

General Commands Reference Guide A

General Commands Reference Guide A

TRACE32 Online Help

TRACE32 Directory

TRACE32 Index

TRACE32 Documents	
General Commands	
General Commands Reference Guide A	1
History	11
AET	12
AET.CLEAR	Clear AET settings 12
AET.DataTrace	Configure AET data-trace 12
AET.GatedClock	Use trace port clock when no data is sent 13
AET.OFF	Switch AET off 13
AET.ON	Switch AET on 14
AET.PATTERN	Enable AET pattern generator 14
AET.PortClock	Select AET port mode 14
AET.PortMode	Select AET port mode 14
AET.PortSize	Select AET trace port width 15
AET.ReadWriteBreak	Control read/write breakpoints 15
AET.Register	Display the AET unit registers 16
AET.RESet	Reset AET settings 16
AET.SHADOW	Set AET shadow memory address 16
AET.STALL	Stall processor to prevent FIFO overflow 17
AET.state	Display AET settings 17
AET.SyncPeriod	Set synchronization frequency 17
AET.TagDataTrace	Tag AET data trace 18
AET.TimingTrace	Select AET trace timestamp information 18
AET.Trace	Control generation of trace information 19
AET.TraceID	Change the default ID for an AET trace source 19
AET.TracePriority	Define priority of AET 19
Analyzer	20
Analyzer	Trace method Analyzer, recording, and analysis commands 20
Analyzer-specific Trace Commands	22
Analyzer.Mode	Set the trace operation mode 22
Analyzer.PIPECompression	Enable compression in PIPE mode 24
Analyzer.Program	Program trigger unit 24
Analyzer.RecordAutoFill	Precision of run-time measurements 25
Analyzer.REMAP	Remap trace port channels 25

Analyzer.REMAP.CLOCK	Select input clock	26
Analyzer.REMAP.RESet	Reset pinout configuration	26
Analyzer.REMAP.state	Display remap configuration window	26
Analyzer.ReProgram	Program trigger unit	27
Analyzer.SAMPLE	Set AutoFocus sample time offset	27
Analyzer.TOut	Trigger output line	28
Analyzer.TraceCLOCK	Improve timestamps on PowerTrace	28
Generic Analyzer Trace Commands		29
Analyzer.ACCESS	Define access path to program code for trace decoding	29
Analyzer.Arm	Arm the trace	29
Analyzer.AutoArm	Arm automatically	29
Analyzer.AutoFocus	Calibrate AUTOFOCUS preprocessor	29
Analyzer.AutoInit	Automatic initialization	29
Analyzer.AutoStart	Automatic start	29
Analyzer.BookMark	Set a bookmark in trace listing	30
Analyzer.BookMarkToggle	Toggles a single trace bookmark	30
Analyzer.Chart	Display trace contents graphically	30
Analyzer.Chart.Address	Time between program events as a chart	30
Analyzer.Chart.AddressGROUP	Address group time chart	30
Analyzer.Chart.ChildTREE	Display callee context of a function as chart	30
Analyzer.Chart.DatasYmbol	Analyze pointer contents graphically	30
Analyzer.Chart.DistriB	Distribution display graphically	31
Analyzer.Chart.Func	Function activity chart	31
Analyzer.Chart.GROUP	Group activity chart	31
Analyzer.Chart.INTERRUPT	Display interrupt chart	31
Analyzer.Chart.INTERRUPTTREE	Display interrupt nesting	31
Analyzer.Chart.Line	Graphical HLL lines analysis	31
Analyzer.Chart.MODULE	Code execution broken down by module as chart	31
Analyzer.Chart.Nesting	Show function nesting at cursor position	32
Analyzer.Chart.PAddress	Which instructions accessed data address	32
Analyzer.Chart.PROGRAM	Code execution broken down by program	32
Analyzer.Chart.PsYmbol	Shows which functions accessed data address	32
Analyzer.Chart.RUNNABLE	Runnable activity chart	32
Analyzer.Chart.sYmbol	Symbol analysis	32
Analyzer.Chart.TASK	Task activity chart	32
Analyzer.Chart.TASKFunc	Task related function run-time analysis (legacy)	33
Analyzer.Chart.TASKINFO	Context ID special messages	33
Analyzer.Chart.TASKINTR	Display ISR2 time chart (ORTI)	33
Analyzer.Chart.TASKKernel	Task run-time chart with kernel markers (flat)	33
Analyzer.Chart.TASKORINTERRUPT	Task and interrupt activity chart	33
Analyzer.Chart.TASKORINTRState	Task and ISR2 state analysis	33
Analyzer.Chart.TASKSRV	Service routine run-time analysis	33
Analyzer.Chart.TASKState	Task state analysis	34

Analyzer.Chart.TASKVSINTERRUPT	Time chart of interrupted tasks	34
Analyzer.Chart.TASKVSINTR	Time chart of task-related interrupts	34
Analyzer.Chart.TREE	Display function chart as tree view	34
Analyzer.Chart.Var	Variable chart	34
Analyzer.Chart.VarState	Variable activity chart	34
Analyzer.CLOCK	Clock to calculate time out of cycle count information	34
Analyzer.ComPare	Compare trace contents	35
Analyzer.ComPareCODE	Compare trace with memory	35
Analyzer.CustomTrace	Custom trace	35
Analyzer.CustomTraceLoad	Load a DLL for trace analysis/Unload all DLLs	35
Analyzer.DISable	Disable the trace	35
Analyzer.DRAW	Plot trace data against time	35
Analyzer.DRAW.channel	Plot no-data values against time	35
Analyzer.DRAW.Data	Plot data values against time	36
Analyzer.DRAW.Var	Plot variable values against time	36
Analyzer.EXPORT	Export trace data for processing in other applications	36
Analyzer.EXPORT.ARTI	Export trace data as ARTI for CP	36
Analyzer.EXPORT.ARTIAP	Export trace data as ARTI for AP	36
Analyzer.EXPORT.Ascii	Export trace data as ASCII	36
Analyzer.EXPORT.Bin	Export trace data as binary file	36
Analyzer.EXPORT.BRANCHFLOW	Export branch events from trace data	37
Analyzer.EXPORT.CSVFunc	Export the function nesting to a CSV file	37
Analyzer.EXPORT.cycles	Export trace data	37
Analyzer.EXPORT.Func	Export function nesting	37
Analyzer.EXPORT.MDF	Export trace data as MDF	37
Analyzer.EXPORT.MTV	Export in MCDS Trace Viewer format	37
Analyzer.EXPORT.TASK	Export task switches	37
Analyzer.EXPORT.TASKEVENTS	Export task event to CSV	38
Analyzer.EXPORT.TracePort	Export trace packets as recorded at trace port	38
Analyzer.EXPORT.VCD	Export trace data in VCD format	38
Analyzer.EXPORT.VERILOG	Export trace data in VERILOG format	38
Analyzer.EXPORT.VHDL	Export trace data in VHDL format	38
Analyzer.ExtractCODE	Extract code from trace	38
Analyzer.FILE	Load a file into the file trace buffer	38
Analyzer.Find	Find specified entry in trace	39
Analyzer.FindAll	Find all specified entries in trace	39
Analyzer.FindChange	Search for changes in trace flow	39
Analyzer.FindProgram	Advanced trace search	39
Analyzer.FindReProgram	Activate advanced existing trace search program	39
Analyzer.FindViewProgram	State of advanced trace search programming	39
Analyzer.FLOWPROCESS	Process flowtrace	39
Analyzer.FLOWSTART	Restart flowtrace processing	40
Analyzer.Get	Display input level	40

