

Fifo — If the trace is full, new records will overwrite older records. The trace records always the last cycles before the break.

Stack — If the trace is full recording will be stopped. The trace always records the first cycles after starting the trace.

Pipe — The trace data is immediately conveyed to the host and distributed to user-defined trace sinks. Not supported with PowerTrace Ethernet 256/512MB. See **<trace>.PipeWRITE**.

See also

- <trace>.Mode

HAnalyzer.PipeWRITE — Define a named pipe as trace sink

Format: **HAnalyzer.PipeWRITE** *<pipe_name>* [*/<options>*]

<options>:
ChannelID *<channel_id>*
MasterID *<master_id>*
XtiMaster DSP | **CPU** | **MCU** (XTIv2)
XtiMaster DSP | **CPU1** | **CPU2** (SDTI)
Payload

This command is used to define a Windows or Unix named pipe as trace sink. Up to 8 named pipes can be defined as trace sinks simultaneously.

The named pipe has to be created by the receiving application, before you can connect to the named pipe. If the pipe is not already connected to a receiving application, the debugger software will report an error.

<pipe_name>	If you use this command without specifying a *<pipe_name>*, all open files currently used as trace sinks are closed.
ChannelID **MasterID**	If you record MIPIs STP trace (System Trace Protocol), then the options **/ChannelID** and **/MasterID** are available. You can use this options to only store messages into the file, which match the given ChannelID or MasterID. You can specify a single value, a range of values or a bitmask for the **ChannelID** and **MasterID**. If you record ARMs ITM trace, the **MasterID** option is not available, because ITM does not use master IDs.
Payload	The **/Payload** option specifies, that only the payload of the ITM or STP messages is stored into the file.

HAnalyzer.state — Display HAnalyzer trace configuration window

Format: **HAnalyzer.state**

Displays the **HAnalyzer.state** window, where you can configure the **HAnalyzer** trace.

Generic HAnalyzer Trace Commands

HAnalyzer.ACCESS Define access path to program code for trace decoding

See command **<trace>.ACCESS** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 131).

HAnalyzer.Arm Arm the trace

See command **<trace>.Arm** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 134).

HAnalyzer.AutoArm Arm automatically

See command **<trace>.AutoArm** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 135).

HAnalyzer.AutoInit Automatic initialization

See command **<trace>.AutoInit** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 140).

HAnalyzer.BookMark Set a bookmark in trace listing

See command **<trace>.BookMark** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 140).

HAnalyzer.BookMarkToggle Toggles a single trace bookmark

See command **<trace>.BookMarkToggle** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 143).

HAnalyzer.Chart — Display trace contents graphically

See command **<trace>.Chart** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 144).

HAnalyzer.CLOCK — Clock to calculate time out of cycle count information

See command **<trace>.CLOCK** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 191).

HAnalyzer.ComPare — Compare trace contents

See command **<trace>.ComPare** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 192).

HAnalyzer.ComPareCODE — Compare trace with memory

See command **<trace>.ComPareCODE** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 194).

HAnalyzer.DISable — Disable the trace

See command **<trace>.DISable** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 197).

HAnalyzer.DRAW — Plot trace data against time

See command **<trace>.DRAW** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 201).

HAnalyzer.EXPORT — Export trace data for processing in other applications

See command **<trace>.EXPORT** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 212).

HAnalyzer.FILE — Load a file into the file trace buffer

See command **<trace>.FILE** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 233).

HAnalyzer.Find — Find specified entry in trace

See command **<trace>.Find** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 235).

HAnalyzer.FindAll — Find all specified entries in trace

See command **<trace>.FindAll** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 237).

HAnalyzer.FindChange — Search for changes in trace flow

See command **<trace>.FindChange** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 238).

HAnalyzer.FindProgram — Advanced trace search

See command **<trace>.FindProgram** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 239).

HAnalyzer.FindReProgram — Activate advanced existing trace search program

See command **<trace>.FindReProgram** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 240).

HAnalyzer.FindViewProgram — State of advanced trace search programming

See command **<trace>.FindViewProgram** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 240).