Analyzer.GOTO	Move cursor to specified trace record	40
Analyzer.Init	Initialize trace	40
Analyzer.JOINFILE	Concatenate several trace recordings	40
Analyzer.List	List trace contents	40
Analyzer.ListNesting	Analyze function nesting	40
Analyzer.ListVar	List variable recorded to trace	40
Analyzer.LOAD	Load trace file for offline processing	41
Analyzer.MERGEFILE	Combine two trace files into one	41
Analyzer.OFF	Switch off	41
Analyzer.PipeWRITE	Connect to a named pipe to stream trace data	41
Analyzer.PlatformCLOCK	Set clock for platform traces	41
Analyzer.PortFilter	Specify utilization of trace memory	41
Analyzer.PortSize	Set external port size	41
Analyzer.PortType	Specify trace interface	42
Analyzer.PROfile	Rolling live plots of trace data	42
Analyzer.PROfile.CTU	Display complex trigger unit counter profile	42
Analyzer.PROfileChart	Profile charts	42
Analyzer.PROfileChart.Address	Address profile chart	42
Analyzer.PROfileChart.AddressGROUP	Address group time chart	42
Analyzer.PROfileChart.AddressRate	Address rate profile chart	42
Analyzer.PROfileChart.COUNTER	Display a profile chart	43
Analyzer.PROfileChart.DatasYmbol	Analyze pointer contents graphically	43
Analyzer.PROfileChart.DIStance	Time interval for a single event	43
Analyzer.PROfileChart.DistriB	Distribution display in time slices	43
Analyzer.PROfileChart.DURation	Time between two events	43
Analyzer.PROfileChart.GROUP	Group profile chart	43
Analyzer.PROfileChart.INTERRUPT	Display interrupt profile chart	43
Analyzer.PROfileChart.Line	HLL-line profile chart	44
Analyzer.PROfileChart.MODULE	Module profile chart	44
Analyzer.PROfileChart.PAddress	Which instructions accessed data address	44
Analyzer.PROfileChart.PROGRAM	Program profile chart	44
Analyzer.PROfileChart.PsYmbol	Which functions accessed data address	44
Analyzer.PROfileChart.Rate	Event frequency	44
Analyzer.PROfileChart.RUNNABLE	Runnable profile chart	44
Analyzer.PROfileChart.sYmbol	Dynamic program behavior graphically (flat)	45
Analyzer.PROfileChart.TASK	Dynamic task behavior graphically (flat)	45
Analyzer.PROfileChart.TASKINFO	Context ID special messages	45
Analyzer.PROfileChart.TASKINTR	ISR2 profile chart (ORTI)	45
Analyzer.PROfileChart.TASKKernel	Task profile chart with kernel markers	45
Analyzer.PROfileChart.TASKORINTERRUPT	Task and interrupt profile chart	45
Analyzer.PROfileChart.TASKSRV	Profile chart of OS service routines	45
Analyzer.PROfileChart.TASKVSINTERRUPT	Interrupted tasks	46
Analyzer.PROfileChart.TASKVSINTR	Profile chart for task-related interrupts	46

Analyzer.PROfileChart.Var	Variable profile chart	46
Analyzer.PROfileSTATistic	Statistical analysis in a table versus time	46
Analyzer.PROfileSTATistic.Address	Statistical analysis for addresses	46
Analyzer.PROfileSTATistic.AddressGROUP	Stat. for address groups	46
Analyzer.PROfileSTATistic.COUNTER	Statistical analysis for counter	46
Analyzer.PROfileSTATistic.DatasYmbol	Statistic analysis for pointer content	47
Analyzer.PROfileSTATistic.DistriB	Distribution statistical analysis	47
Analyzer.PROfileSTATistic.GROUP	Statistical analysis for groups	47
Analyzer.PROfileSTATistic.INTERRUPT	Statistical analysis for interrupts	47
Analyzer.PROfileSTATistic.Line	Statistical analysis for HLL lines	47
Analyzer.PROfileSTATistic.MODULE	Statistical analysis for modules	47
Analyzer.PROfileSTATistic.PAddress	Which instr. accessed data address	47
Analyzer.PROfileSTATistic.PROGRAM	Statistical analysis for programs	48
Analyzer.PROfileSTATistic.PsYmbol	Which functions accessed data address	48
Analyzer.PROfileSTATistic.RUNNABLE	Statistical analysis for runnables	48
Analyzer.PROfileSTATistic.sYmbol	Statistical analysis for symbols	48
Analyzer.PROfileSTATistic.TASK	Statistical analysis for tasks	48
Analyzer.PROfileSTATistic.TASKINFO	Context ID special messages	48
Analyzer.PROfileSTATistic.TASKINTR	Statistical analysis for ISR2 (ORTI)	48
Analyzer.PROfileSTATistic.TASKKernel	Stat. analysis with kernel markers	49
Analyzer.PROfileSTATistic.TASKORINTERRUPT	Interrupts and tasks	49
Analyzer.PROfileSTATistic.TASKSRV	Analysis of OS service routines	49
Analyzer.PROfileSTATistic.TASKVSINTERRUPT	Interrupted tasks	49
Analyzer.PROTOcol	Protocol analysis	49
Analyzer.PROTOcol.Chart	Graphic display for user-defined protocol	49
Analyzer.PROTOcol.Draw	Graphic display for user-defined protocol	49
Analyzer.PROTOcol.EXPORT	Export trace buffer for user-defined protocol	50
Analyzer.PROTOcol.Find	Find in trace buffer for user-defined protocol	50
Analyzer.PROTOcol.list	Display trace buffer for user-defined protocol	50
Analyzer.PROTOcol.PROfileChart	Profile chart for user-defined protocol	50
Analyzer.PROTOcol.PROfileSTATistic	Profile chart for user-defined protocol	50
Analyzer.PROTOcol.STATistic	Display statistics for user-defined protocol	50
Analyzer.REF	Set reference point for time measurement	50
Analyzer.RESet	Reset command	51
Analyzer.SAVE	Save trace for postprocessing in TRACE32	51
Analyzer.SelfArm	Automatic restart of trace recording	51
Analyzer.ShowFocus	Display data eye for AUTOFOCUS preprocessor	51
Analyzer.ShowFocusClockEye	Display clock eye	51
Analyzer.ShowFocusEye	Display data eye	51
Analyzer.SIZE	Define buffer size	51
Analyzer.SnapShot	Restart trace capturing once	52
Analyzer.SPY	Adaptive stream and analysis	52
Analyzer.state	Display trace configuration window	52

Analyzer.STATistic	Statistic analysis	52
Analyzer.STATistic.Address	Time between up to 8 program events	52
Analyzer.STATistic.AddressDIsTance	Time interval for single program event	52
Analyzer.STATistic.AddressDURation	Time between two program events	52
Analyzer.STATistic.AddressGRoUP	Address group run-time analysis	53
Analyzer.STATistic.ChildTREE	Show callee context of a function	53
Analyzer.STATistic.COLOr	Assign colors to function for colored graphics	53
Analyzer.STATistic.CYcle	Analyze cycle types	53
Analyzer.STATistic.DatasYmbol	Analyze pointer contents numerically	53
Analyzer.STATistic.DIsTance	Time interval for a single event	53
Analyzer.STATistic.DistriB	Distribution analysis	53
Analyzer.STATistic.DURation	Time between two events	54
Analyzer.STATistic.FIRST	Start point for statistic analysis	54
Analyzer.STATistic.Func	Nesting function runtime analysis	54
Analyzer.STATistic.FuncDURation	Statistic analysis of single function	54
Analyzer.STATistic.FuncDURationInternal	Statistic analysis of single func.	54
Analyzer.STATistic.GRoUP	Group run-time analysis	54
Analyzer.STATistic.Ignore	Ignore false records in statistic	54
Analyzer.STATistic.INTERRUPT	Interrupt statistic	55
Analyzer.STATistic.InterruptIsFunction	Statistics interrupt processing	55
Analyzer.STATistic.InterruptIsKernel	Statistics interrupt processing	55
Analyzer.STATistic.InterruptIsKernelFunction	Statistics interrupt processing	55
Analyzer.STATistic.InterruptIsTaskswitch	Statistics interrupt processing	55
Analyzer.STATistic.INTERRUPTTREE	Display interrupt nesting	55
Analyzer.STATistic.LAST	End point for statistic analysis	55
Analyzer.STATistic.Line	High-level source code line analysis	56
Analyzer.STATistic.LINKAge	Per caller statistic of function	56
Analyzer.STATistic.Measure	Analyze the performance of a single signal	56
Analyzer.STATistic.MODULE	Code execution broken down by module	56
Analyzer.STATistic.PAddress	Which instructions accessed data address	56
Analyzer.STATistic.ParentTREE	Show the call context of a function	56
Analyzer.STATistic.PROCESS	Re-process statistics	56
Analyzer.STATistic.PROGRAM	Code execution broken down by program	57
Analyzer.STATistic.PsYmbol	Shows which functions accessed data address	57
Analyzer.STATistic.RUNNABLE	Runnable runtime analysis	57
Analyzer.STATistic.RUNNABLEDURation	Runnable duration analysis	57
Analyzer.STATistic.Sort	Specify sorting criteria for statistic commands	57
Analyzer.STATistic.sYmbol	Flat run-time analysis	57
Analyzer.STATistic.TASK	Task activity statistic	57
Analyzer.STATistic.TASKFunc	Task related function run-time analysis	58
Analyzer.STATistic.TASKINFO	Context ID special messages	58
Analyzer.STATistic.TASKINTR	ISR2 statistic (ORTI)	58
Analyzer.STATistic.TASKKernel	Task analysis with kernel markers (flat)	58