HAnalyzer.FLOWPROCESS — Process flowtrace

See command **<trace>.FLOWPROCESS** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 241).

HAnalyzer.FLOWSTART — Restart flowtrace processing

See command **<trace>.FLOWSTART** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 241).

HAnalyzer.Get — Display input level

See command **<trace>.Get** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 242).

HAnalyzer.GOTO — Move cursor to specified trace record

See command **<trace>.GOTO** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 244).

HAnalyzer.Init — Initialize trace

See command **<trace>.Init** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 246).

HAnalyzer.List — List trace contents

See command **<trace>.List** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 248).

HAnalyzer.ListNesting — Analyze function nesting

See command **<trace>.ListNesting** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 263).

HAnalyzer.ListVar — List variable recorded to trace

See command **<trace>.ListVar** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 266).

HAnalyzer.LOAD — Load trace file for offline processing

See command **<trace>.LOAD** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 270).

HAnalyzer.OFF — Switch off

See command **<trace>.OFF** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 278).

HAnalyzer.PROfileChart — Profile charts

See command **<trace>.PROfileChart** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 283).

HAnalyzer.PROfileSTATistic — Statistical analysis in a table versus time

See command **<trace>.PROfileSTATistic** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 322).

HAnalyzer.REF — Set reference point for time measurement

See command **<trace>.REF** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 357).

HAnalyzer.RESet — Reset command

See command **<trace>.RESet** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 357).

HAnalyzer.SAVE — Save trace for postprocessing in TRACE32

See command **<trace>.SAVE** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 358).

HAnalyzer.SIZE — Define buffer size

See command **<trace>.SIZE** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 373).

HAnalyzer.STATistic — Statistic analysis

See command **<trace>.STATistic** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 378).

HAnalyzer.Timing — Waveform of trace buffer

See command **<trace>.Timing** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 499).

HAnalyzer.TraceCONNECT — Select on-chip peripheral sink

See command **<trace>.TraceCONNECT** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 501).

HAnalyzer.TRACK — Set tracking record

See command **<trace>.TRACK** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 502).

HAnalyzer.View — Display single record

See command **<trace>.View** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 504).

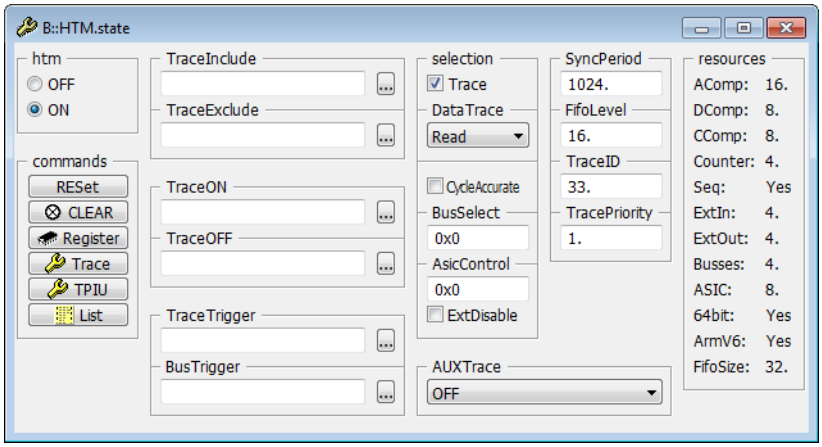
HAnalyzer.ZERO — Align timestamps of trace and timing analyzers

See command **<trace>.ZERO** in 'General Commands Reference Guide T' (general_ref_t.pdf, page 505).

HTM — CoreSight HTM (AHB Trace Macrocell)

CoreSight HTM (AHB Trace Macrocell) provides address and data trace information on AHB bus accesses.

The **HTM** command group is used to configure the trace source HTM. To perform the configuration, you can use the TRACE32 command line, a PRACTICE script (*.cmm), or the **HTM.state** window.



To analyze the recorded trace data, use the **HTM<trace>** command groups.

See also

- HTM.AsicControl
- HTM.AuxTrace
- HTM.AXIFifoClock
- HTM.AXIMaster
- HTM.BusSelect
- HTM.BusTrigger
- HTM.CLEAR
- HTM.CLOCK
- HTM.CycleAccurate
- HTM.DataTrace
- HTM.ExtDisable
- HTM.FifoLevel
- HTM.OFF
- HTM.ON
- HTM.PortRoute
- HTM.Register
- HTM.RESet
- HTM.Set
- HTM.state
- HTM.SyncPeriod
- HTM.Trace
- HTM.TraceExclude
- HTM.TraceID
- HTM.TraceInclude
- HTM.TraceOFF
- HTM.TraceON
- HTM.TracePriority
- HTM.TraceTrigger

▲ 'HTM<trace> - Trace Data Analysis' in 'General Commands Reference Guide H'

HTM.AsicControl — Set HTMASICCONTROL register

Format: **HTM.AsicControl** *<code>*

Informs TRACE32 about the target settings of the HTMASICCONTROL register.

See also

- HTM
- HTM.state

Format: **HTM.AuxTrace OFF** |
HML+HPROT0+HTRANS+HW+HRESP+WS |
HPROT01+HTRANS+HW+HRESP+WS |
…

Specifies signals for the auxiliary packets. Please refer to the *HTM Technical Reference Manual* for details.

See also

- HTM
- HTM.state

HTM.AXIFifoClock AXI FIFO clock for WPT HTM

Format: **HTM.AXIFifoClock 1/2** | **1/3** | **1/4**

Defines the AXI FIFO Clock for the NXP proprietary trace module WPT (**SYStem.CONFIG.HTM.Type WPT**).

See also

- HTM
- HTM.state

HTM.AXIMaster AXI master for WPT HTM

Format: **HTM.AXIFifoClock** *<master>*

Defines the AXI master for the NXP proprietary trace module WPT (**SYStem.CONFIG.HTM.Type WPT**).

See also

- HTM
- HTM.state

Format: **HTM.BusSelect** *<bus>*

Informs TRACE32 about the target settings of the HTMBUSSELECT register.

See also

- HTM
- HTM.state

HTM.BusTrigger Bus trigger definition

Format: **HTM.BusTrigger** *<address>* | *<range>* *<access>*

<access>: **Fetch** | **Access** | **Read** | **Write** | **Any**

Generates a trigger on HTMEXTOUT0, if the specified event occurs.

Example:

```
HTM.BusTrigger 0x1000 Write
```

See also

- HTM
- HTM.state

HTM.CLEAR Clear HTM.Set settings

Format: **HTM.CLEAR**

Switches the HTM ON, clears the trace and all setting done by the command **HTM.Set**.

See also

- HTM
- HTM.state

Format: **HTM.CLOCK** *<core_clock>*

Used to calculate absolute timings out of cycle count packets (**HTM.CycleAccurate ON**).

See also

- HTM
- HTM.state

HTM.CycleAccurate Cycle accurate tracing

Format: **HTM.CycleAccurate** [**ON** | **OFF**]
HTM.CycleTrace (deprecated)

OFF	Trace information is time-stamped by TRACE32.
ON	Cycle count packets are inserted into the trace stream. The calculation of the time stamps is based on: • The cycle count information • And the core clock specified with the command **Trace.CLOCK** *<core_clock>*.

Example:

```
Trace.Clock 50.MHz

HTM.CycleAccurate ON
```

See also

- HTM
- HTM.state

Format:	**HTM.DataTrace ON** | **OFF** | *<data>*
<data>:	**Address** | **ReadAddress** | **WriteAddress** | **Data** | **Read** | **Write**

Defines how data accesses are broadcast.

OFF	No information about data accesses is broadcast.
ON	The address and data information is broadcast for all data accesses.
Address	Only the address information for data accesses is broadcast.
ReadAddress	Only the address information for read accesses is broadcast.
WriteAddress	Only the address information for write accesses is broadcast.
Data	Only the data information for data accesses is broadcast.
Read	The address and data information is broadcast for read accesses.
Write	The address and data information is broadcast for write accesses.