Analyzer.STATistic.TASKLOCK	Analyze lock accesses from tasks	58
Analyzer.STATistic.TASKORINTERRUPT	Statistic of interrupts and tasks	58
Analyzer.STATistic.TASKORINTRState	Task and ISR2 statistic analysis	58
Analyzer.STATistic.TASKSRV	Analysis of time in OS service routines	59
Analyzer.STATistic.TASKState	Performance analysis	59
Analyzer.STATistic.TASKStateDURation	Task state runtime analysis	59
Analyzer.STATistic.TASKTREE	Tree display of task specific functions	59
Analyzer.STATistic.TASKVSINTERRUPT	Statistic of interrupts, task-related	59
Analyzer.STATistic.TASKVSINTR	ISR2 statistic (ORTI), task related	59
Analyzer.STATistic.TREE	Tree display of nesting function run-time analysis	59
Analyzer.STATistic.Use	Use records	60
Analyzer.STATistic.Var	Statistic of variable accesses	60
Analyzer.STREAMCompression	Select compression mode for streaming	60
Analyzer.STREAMFILE	Specify temporary streaming file path	60
Analyzer.STREAMFileLimit	Set size limit for streaming file	60
Analyzer.STREAMLOAD	Load streaming file from disk	60
Analyzer.STREAMSAVE	Save streaming file to disk	60
Analyzer.TDelay	Trigger delay	61
Analyzer.TERMination	Use trace line termination of preprocessor	61
Analyzer.TestFocus	Test trace port recording	61
Analyzer.TestFocusClockEye	Scan clock eye	61
Analyzer.TestFocusEye	Check signal integrity	61
Analyzer.TestUtilization	Tests trace port utilization	61
Analyzer.THreshold	Optimize threshold for trace lines	61
Analyzer.Timing	Waveform of trace buffer	62
Analyzer.TraceCONNECT	Select on-chip peripheral sink	62
Analyzer.TRACK	Set tracking record	62
Analyzer.TSElect	Select trigger source	62
Analyzer.View	Display single record	62
Analyzer.ZERO	Align timestamps of trace and timing analyzers	62
APU		63
APU	Auxiliary processing unit	63
APU.Break	APU breakpoints	63
APU.Break.Delete	Delete APU breakpoint	63
APU.Break.direct	Stop the APU	64
APU.Break.Init	Initialize APU breakpoint system	64
APU.Break.List	List APU breakpoints	64
APU.Break.RESet	Reset APU breakpoint system	65
APU.Break.Set	Set permanent APU breakpoint	65
APU.command	Execute APU specific command	66
APU.Data	APU data command group	66
APU.Data.dump	Data memory display	66
APU.Data.List	Symbolic display	67

APU.Data.LOAD	Load file	67
APU.Data.Set	Data memory modification	68
APU.Go	Start the APU	68
APU.GREP	Search for string	69
APU.List	View program	69
APU.ListHll	View program source	70
APU.LOAD	Load APU library	70
APU.Register	Show APU register window	70
APU.Register.Set	Register modification	71
APU.Register.view	Register display	71
APU.RESet	Reset APU core	72
APU.Step	Single-stepping	72
APU.StepHll	HLL single-stepping	72
APU.View	Display APU peripherals	73
ART		74
ART	Trace method for Advanced Register Trace	74
ART-specific Trace Commands		76
ART.Mode	Set the trace operation mode	76
ART Trace Commands		77
ART.Arm	Arm the trace	77
ART.AutoArm	Arm automatically	77
ART.AutoInit	Automatic initialization	77
ART.BookMark	Set a bookmark in trace listing	77
ART.Chart	Display trace contents graphically	77
ART.ComPare	Compare trace contents	77
ART.DISable	Disable the trace	78
ART.DRAW	Plot trace data against time	78
ART.EXPORT	Export trace data for processing in other applications	78
ART.FILE	Load a file into the file trace buffer	78
ART.Find	Find specified entry in trace	78
ART.FindAll	Find all specified entries in trace	78
ART.FindChange	Search for changes in trace flow	78
ART.GOTO	Move cursor to specified trace record	78
ART.Init	Initialize trace	79
ART.List	List trace contents	79
ART.ListNesting	Analyze function nesting	79
ART.LOAD	Load trace file for offline processing	79
ART.OFF	Switch off	79
ART.PROfileChart	Profile charts	79
ART.PROTOcol	Protocol analysis	79
ART.PROTOcol.Chart	Graphic display for user-defined protocol	80
ART.PROTOcol.Draw	Graphic display for user-defined protocol	80

ART.PROTOcol.EXPORT	Export trace buffer for user-defined protocol	80
ART.PROTOcol.Find	Find in trace buffer for user-defined protocol	80
ART.PROTOcol.list	Display trace buffer for user-defined protocol	80
ART.PROTOcol.PROfileChart	Profile chart for user-defined protocol	80
ART.PROTOcol.PROfileSTATistic	Profile chart for user-defined protocol	80
ART.PROTOcol.STATistic	Display statistics for user-defined protocol	81
ART.REF	Set reference point for time measurement	81
ART.RESet	Reset command	81
ART.SAVE	Save trace for postprocessing in TRACE32	81
ART.SelfArm	Automatic restart of trace recording	81
ART.SIZE	Define buffer size	81
ART.SnapShot	Restart trace capturing once	81
ART.state	Display trace configuration window	81
ART.STATistic	Statistic analysis	82
ART.Timing	Waveform of trace buffer	82
ART.TRACK	Set tracking record	82
ART.View	Display single record	82
ART.ZERO	Align timestamps of trace and timing analyzers	82
AutoSTOre		83
AutoSTOre	Save and restore settings (history, GUI, etc.) automatically	83
AVX		85
AVX	AVX registers (Advanced Vector Extension)	85
AVX.Init	Initialize AVX registers	85
AVX.Set	Modify AVX registers	85
AVX.view	Display AVX registers	86
AVX512		87
AVX512	AVX512 registers (Advanced Vector Extension)	87
AVX512.Init	Initialize AVX512 registers	87
AVX512.Set	Modify AVX512 registers	87
AVX512.view	Display AVX512 registers	88

History

14-10-22 Updated description of [Analyzer.TraceCLOCK](#).

AET.CLEAR

Clear AET settings

Format:

AET.CLEAR

Switches on the AET module and clears all AET settings.

AET.DataTrace

Configure AET data-trace

Format:

AET.DataTrace <def>

<def>:

ON
Read
Write
Address
ReadAddress
WriteAddress
Data
ReadData
WriteData
Limited
ReadLimited
WriteLimited
Only
OFF

Configures which elements are included in the data trace.

AET.DataTrace	Trace of Program Flow	Trace Data Values of Read Accesses	Trace Data Values of Write Accesses	Trace Addresses of Read Accesses	Trace Addresses of Write Accesses
ON	☒	☒	☒	☒	☒
Read	☒	☒		☒	
Write	☒		☒		☒
Address	☒			☒	☒

AET.DataTrace	Trace of Program Flow	Trace Data Values of Read Accesses	Trace Data Values of Write Accesses	Trace Addresses of Read Accesses	Trace Addresses of Write Accesses
ReadAddress	■			■	
WriteAddress	■				■
Data	■	■	■		
ReadData	■	■			
WriteData	■		■		
Limited					
ReadLimited					
WriteLimited					
Only		■	■	■	■
OFF	■				

AET.GatedClock

Use trace port clock when no data is sent

Format:

AET.GatedClock [ON | OFF]

Default: OFF

Turns off the trace port clock when no data is sent.

AET.OFF

Switch AET off

Format:

AET.OFF

Disables AET functionality.

Format:	AET.ON
---------	---------------

Enables AET functionality.

AET.PATTERN

Enable AET pattern generator

Format:	AET.PATTERN [ON OFF]
---------	-------------------------------

Allows to manually enable the AET pattern generator for measurements

AET.PortClock

Select AET port mode

Refer to [AET.PortMode](#)

AET.PortMode

Select AET port mode

Format:	AET.PortMode <i><mode></i>
<i><mode></i> :	FCK (default) 1/1 1/2 1/3 1/4 1/5 1/6 1/7 1/8 1/9 1/10 1/11 1/12 1/13 1/14 1/15 1/16

Set the AET trace port clock divider

Format:	AET.PortSize [5 10 10A 10E 10X 10XE 10K 10KE]
<mode>:	5 10 10A 10E 10X 10XE 10K 10KE

Set the AET trace port size and mode.

5, 10	Number of trace data signals.
10A, 10E	Variants of “10”. The selected size is the same, but additionally the Debug Resource Manager (DRM) gets configured which maps trace signals to output pins.
10X, 10XE	Variants of “10X” and “10E”. The selected size is the same, but additionally the Debug Resource Manager (DRM) gets configured which maps trace signals to output pins:
10K, 10KE	Variants of “10” and “10E”. The selected size is the same, but additionally the Debug Resource Manager (DRM) gets configured which maps trace signals to output pins.

AET.ReadWriteBreak

Control read/write breakpoints

Format:	AET.ReadWriteBreak [ON OFF]
---------	-------------------------------

ON (default)	Setting read/write breakpoints uses AET triggers.
OFF	Setting read/write breakpoints does not use AET triggers.

Format:	AET.Register [<i><file></i> / <i><option></i>]
<i><option></i> :	SpotLight DualPort Track CORE <i><core_number></i>

Displays the AET unit registers.

<i><option></i>	For a description of the options, see PER.view .
-----------------------	--

AET.RESet

Reset AET settings

Format:	AET.RESet
---------	------------------

Reset all AET settings to default.

AET.SHADOW

Set AET shadow memory address

Format:	AET.SHADOW [<i><address></i>]
---------	--

Allows to define a memory area where contents written to the AET units are cached. This is helpful if write-only registers need to be debugged.

Format: AET.STALL [ON | OFF]

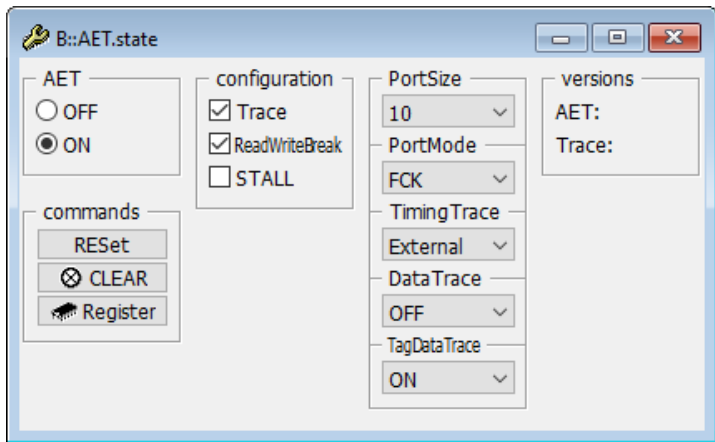
- ON
- Allows the AET unit to stall the processor to prevent an output FIFO overflow. If enabled, the trace will be no longer real time.
- OFF (default)
- Do not allow the AET unit to stall the processor to prevent an output FIFO overflow.

AET.state

Display AET settings

Format: AET.state

Shows the AET configuration window.



AET.SyncPeriod

Set synchronization frequency

Format: AET.SyncPeriod [<packets>]

Synchronisation and timestamp packets are generated periodically. The command **AET.SyncPeriod** allows to change the number of packets between two timestamps.

Format:	AET.TagDataTrace <def>
<def>:	OFF Single Range ON

Default: ON

Specifies which data trace packets are tagged with the PC for correlation when a **/TraceData** breakpoint is set.

OFF	Data trace packets are not tagged
Single	Only data trace packets with addresses aligned to 8bit or 32bit within the /TraceData breakpoint are tagged with the PC for correlation.
Range	Only data trace packets with addresses covered by the /TraceData breakpoint are tagged with the PC for correlation.
On	All data trace packets are tagged with the PC for correlation

AET.TimingTrace

Select AET trace timestamp information

Format:	AET.TimingTrace <mode>
<mode>:	OFF External Detail

There are three methods to timestamp trace information generated by an AET trace port:

OFF	No timestamps are generated.
External (default)	The trace information is timestamped with the global TRACE32 timestamp when it is stored into the trace memory of the TRACE32 trace tool.
Detail	Timestamp Packets are exported by the AET unit.

Format:

AET.Trace [ON | OFF]

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

AET.TraceID

Change the default ID for an AET trace source

Format:

AET.TraceID <id>

By default TRACE32 automatically assigns a trace source ID to all cores with a CoreSight trace source. The command **AET.TraceID** allows to assign an ID to a trace source overriding the defaults.

AET.TracePriority

Define priority of AET

Format:

AET.TracePriority <priority>

The CoreSight Trace Funnel combines 2 to 8 ATB input ports to a single ATB output. An arbiter determines the priority of the ATB input port. Port 0 has the highest priority (0) and port 7 the lowest priority (7) by default.

The command **AET.TracePriority** allows to change the default priority of an ATB input port.

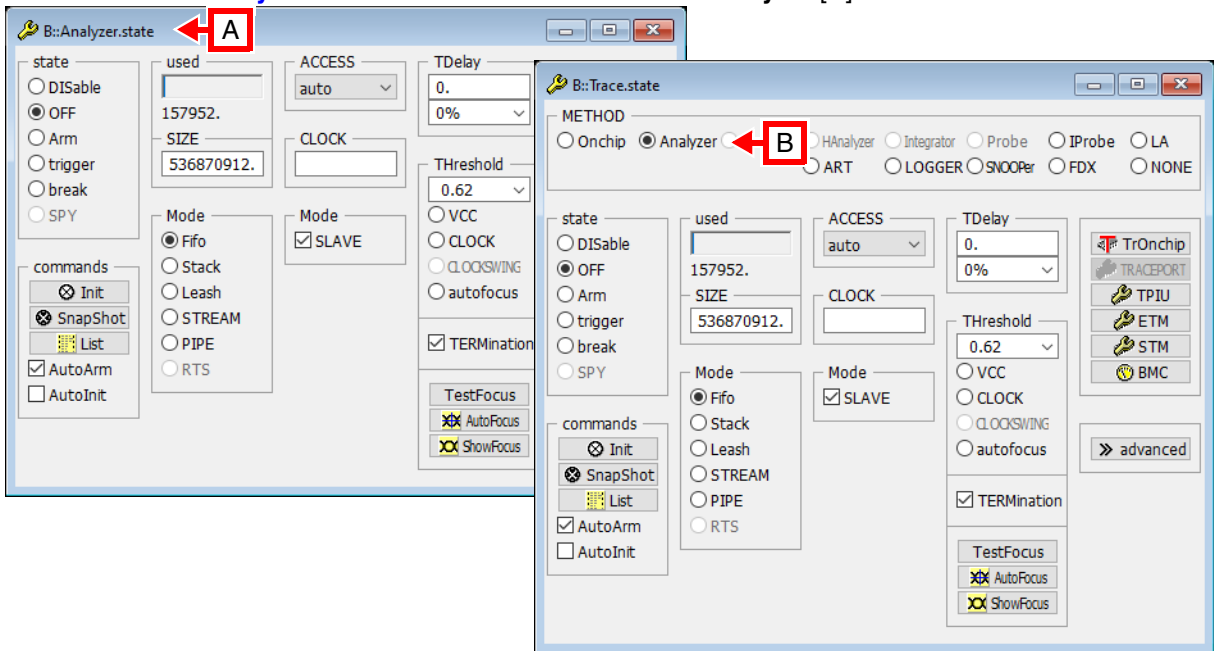
Analyzer is the command group that controls the trace of the following TRACE32 tools:

- **TRACE32 PowerTrace Ethernet, PowerTrace I, PowerTrace II, PowerTrace III, PowerTrace PX, PowerTrace Serial**
- **TRACE32 RiscTrace**
- **TRACE32 Instruction Set Simulator**
- **TRACE32 Font-End to virtual targets supporting trace**

The purpose of the trace method **Analyzer** is to provide a real-time trace to sample the program flow and, depending on the on-chip trace module, also data accesses.

For selecting and configuring the trace method Analyzer, use the TRACE32 command line or a PRACTICE script (*.cmm) or the **Analyzer.state** window [A].

Alternatively, use the **Trace.state** window: click the option **Analyzer** or execute the command **Trace.METHOD Analyzer** in order to select the trace method **Analyzer** [B].



The chapter “[Analyzer-specific Trace Commands](#)”, page 22 describes the Analyzer-specific configuration commands. While the chapter “[Generic Analyzer Trace Commands](#)”, page 29 lists the Analyzer trace analysis and display commands, which are shared with other TRACE32 trace methods.