Example:

```
HTM.DataTrace Read
```

See also

- HTM
- HTM.state

HTM.ExtDisable Set EXTDISABLE bit

Format:	**HTM.ExtDisable** [**ON** | **OFF**]

Informs TRACE32 about the target settings of the EXTDISABLE bit in HTMCONTROL register.

See also

- HTM
- HTM.state

Format: **HTM.FifoLevel** *<bytes>*

If the HTM FIFO is almost full (available free space smaller the FIFO level) data packets and auxiliary packets are not sent. Data suppressed packets are sent instead.

Example:

```
HTM.FifoLevel 16.
```

See also

- HTM
- HTM.state

HTM.OFF Switch HTM off

Format: **HTM.OFF**

Disables the HTM functionality.

See also

- HTM
- HTM.state

HTM.ON Switch HTM on

Format: **HTM.ON**

Enables the HTM functionality.

See also

- HTM
- HTM.state

Format: **ETM.PortRoute** [**Analyzer** | **CAnalyzer** | **Onchip**]

Prepares the selected trace hardware for HTM trace capture.

Analyzer	PowerTrace (via TPIU)
CAnalyzer	Compact-Analyzer: CombiProbe or µTrace (MicroTrace)
Onchip	Onchip trace buffer (ETB, ETF or ETR)

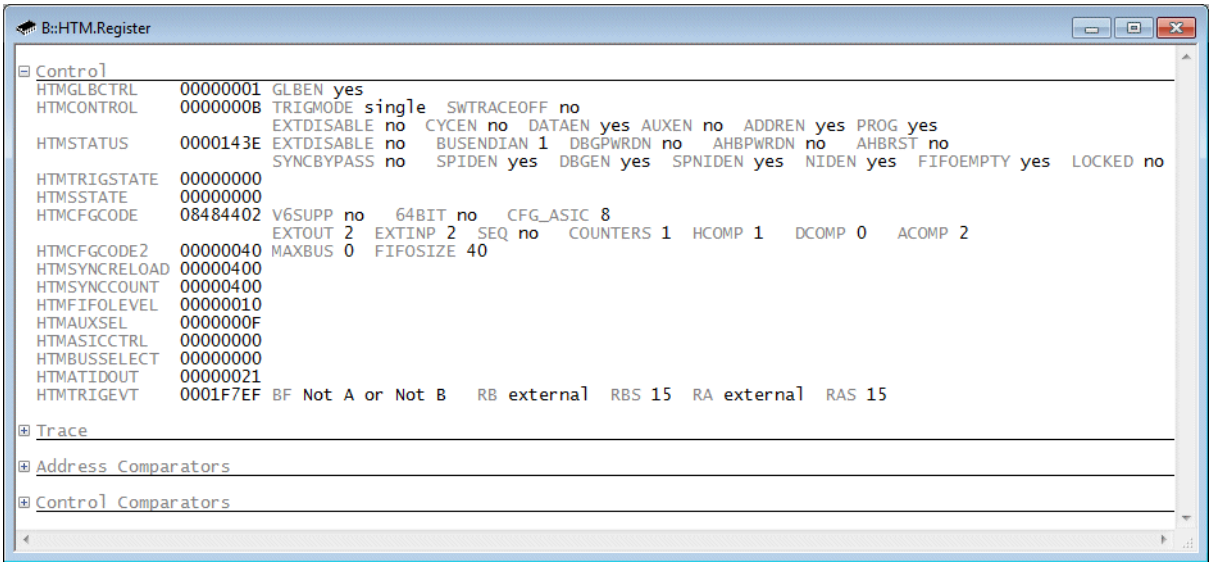
See also

- HTM
- HTM.state

Format: **HTM.Register** [*/<option>*]

<option>: **SpotLight** | **DualPort** | **Track** | **AlternatingBackGround**
CORE *<core_number>*

Displays the HTM control registers. The contents will vary with the HTM version.



<option> For a description of the options, see **PER.view**.