See also

■ [Trace.METHOD](#)

▲ ['Release Information' in 'Legacy Release History'](#)

Analyzer.Mode

Set the trace operation mode

Format:	Analyzer.Mode [<i><mode></i>]
<i><mode></i> :	Fifo Stack Leash STREAM PIPE RTS SLAVE [ON OFF] PrePost [ON OFF] PostTrace PrePost [ON OFF] Prestore [ON OFF] (deprecated) Use ETM.DataTracePrestore instead. Poststore BusTrace ClockTrace FLowTrace RawTrace MasterBrk [ON OFF] SlaveBrk [ON OFF]

Selects the trace operation mode. The most common operation modes are:

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.
Leash	Stops the program execution when trace is nearly full.

STREAM

The trace data is immediately conveyed to a file on the host after it was placed into the trace. This procedure extends the size of the trace memory to up to several T Frames.

STREAM mode can only be used together with the Analyzer mode for the following TRACE32 trace tools:

- PowerTrace Serial
- PowerTrace II / PowerTrace III

TRACE32 POWERTRACE/ ETHERNET supports STREAM mode for some trace protocols. If it is not supported, the command `Trace.Mode STREAM` is blocked.

STREAM mode can only be used if the average data rate at the trace port does not exceed the maximum transmission rate of the host interface in use. Peak loads at the trace port are intercepted by the trace memory, which can be considered to be operating as a large FIFO.

The streaming file is placed into the TRACE32 temp directory (`OS.PresentTemporaryDirectory()`) by default and is named `<trace32_instance_id>streama.t32 (OS.ID())`. If you explicitly want to specify a location for the streaming file use the command `<trace>.STREAMFILE <file>`.

Note that the contents of the streaming file are in a proprietary format and not intended for use in external applications. See `Trace.EXPORT` for details on how to export trace data for external applications.

Please be aware that the streaming file is deleted as soon as you de-select the STREAM mode or when you quit TRACE32.

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`.

RTS

The RTS radio button is only an indicator that shows if Real-time Profiling is enabled. For enabling RTS use the command `RTS.ON`.

Further operation modes:

BusTrace	Bus trace mode.
ClockTrace	Clock trace mode.
FlowTrace	Flow trace mode.
SLAVE [ON OFF]	ON (default): Ties the trace to the execution of the program, i.e. trace and the filter/trigger work only during user program execution (very rarely used command). OFF : Separates the trace from the program execution, i.e. trace is recording even when the program execution is stopped (rarely used).

See also

- <trace>.Mode
- COVerge.METHOD
- ▲ 'Release Information' in 'Legacy Release History'

Analyzer.PIPECompression

Enable compression in PIPE mode

Format:	Analyzer.PIPECompression [ON OFF]
---------	-------------------------------------

Default: OFF.

Enables compression in PIPE mode ([Analyzer.Mode PIPE](#)).

Analyzer.Program

Program trigger unit

Format:	Analyzer.Program [<file>]
---------	---------------------------

Opens an editor window, which is used for programming the trigger unit for Nexus MPC5xxx. The program entry in the window is guided by softkeys and the online-help system. To program the trace press the **Compile** button. After successful programming, the file name will be displayed in the trace status window. The default file name is the file currently used by the trigger unit. The file name default extension is ***.ts**. This command can be executed by clicking the **Edit** button in the trace state window. If the state of the trace is

ARM and this command is executed, the state is switched to **OFF** before programming the trigger unit, automatically. Refer to “**Complex Trigger Unit for Nexus MPC5xxx**” (app_ctu_mpc5xxx.pdf) for more information.

See also

- [SETUP.EDITOR](#)
- ▲ ['Release Information' in 'Legacy Release History'](#)

Analyzer.RecordAutoFill

Precision of run-time measurements

Format: **Analyzer.RecordAutoFill AUTO | ON | OFF**

Improves precision of run-time measurement for selective trace (e.g. using **TraceEnable**). Only supported for RH850 AURORA trace.

Analyzer.REMAP

Remap trace port channels

Format: **Analyzer.REMAP [RAW | SiGnal | PIN] <channel> <value>**

Allows the user to remap the trace port channels to a non-standard configuration.

RAW	Deprecated. Use PIN or SiGnal instead.
SiGnal	Remaps <channel> to preprocessor independent signal (data <value>).
PIN	Remaps <channel> to preprocessor dependent pin number <value>.
<channel>	Channel name as defined by trace port.
<value>	New assignment of <channel>.

See also

- [Analyzer.REMAP.CLOCK](#)
- [Analyzer.REMAP.RESet](#)
- [Analyzer.REMAP.state](#)
- ▲ ['Pin Remapping' in 'AutoFocus User's Guide'](#)

Format: Analyzer.REMAP.CLOCK CLK0 | CLK1 | CLK2 | CLK3

Selects one of the four possible clock inputs as trace clock.

See also

- [Analyzer.REMAP](#)
- [Analyzer.REMAP.state](#)

Analyzer.REMAP.RESet

Reset pinout configuration

Format: Analyzer.REMAP.RESet

Resets the preprocessor pinout to its default configuration.

See also

- [Analyzer.REMAP](#)
- [Analyzer.REMAP.state](#)
- ▲ ['Pin Remapping' in 'AutoFocus User's Guide'](#)

Analyzer.REMAP.state

Display remap configuration window

Format: Analyzer.REMAP.state

Displays the remap configuration window.

See also

- [Analyzer.REMAP](#)
- [Analyzer.REMAP.CLOCK](#)
- [Analyzer.REMAP.RESet](#)
- ▲ ['Pin Remapping' in 'AutoFocus User's Guide'](#)

Format: Analyzer.ReProgram [*<file>*]

Programs the trigger unit of the trace with an existing program. The program should be error free, when using this command. To write an error free program, use the command **Analyzer.Program**. Refer to “**Complex Trigger Unit for Nexus MPC5xxx**” (app_ctu_mpc5xxx.pdf) for more information.

Analyzer.SAMPLE

Set AutoFocus sample time offset

Format: Analyzer.SAMPLE [*<channel>*] *<time1>* [*<time2>*]

Use this command to manually configure sample times of the trace channels. It is typically used to restore values previously stored using the store button of the **Trace.ShowFocus** window, or the **STOre ANALYZERFOCUS** command.

<i><channel></i>	Trace signal to be configured If the parameter is omitted, all signals are configured with the <i><time></i> setting.
<i><time1></i>	Parameter Type: Float . The value is interpreted as time in nanoseconds. Sample time offset to trace clock. - Positive value: later - Negative value: earlier
<i><time2></i>	Parameter Type: Float . The value is interpreted as time in nanoseconds. For double data rate mode only: Sample time offset to inverted trace clock (usually this is the sample time relative to the falling edge of the trace clock). If the parameter is omitted, <i><time1></i> is used for both sample time offsets.

The available AutoFocus settings and features depend on the preprocessor used:

Preprocessor:	Features:	
AutoFocus II	configuration: time resolution: max. range: max. inter-channel range:	individual sample time per channel about 78ps per step +/- 63 * 0.078ns $ABS(MAX(time1)-MIN(time1)) < 63*0.078ns$
NEXUS-AutoFocus	configuration: time resolution: max. range: DDR-Mode:	single sample time applied to all channels about 750ps per step +/- 15 * 0.75ns $ABS(time1-time2) < 15*0.75ns$

Analyzer.TOut

Trigger output line

Format:

Analyzer.TOut BusA | EXT

Sets up the trigger output line.

See also

- [Trace](#)

Analyzer.TraceCLOCK

Improve timestamps on PowerTrace

Format:

Analyzer.TraceCLOCK <data_rate>

Usually several records have the same timestamp on PowerTrace (often 8 or 12 records) due to chip internal FIFOs. In the case of a known exported data rate, this command can be used to improve the timestamps for certain trace protocols.

A special use-case of this command is when the AutoFocus preprocessor is connected to a Single-Wire-Trace module (i.e. Trace.PortType SWV). Then this command is used to specify the bit rate. The <data_rate> is then set to 1/(min-bit-size).

See also

- [Trace](#)
- ▲ ['Release Information' in 'Legacy Release History'](#)

Analyzer.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).

Analyzer.Arm Arm the trace

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

Analyzer.AutoArm Arm automatically

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

Analyzer.AutoFocus Calibrate AUTOFOCUS preprocessor

See command [`<trace>.AutoFocus`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 135).

Analyzer.AutoInit Automatic initialization

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

Analyzer.AutoStart Automatic start

See command [`<trace>.AutoStart`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 140).

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

Analyzer.BookMarkToggle

Toggles a single trace bookmark

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

Analyzer.Chart

Display trace contents graphically

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

Analyzer.Chart.Address

Time between program events as a chart

See command [`<trace>.Chart.Address`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 153).