See also

- HTM
- HTM.state

HTM.RESet Reset HTM settings

Format: **HTM.RESet**

Resets the settings in the **HTM.state** window to default and resets the HTM register.

See also

- HTM
- HTM.state

Format: **HTM.Set**

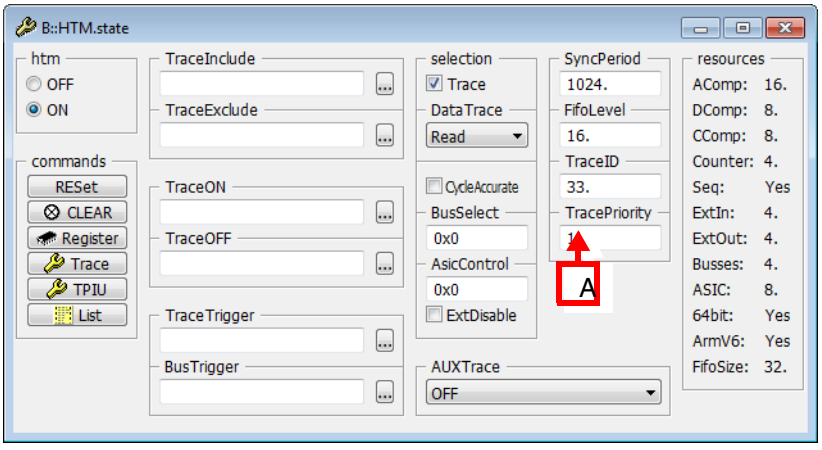
See also

- HTM
- HTM.state

HTM.state Display HTM configuration window

Format: **HTM.state**

Opens the **HTM.state** window, where you can configure the trace source HTM.



A For descriptions of the commands in the **HTM.state** window, please refer to the **HTM.*** commands in this chapter.
Example: For information about the **TracePriority** box, see **HTM.TracePriority**.

See also

- HTM
- HTM.AXIMaster
- HTM.CLOCK
- HTM.FifoLevel
- HTM.Register
- HTM.Trace
- HTM.TraceOFF
- HTM.AsicControl
- HTM.BusSelect
- HTM.CycleAccurate
- HTM.OFF
- HTM.RESet
- HTM.TraceExclude
- HTM.TraceON
- HTM.AuxTrace
- HTM.BusTrigger
- HTM.DataTrace
- HTM.ON
- HTM.Set
- HTM.TraceID
- HTM.TracePriority
- HTM.AXIFifoClock
- HTM.CLEAR
- HTM.ExtDisable
- HTM.PortRoute
- HTM.SyncPeriod
- HTM.TraceInclude
- HTM.TraceTrigger

HTM.SyncPeriod — Set period of sync packet injection

Format: **HTM.SyncPeriod** [*<packets>*]

Sets the number of regular HTM packets which will be output to the trace stream between two synchronization packets.

<packets> — If omitted, then the default number of regular packets between synchronization packets is chosen by the debugger or the chip.

See also

- HTM
- HTM.state

HTM.Trace — Trace packet control

Format: **HTM.Trace** [**ON** | **OFF**]

ON (default)	Trace information is generated and triggers/external signals are activated as programmed.
OFF	No trace packets are generated. Only triggers/external signals are activated as programmed.

See also

- HTM
- HTM.state

Format: **HTM.TraceExclude** *<range>* [*<range>* ...] *<access>*

<access>: **Fetch** | **Access** | **Read** | **Write** | **Any**

This command can be used to exclude data accesses within the specified *<address_range>* from broadcasting. The number of exclude-ranges depends on the number of address comparator pairs provided by the HTM.

Example:

```
HTM.TraceExclude 0x1d00--0x1dff
```

See also

- HTM
- HTM.state

HTM.TraceID

Set trace ID manually

Format: **HTM.TraceID** *<id>*

TRACE32 automatically assigns an appropriate source ID to the HTM. This command allows the user to specify a user-defined trace ID for the HTM.