Analyzer.Chart.AddressGROUP

Address group time chart

See command [`<trace>.Chart.AddressGROUP`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 155).

Analyzer.Chart.ChildTREE

Display callee context of a function as chart

See command [`<trace>.Chart.ChildTREE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 156).

Analyzer.Chart.DatasYmbol

Analyze pointer contents graphically

See command [`<trace>.Chart.DatasYmbol`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 157).

See command [`<trace>.Chart.DistriB`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 159).

Analyzer.Chart.Func

Function activity chart

See command [`<trace>.Chart.Func`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 161).

Analyzer.Chart.GROUP

Group activity chart

See command [`<trace>.Chart.GROUP`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 162).

Analyzer.Chart.INTERRUPT

Display interrupt chart

See command [`<trace>.Chart.INTERRUPT`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 163).

Analyzer.Chart.INTERRUPTTREE

Display interrupt nesting

See command [`<trace>.Chart.INTERRUPTTREE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 164).

Analyzer.Chart.Line

Graphical HLL lines analysis

See command [`<trace>.Chart.Line`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 165).

Analyzer.Chart.MODULE

Code execution broken down by module as chart

See command [`<trace>.Chart.MODULE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 166).

See command [<trace>.Chart.Nesting](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 167).

See command [<trace>.Chart.PAddress](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 168).

See command [<trace>.Chart.PROGRAM](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 169).

See command [<trace>.Chart.PsYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 170).

See command [<trace>.Chart.RUNNABLE](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 172).

See command [<trace>.Chart.sYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 173).

See command [<trace>.Chart.TASK](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 176).

Analyzer.Chart.TASKFunc Task related function run-time analysis (legacy)

See command [`<trace>.Chart.TASKFunc`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 177).

Analyzer.Chart.TASKINFO Context ID special messages

See command [`<trace>.Chart.TASKINFO`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 177).

Analyzer.Chart.TASKINTR Display ISR2 time chart (ORTI)

See command [`<trace>.Chart.TASKINTR`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 178).

Analyzer.Chart.TASKKernel Task run-time chart with kernel markers (flat)

See command [`<trace>.Chart.TASKKernel`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 179).

Analyzer.Chart.TASKORINTERRUPT Task and interrupt activity chart

See command [`<trace>.Chart.TASKORINTERRUPT`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 180).

Analyzer.Chart.TASKORINTRState Task and ISR2 state analysis

See command [`<trace>.Chart.TASKORINTRState`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 181).

Analyzer.Chart.TASKSRV Service routine run-time analysis

See command [`<trace>.Chart.TASKSRV`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 182).

See command [`<trace>.Chart.TASKState`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 183).

Analyzer.Chart.TASKVSINTERRUPT

Time chart of interrupted tasks

See command [`<trace>.Chart.TASKVSINTERRUPT`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 185).

Analyzer.Chart.TASKVSINTR

Time chart of task-related interrupts

See command [`<trace>.Chart.TASKVSINTR`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 186).

Analyzer.Chart.TREE

Display function chart as tree view

See command [`<trace>.Chart.TREE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 187).

Analyzer.Chart.Var

Variable chart

See command [`<trace>.Chart.Var`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 188).

Analyzer.Chart.VarState

Variable activity chart

See command [`<trace>.Chart.VarState`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 189).

Analyzer.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).

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

Analyzer.ComPareCODE**Compare trace with memory**

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

Analyzer.CustomTrace**Custom trace**

See command [**<trace>.CustomTrace**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 195).

Analyzer.CustomTraceLoad**Load a DLL for trace analysis/Unload all DLLs**

See command [**<trace>.CustomTraceLoad**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 196).

Analyzer.DISable**Disable the trace**

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

Analyzer.DRAW**Plot trace data against time**

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

Analyzer.DRAW.channel**Plot no-data values against time**

See command [**<trace>.DRAW.channel**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 204).

See command [<trace>.DRAW.Data](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 206).

See command [<trace>.DRAW.Var](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 210).

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

See command [<trace>.EXPORT.ARTI](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 213).

See command [<trace>.EXPORT.ARTIAP](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 214).

See command [<trace>.EXPORT.Ascii](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 215).

See command [<trace>.EXPORT.Bin](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 216).

See command [**<trace>.EXPORT.BRANCHFLOW**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 218).

See command [**<trace>.EXPORT.CSVFunc**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 219).

See command [**<trace>.EXPORT.cycles**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 220).

See command [**<trace>.EXPORT.Func**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 223).

See command [**<trace>.EXPORT.MDF**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 224).

See command [**<trace>.EXPORT.MTV**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 225).

See command [**<trace>.EXPORT.TASK**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 226).

See command [**<trace>.EXPORT.TASKEVENTS**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 227).

Analyzer.EXPORT.TracePort**Export trace packets as recorded at trace port**

See command [**<trace>.EXPORT.TracePort**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 228).

Analyzer.EXPORT.VCD**Export trace data in VCD format**

See command [**<trace>.EXPORT.VCD**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 230).

Analyzer.EXPORT.VERILOG**Export trace data in VERILOG format**

See command [**<trace>.EXPORT.VERILOG**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 231).

Analyzer.EXPORT.VHDL**Export trace data in VHDL format**

See command [**<trace>.EXPORT.VHDL**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 232).

Analyzer.ExtractCODE**Extract code from trace**

See command [**<trace>.ExtractCODE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 232).

Analyzer.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).

Analyzer.Find

Find specified entry in trace

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

Analyzer.FindAll

Find all specified entries in trace

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

Analyzer.FindChange

Search for changes in trace flow

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

Analyzer.FindProgram

Advanced trace search

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

Analyzer.FindReProgram

Activate advanced existing trace search program

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

Analyzer.FindViewProgram

State of advanced trace search programming

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

Analyzer.FLOWPROCESS

Process flowtrace

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

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

Analyzer.Get**Display input level**

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

Analyzer.GOTO**Move cursor to specified trace record**

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

Analyzer.Init**Initialize trace**

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

Analyzer.JOINFILE**Concatenate several trace recordings**

See command [`<trace>.JOINFILE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 246).

Analyzer.List**List trace contents**

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

Analyzer.ListNesting**Analyze function nesting**

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

Analyzer.ListVar**List variable recorded to trace**

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

Analyzer.LOAD

Load trace file for offline processing

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

Analyzer.MERGEFILE

Combine two trace files into one

See command [<trace>.MERGEFILE](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 272).

Analyzer.OFF

Switch off

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

Analyzer.PipeWRITE

Connect to a named pipe to stream trace data

See command [<trace>.PipeWRITE](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 278).

Analyzer.PlatformCLOCK

Set clock for platform traces

See command [<trace>.PlatformCLOCK](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 278).

Analyzer.PortFilter

Specify utilization of trace memory

See command [<trace>.PortFilter](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 279).

Analyzer.PortSize

Set external port size

See command [<trace>.PortSize](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 280).

See command [**<trace>.PortType**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 280).

See command [**<trace>.PROfile**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 282).

See command [**<trace>.PROfile.CTU**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 282).

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

See command [**<trace>.PROfileChart.Address**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 289).

See command [**<trace>.PROfileChart.AddressGROUP**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 290).

See command [**<trace>.PROfileChart.AddressRate**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 292).

See command [<trace>.PROfileChart.COUNTER](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 293).

See command [<trace>.PROfileChart.DatasYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 295).

See command [<trace>.PROfileChart.DIstance](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 296).

See command [<trace>.PROfileChart.DistriB](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 297).

See command [<trace>.PROfileChart.DURation](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 298).

See command [<trace>.PROfileChart.GROUP](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 301).

See command [<trace>.PROfileChart.INTERRUPT](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 302).

See command [<trace>.PROfileChart.Line](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 303).

See command [<trace>.PROfileChart.MODULE](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 304).

See command [<trace>.PROfileChart.PAddress](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 305).

See command [<trace>.PROfileChart.PROGRAM](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 306).

See command [<trace>.PROfileChart.PsYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 307).

See command [<trace>.PROfileChart.Rate](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 309).

See command [<trace>.PROfileChart.RUNNABLE](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 311).

Analyzer.PROfileChart.sYmbol Dynamic program behavior graphically (flat)

See command [<trace>.PROfileChart.sYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 312).

Analyzer.PROfileChart.TASK Dynamic task behavior graphically (flat)

See command [<trace>.PROfileChart.TASK](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 313).

Analyzer.PROfileChart.TASKINFO Context ID special messages

See command [<trace>.PROfileChart.TASKINFO](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 314).

Analyzer.PROfileChart.TASKINTR ISR2 profile chart (ORTI)

See command [<trace>.PROfileChart.TASKINTR](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 315).

Analyzer.PROfileChart.TASKKernel Task profile chart with kernel markers

See command [<trace>.PROfileChart.TASKKernel](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 316).

Analyzer.PROfileChart.TASKORINTERRUPT Task and interrupt profile chart

See command [<trace>.PROfileChart.TASKORINTERRUPT](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 317).

Analyzer.PROfileChart.TASKSRV Profile chart of OS service routines

See command [<trace>.PROfileChart.TASKSRV](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 318).

See command [<trace>.PROfileChart.TASKVSINTERRUPT](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 319).

Analyzer.PROfileChart.TASKVSINTR Profile chart for task-related interrupts

See command [<trace>.PROfileChart.TASKVSINTR](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 320).

Analyzer.PROfileChart.Var Variable profile chart

See command [<trace>.PROfileChart.Var](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 321).

Analyzer.PROfileSTATistic Statistical analysis in a table versus time

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

Analyzer.PROfileSTATistic.Address Statistical analysis for addresses

See command [<trace>.PROfileSTATistic.Address](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 325).

Analyzer.PROfileSTATistic.AddressGROUP Stat. for address groups

See command [<trace>.PROfileSTATistic.AddressGROUP](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 325).

Analyzer.PROfileSTATistic.COUNTER Statistical analysis for counter

See command [<trace>.PROfileSTATistic.COUNTER](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 326).

Analyzer.PROfileSTATistic.DatasYmbol Statistic analysis for pointer content

See command [**<trace>.PROfileSTATistic.DatasYmbol**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 326).

Analyzer.PROfileSTATistic.DistriB Distribution statistical analysis

See command [**<trace>.PROfileSTATistic.DistriB**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 327).

Analyzer.PROfileSTATistic.GROUP Statistical analysis for groups

See command [**<trace>.PROfileSTATistic.GROUP**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 328).

Analyzer.PROfileSTATistic.INTERRUPT Statistical analysis for interrupts

See command [**<trace>.PROfileSTATistic.INTERRUPT**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 329).

Analyzer.PROfileSTATistic.Line Statistical analysis for HLL lines

See command [**<trace>.PROfileSTATistic.Line**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 330).

Analyzer.PROfileSTATistic.MODULE Statistical analysis for modules

See command [**<trace>.PROfileSTATistic.MODULE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 331).

Analyzer.PROfileSTATistic.PAddress Which instr. accessed data address

See command [**<trace>.PROfileSTATistic.PAddress**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 332).

See command [<trace>.PROfileSTATistic.PROGRAM](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 332).

Analyzer.PROfileSTATistic.PsYmbol
address

Which functions accessed data

See command [<trace>.PROfileSTATistic.PsYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 333).

Analyzer.PROfileSTATistic.RUNNABLE

Statistical analysis for runnables

See command [<trace>.PROfileSTATistic.RUNNABLE](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 333).

Analyzer.PROfileSTATistic.sYmbol

Statistical analysis for symbols

See command [<trace>.PROfileSTATistic.sYmbol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 334).

Analyzer.PROfileSTATistic.TASK

Statistical analysis for tasks

See command [<trace>.PROfileSTATistic.TASK](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 335).

Analyzer.PROfileSTATistic.TASKINFO

Context ID special messages

See command [<trace>.PROfileSTATistic.TASKINFO](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 335).

Analyzer.PROfileSTATistic.TASKINTR

Statistical analysis for ISR2 (ORTI)

See command [<trace>.PROfileSTATistic.TASKINTR](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 336).

See command [<trace>.PROfileSTATistic.TASKKernel](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 337).

See command [<trace>.PROfileSTATistic.TASKORINTERRUPT](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 337).

See command [<trace>.PROfileSTATistic.TASKSRV](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 338).

See command [<trace>.PROfileSTATistic.TASKVSINTERRUPT](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 338).

See command [<trace>.PROTOcol](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 339).

See command [<trace>.PROTOcol.Chart](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 339).

See command [<trace>.PROTOcol.Draw](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 341).

Analyzer.PROTOcol.EXPORT

Export trace buffer for user-defined protocol

See command [<trace>.PROTOcol.EXPORT](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 342).

Analyzer.PROTOcol.Find

Find in trace buffer for user-defined protocol

See command [<trace>.PROTOcol.Find](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 343).

Analyzer.PROTOcol.list

Display trace buffer for user-defined protocol

See command [<trace>.PROTOcol.list](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 344).

Analyzer.PROTOcol.PROfileChart

Profile chart for user-defined protocol

See command [<trace>.PROTOcol.PROfileChart](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 347).

Analyzer.PROTOcol.PROfileSTATistic

Profile chart for user-defined protocol

See command [<trace>.PROTOcol.PROfileSTATistic](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 348).

Analyzer.PROTOcol.STATistic

Display statistics for user-defined protocol

See command [<trace>.PROTOcol.STATistic](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 350).

Analyzer.REF

Set reference point for time measurement

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

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

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

See command [**<trace>.SelfArm**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 362).

See command [**<trace>.ShowFocus**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 365).

See command [**<trace>.ShowFocusClockEye**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 368).

See command [**<trace>.ShowFocusEye**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 370).

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

See command [<trace>.SnapShot](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 373).

See command [<trace>.SPY](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 374).

See command [<trace>.state](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 376).

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

See command [<trace>.STATistic.Address](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 387).

See command [<trace>.STATistic.AddressDIstance](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 388).

See command [<trace>.STATistic.AddressDURation](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 389).

See command [**<trace>.STATistic.AddressGROUP**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 391).

See command [**<trace>.STATistic.ChildTREE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 393).

See command [**<trace>.STATistic.COLOR**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 394).

See command [**<trace>.STATistic.CYcle**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 395).

See command [**<trace>.STATistic.DatasYmbol**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 398).

See command [**<trace>.STATistic.DIStance**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 400).

See command [**<trace>.STATistic.DistriB**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 401).

See command [`<trace>.STATistic.DURation`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 402).

See command [`<trace>.STATistic.FIRST`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 404).

See command [`<trace>.STATistic.Func`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 406).

See command [`<trace>.STATistic.FuncDURation`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 422).

See command [`<trace>.STATistic.FuncDURationInternal`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 423).

See command [`<trace>.STATistic.GROUP`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 424).

See command [`<trace>.STATistic.Ignore`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 426).

See command [**<trace>.STATistic.INTERRUPT**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 427).

Analyzer.STATistic.InterruptIsFunctionStatistics interrupt processing

See command [**<trace>.STATistic.InterruptIsFunction**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 428).

Analyzer.STATistic.InterruptIsKernelStatistics interrupt processing

See command [**<trace>.STATistic.InterruptIsKernel**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 430).

Analyzer.STATistic.InterruptIsKernelFunction
processingStatistics interrupt

See command [**<trace>.STATistic.InterruptIsKernelFunction**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 430).

Analyzer.STATistic.InterruptIsTaskswitchStatistics interrupt processing

See command [**<trace>.STATistic.InterruptIsTaskswitch**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 430).

Analyzer.STATistic.INTERRUPTTREEDisplay interrupt nesting

See command [**<trace>.STATistic.INTERRUPTTREE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 431).

Analyzer.STATistic.LASTEnd point for statistic analysis

See command [**<trace>.STATistic.LAST**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 433).

See command [`<trace>.STATistic.Line`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 435).

Analyzer.STATistic.LINKagePer caller statistic of function

See command [`<trace>.STATistic.LINKage`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 436).

Analyzer.STATistic.MeasureAnalyze the performance of a single signal

See command [`<trace>.STATistic.Measure`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 438).

Analyzer.STATistic.MODULECode execution broken down by module

See command [`<trace>.STATistic.MODULE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 440).

Analyzer.STATistic.PAddressWhich instructions accessed data address

See command [`<trace>.STATistic.PAddress`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 441).

Analyzer.STATistic.ParentTREEShow the call context of a function

See command [`<trace>.STATistic.ParentTREE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 442).

Analyzer.STATistic.PROCESSRe-process statistics

See command [`<trace>.STATistic.PROCESS`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 444).

See command [**<trace>.STATistic.PROGRAM**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 445).

See command [**<trace>.STATistic.PsYmbol**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 446).

See command [**<trace>.STATistic.RUNNABLE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 448).

See command [**<trace>.STATistic.RUNNABLEEDURation**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 449).