See also

- HTM
- HTM.state

Format:	**HTM.TraceInclude** *<range>* [*<range>* ...] *<access>*
<access>:	**Fetch** | **Access** | **Read** | **Write** | **Any**

This command can be used to restrict the broadcasting of data accesses to the specified *<address_range>*. The number of include-ranges depends on the number of address comparator pair provided by the HTM.

Example:

```
HTM.TraceInclude Write 0x4A000000--0x4B001000
```

See also

- HTM
- HTM.state

HTM.TraceOFF HTM stops to emit trace information on event

Format:	**HTM.TraceOFF** *<address>* | *<address_range>* *<access>*
<access>:	**Fetch** | **Access** | **Read** | **Write** | **Any**

Advises the HTM to stop emitting trace information when the specified event becomes true.

See also

- HTM
- HTM.state

Format:	**HTM.TraceON** *<address>* | *<address_range>* *<access>*
<access>:	**Fetch** | **Access** | **Read** | **Write** | **Any**

Advises the HTM to start emitting trace information when the specified event becomes true.

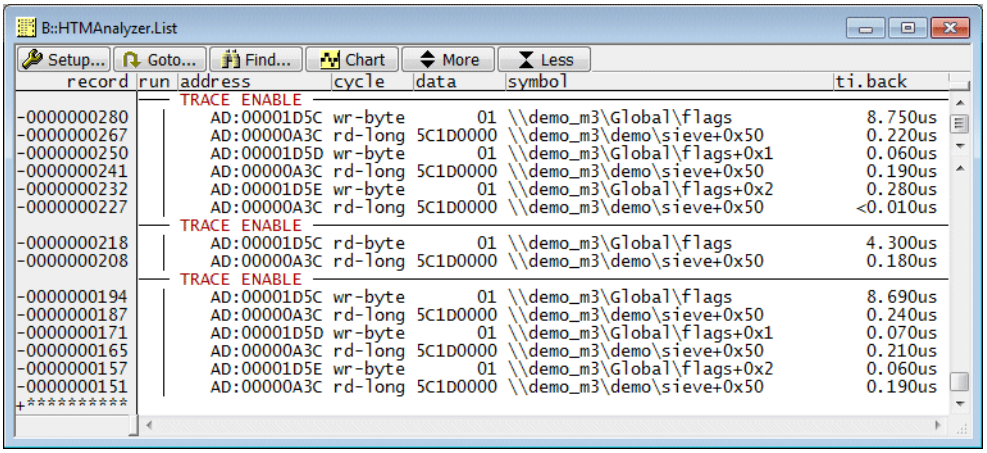
Example:

```
HTM.TraceON flags Read                          ; advise the HTM to start to
                                                ; emit trace information when
                                                ; the contents of the address
                                                ; flags is read

HTM.TraceOFF V.RANGE(flags[3]) Read             ; advise the HTM to stop to
                                                ; emit trace information when
                                                ; the contents of the variable
                                                ; flags[3] is written

...

HTMAnalyzer.List                                ; display the trace contents
```



record	run	address	cycle	data	symbol	ti.back
		TRACE ENABLE				
-0000000280		AD:00001D5C	wr-byte	01	\\demo_m3\Global\flags	8.750us
-0000000267		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	0.220us
-0000000250		AD:00001D5D	wr-byte	01	\\demo_m3\Global\flags+0x1	0.060us
-0000000241		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	0.190us
-0000000232		AD:00001D5E	wr-byte	01	\\demo_m3\Global\flags+0x2	0.280us
-0000000227		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	<0.010us
		TRACE ENABLE				
-0000000218		AD:00001D5C	rd-byte	01	\\demo_m3\Global\flags	4.300us
-0000000208		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	0.180us
		TRACE ENABLE				
-0000000194		AD:00001D5C	wr-byte	01	\\demo_m3\Global\flags	8.690us
-0000000187		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	0.240us
-0000000171		AD:00001D5D	wr-byte	01	\\demo_m3\Global\flags+0x1	0.070us
-0000000165		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	0.210us
-0000000157		AD:00001D5E	wr-byte	01	\\demo_m3\Global\flags+0x2	0.060us
-0000000151		AD:00000A3C	rd-long	5C1D0000	\\demo_m3\demo\sieve+0x50	0.190us
+**********						

See also

- HTM
- HTM.state

Format: **HTM.TracePriority** *<priority>*

TRACE32 automatically assigns an appropriate priority to the HTM. This command allows the user to change the priority for the HTM trace information.

See also

- HTM
- HTM.state

HTM.TraceTrigger Trace trigger definition

Format: **HTM.TraceTrigger** *<address>* | *<range>* *<access>*

<access>: **Fetch** | **Access** | **Read** | **Write** | **Any**

Generates a trigger for the trace, if the specified event occurs.

A trigger to the trace requires that the trigger output of the HTM is connected to the TPIU. The following options are possible:

- The trigger output of the HTM is hard-wired to the TPIU on the chip.
- The trigger output of the HTM can be connected to the TPIU via CTI (Cross-Trigger Interface).
- No connection between the trigger output and the TPIU can be established.

Example:

```
HTM.TraceTrigger 0x1000 Write
```

See also

- HTM
- HTM.state

HTM<trace> Command groups for HTM<trace>

See also

- HTMAnalyzer
- HTMCAnalyzer
- HTMHAnalyzer
- HTMLA
- HTMOnchip
- HTMTrace

▲ 'HTM - Configuration of the Trace Source' in 'General Commands Reference Guide H'

Overview HTM<trace>

Using the **HTM<trace>** command groups, you can configure the trace recording as well as analyze and display the recorded HTM trace data. The command groups consist of the name of the trace source, here **HTM**, plus the TRACE32 trace method you have chosen for recording the HTM trace data.

For more information about the TRACE32 convention of combining *<trace_source>* and *<trace_method>* to a *<trace>* command group that is aimed at a specific trace source, see **"Replacing <trace> with Trace Source and Trace Method - Examples"** (general_ref_t.pdf).

Not any arbitrary combination of *<trace_source>* and *<trace_method>* is possible. For an overview of the available command groups **"Related Trace Command Groups"** (general_ref_t.pdf).

Example:

```
HTMTrace.state            ;optional step: open the window in which the
                          ;trace recording is configured.
HTMTrace.METHOD Analyzer  ;select the trace method Analyzer for
;<configuration>          ;recording trace data.

HTM.state                 ;optional step: open the window in which
                          ;the trace source HTM is configured.
HTM.ON                    ;switch the trace source HTM on.
;<configuration>

;trace data is recorded using the commands Go, WAIT, Break

HTMAnalyzer.List          ;display the HTM trace data recorded with the
                          ;trace method Analyzer as a trace listing.

HTMTrace.List             ;this is the generic replacement for the above
                          ;HTMAnalyzer.List command.
```

[Example]

Format: **HTMAnalyzer.***<sub_cmd>*

The **HTMAnalyzer** command group allows to display and analyze the information emitted by the CoreSight AHB Trace Macrocell (**HTM**).

The HTM trace is transferred off-chip via the Trace Port Interface Unit (see **TPIU** command group) and is recorded into the trace memory (Analyzer) of a TRACE32 trace tool (e.g. PowerTrace II).

<sub_cmd>	For descriptions of the subcommands, please refer to the general *<trace>* command descriptions in **"General Commands Reference Guide T"** (general_ref_t.pdf). **Example**: For a description of **HTMAnalyzer.List** refer to **<trace>.List**

See also

- HTM<trace>
- ▲ 'Trace' in 'General Commands Reference Guide T'

HTMCAnalyzer Analyze HTM info. recorded by TRACE32 CombiProbe

Format: **HTMCAnalyzer.***<sub_cmd>*

The **HTMCAnalyzer** command group allows to display and analyze the information emitted by the CoreSight AHB Trace Macrocell (HTM).

The HTM trace is transferred off-chip via the Trace Port Interface Unit (see **TPIU** command group) and is recorded into the trace memory of the TRACE32 CombiProbe.

<sub_cmd>	For descriptions of the subcommands, please refer to the general *<trace>* command descriptions in **"General Commands Reference Guide T"** (general_ref_t.pdf). **Example**: For a description of **HTMCAnalyzer.List** refer to **<trace>.List**

See also

- HTM<trace>

Format: **HTMHAnalyzer.***<sub_cmd>*

The **HTMHAnalyzer** command group allows to display and analyze the information emitted by the CoreSight AHB Trace Macrocell (HTM).

The HTM trace is transferred off-chip via the USB port and is recorded into the trace memory of the TRACE32 host analyzer.

<sub_cmd>	For descriptions of the subcommands, please refer to the general *<trace>* command descriptions in **"General Commands Reference Guide T"** (general_ref_t.pdf). **Example**: For a description of **HTMHAnalyzer.List** refer to **<trace>.List**

See also

- HTM<trace>

HTMLA HTM logic analyzer

Format: **HTMLA.***<sub_cmd>*

The **HTMLA** command group allows to display and analyze the information emitted by the CoreSight AHB Trace Macrocell (HTM). Trace data is collected from Lauterbach's Logic Analyzer or from a binary file.

<sub_cmd>	For descriptions of the subcommands, please refer to the general *<trace>* command descriptions in **"General Commands Reference Guide T"** (general_ref_t.pdf). **Example**: For a description of **HTMLA.List** refer to **<trace>.List**

See also

- HTM<trace>

Format:	**HTMOnchip.**<*sub_cmd*>

The **HTMOnchip** command group allows to display and analyze the information emitted by the CoreSight AHB Trace Macrocell (**HTM**).

The HTM trace is sent to the ETB onchip trace memory (Onchip).

<*sub_cmd*>	For descriptions of the subcommands, please refer to the general <*trace*> command descriptions in **"General Commands Reference Guide T"** (general_ref_t.pdf). **Example**: For a description of **HTMOnchip.List** refer to **<trace>.List**

See also

- HTM<trace>

HTMTrace Method-independent analysis of HTM trace data

[Example]

Format:	**HTMTrace.**<*sub_cmd*>

The **HTMTrace** command group can be used as a generic replacement for the above **HTM<trace>** command groups.

<*sub_cmd*>	For descriptions of the subcommands, please refer to the general <*trace*> command descriptions in **"General Commands Reference Guide T"** (general_ref_t.pdf). **Example**: For a description of **HTMTrace.List** refer to **<trace>.List**

See also

- HTM<trace>

HVX — HVX registers (Hexagon Vector Extensions)

Hexagon only

The **HVX** command group is used to display and modify the HVX (Hexagon Vector Extensions) registers for Hexagon.

See also

- HVX.Init
- HVX.OFF
- HVX.ON
- HVX.Set
- HVX.view
- HVX()
- ▲ 'HVX Function' in 'General Function Reference'

HVX.Init — Initialize HVX registers

Hexagon only

Format: **HVX.Init**
HVX.RESet (deprecated)

Sets all registers of the active HVX extension to zero.

See also

- HVX
- HVX.view

HVX.OFF — Inhibit HVX accesses by the debugger

Hexagon only

Format: **HVX.OFF**

Inhibits accesses to the HVX by the debugger. Usually required until the HVX is on.

See also

- HVX
- HVX.view

HVX.ON — Permit HVX accesses by the debugger

Hexagon only

Format: **HVX.ON**

Permits accesses to the HVX by the debugger (default).

See also

- HVX
- HVX.view

HVX.Set — Modify HVX registers

Hexagon only

Format: **HVX.Set** *<register> <value> …* [*/<option>*]

Modifies the HVX registers.

<option> For a description of the options, see **Register.view**.

See also

- HVX
- HVX.view

HVX.view — Open HVX register window

Hexagon only

Format: **HVX.view** [*/<option>*]

Opens an HVX register window.

<option> For a description of the options, see **Register.view**.

See also

- HVX
- HVX.Init
- HVX.OFF
- HVX.ON
- HVX.Set