See command [**<trace>.STATistic.Sort**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 450).

See command [**<trace>.STATistic.sYmbol**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 458).

See command [**<trace>.STATistic.TASK**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 461).

See command [`<trace>.STATistic.TASKFunc`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 464).

See command [`<trace>.STATistic.TASKINFO`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 464).

See command [`<trace>.STATistic.TASKINTR`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 465).

See command [`<trace>.STATistic.TASKKernel`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 466).

See command [`<trace>.STATistic.TASKLOCK`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 469).

See command [`<trace>.STATistic.TASKORINTERRUPT`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 470).

See command [`<trace>.STATistic.TASKORINTRState`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 471).

See command [**<trace>.STATistic.TASKSRV**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 472).

Analyzer.STATistic.TASKStatePerformance analysis

See command [**<trace>.STATistic.TASKState**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 474).

Analyzer.STATistic.TASKStateDURationTask state runtime analysis

See command [**<trace>.STATistic.TASKStateDURation**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 478).

Analyzer.STATistic.TASKTREETree display of task specific functions

See command [**<trace>.STATistic.TASKTREE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 479).

Analyzer.STATistic.TASKVSINTERRUPTStatistic of interrupts, task-related

See command [**<trace>.STATistic.TASKVSINTERRUPT**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 480).

Analyzer.STATistic.TASKVSINTRISR2 statistic (ORTI), task related

See command [**<trace>.STATistic.TASKVSINTR**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 481).

Analyzer.STATistic.TREETree display of nesting function run-time analysis

See command [**<trace>.STATistic.TREE**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 482).

See command [`<trace>.STATistic.Use`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 483).

Analyzer.STATistic.VarStatistic of variable accesses

See command [`<trace>.STATistic.Var`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 484).

Analyzer.STREAMCompressionSelect compression mode for streaming

See command [`<trace>.STREAMCompression`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 485).

Analyzer.STREAMFILESpecify temporary streaming file path

See command [`<trace>.STREAMFILE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 486).

Analyzer.STREAMFileLimitSet size limit for streaming file

See command [`<trace>.STREAMFileLimit`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 487).

Analyzer.STREAMLOADLoad streaming file from disk

See command [`<trace>.STREAMLOAD`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 488).

Analyzer.STREAMSAVESave streaming file to disk

See command [`<trace>.STREAMSAVE`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 490).

See command [`<trace>.TDelay`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 491).

Analyzer.TERMination

Use trace line termination of preprocessor

See command [`<trace>.TERMination`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 493).

Analyzer.TestFocus

Test trace port recording

See command [`<trace>.TestFocus`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 494).

Analyzer.TestFocusClockEye

Scan clock eye

See command [`<trace>.TestFocusClockEye`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 496).

Analyzer.TestFocusEye

Check signal integrity

See command [`<trace>.TestFocusEye`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 497).

Analyzer.TestUtilization

Tests trace port utilization

See command [`<trace>.TestUtilization`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 497).

Analyzer.THreshold

Optimize threshold for trace lines

See command [`<trace>.THreshold`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 498).

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

Analyzer.TraceCONNECT

Select on-chip peripheral sink

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

Analyzer.TRACK

Set tracking record

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

Analyzer.TSElect

Select trigger source

See command [`<trace>.TSElect`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 503).

Analyzer.View

Display single record

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

Analyzer.ZERO

Align timestamps of trace and timing analyzers

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

APU

Auxiliary processing unit

APU (Auxiliary Processing Unit) commands are necessary to debug sub-cores where the debugger is implemented in a library usually written by the manufacturer or any other third party. Most **APU** commands are comparable to common debug commands. Therefore see according commands in the “General Commands Reference Guides” for more detailed information.

Please report any issues regarding the sub-core debugger to its manufacturer. For writing your own subcore debugger, see the “**API for Auxiliary Processing Unit**” (api_apu.pdf) documentation.

See also

- [■ APU.Break](#)
[■ APU.GREP](#)
[■ APU.Register](#)
[■ APU.View](#)
- [■ APU.command](#)
[■ APU.List](#)
[■ APU.RESet](#)
- [■ APU.Data](#)
[■ APU.ListHll](#)
[■ APU.Step](#)
- [■ APU.Go](#)
[■ APU.LOAD](#)
[■ APU.StepHll](#)

▲ 'Introduction' in 'API for Auxiliary Processing Unit'

APU.Break

APU breakpoints

See also

- [■ APU.Break.Delete](#)
[■ APU.Break.RESet](#)
- [■ APU.Break.direct](#)
[■ APU.Break.Set](#)
- [■ APU.Break.Init](#)
[■ APU](#)
- [■ APU.Break.List](#)
[■ APU.command](#)

APU.Break.Delete

Delete APU breakpoint

Format:

APU.Break.Delete [*<address>* | *<range>*] [*<option>*]

Deletes APU breakpoints from list.

Delete the breakpoint at the given address or inside the range. Without a parameter, this command removes all APU breakpoints.

```
APU.Break.Delete           ; Delete all APU breakpoints
```

See also

- [■ APU.Break](#)

Format: **APU.Break.direct** [<address>]

Stops the execution of the APU.

If no address is given the program is aborted at the next assembler instruction. If an address is given a temporary breakpoint is set, which is deleted when hit.

APU.Break.direct 0x2 ; Break if the APU reaches address 0x2.

See also

■ [APU.Break](#)

APU.Break.Init

Initialize APU breakpoint system

Format: **APU.Break.Init**

Removes all temporary breakpoints, set with the [Break](#) or [Go](#) command. Permanent breakpoints are not affected.

See also

■ [APU.Break](#)

APU.Break.List

List APU breakpoints

Format: **APU.Break.List**

Opens the **APU.Break.List** window, listing all APU breakpoints.

See also

■ [APU.Break](#)

Format:	APU.Break.RESet
---------	------------------------

Removes all breakpoints.

See also

■ [APU.Break](#)

APU.Break.Set

Set permanent APU breakpoint

Format:	APU.Break.Set [<address> <range>] [/<break>]
<break>:	Program ReadWrite Read Write

Sets a breakpoint. The availability of on-chip breakpoints depends on the implementation of the APU Debugger and the Sub Core.

For information about the options, see [Break.Set](#).

```
APU.Break.Set 0x2          ; Break at address 0x2.
```

See also

■ [APU.Break](#)

Format: **APU.command** [*<cmd>*]

Executes any APU specific command. Processing is up to the APU, so there is no general list of options and parameters. For more information, please contact the developer of the APU debugger library.

<cmd>

Any command to be sent to the APU library.

See also

- [APU](#)

■ [APU.BREAK](#)

■ [APU.Data](#)

■ [APU.Go](#)

■ [APU.GREP](#)

■ [APU.List](#)

■ [APU.ListHll](#)

■ [APU.LOAD](#)

■ [APU.Register](#)

■ [APU.RESet](#)

■ [APU.Step](#)

■ [APU.StepHll](#)

■ [APU.View](#)

APU.Data

APU data command group

See also

- [APU.Data.dump](#)

■ [APU.Data.List](#)

■ [APU.Data.LOAD](#)

■ [APU.Data.Set](#)

■ [APU](#)

■ [APU.command](#)

APU.Data.dump

Data memory display

Format: **APU.Data.dump** [*<address>* | *<range>* | *<value>*] [*/<option>*]

Opens a memory dump window for the APU core similar to the [Data.dump](#) command.

For more information on the parameters and options, see [Data.dump](#).

See also

- [APU.Data](#)

Format: **APU.Data.List** [*<address>* | *<range>*] [*/<option>*]

Opens a source listing window for the APU core similar to the [List.auto](#) command.

For more information on the parameters, see [List.auto](#).

See also

■ [APU.Data](#)

APU.Data.LOAD

Load file

Format: **APU.Data.LOAD.***<file_format>* [*<file>*] [*<address>* | *<range>*] [*/<option>*]

<file_format>: **auto** | **Binary** | **IntelHex** | **S1record** | **S2record** | **S3record** | ...

Loads a binary file of type *<file_format>* to the APU memory. Note that not all formats are supported since there is no support for a specific compiler format.

For more information on the parameters, see [Data.LOAD](#).

See also

■ [APU.Data](#)

Format:	APU.Data.Set [<i><address></i> <i><range></i>] % <i><format></i> [<i><data></i> <i><string></i>]
<i><format></i> :	Byte Word Long

Modifies the APU memory.

<address>, etc. For more information on the parameters, see [Data.Set](#).

See also

- [APU.Data](#)

APU.Go

Start the APU

Format:	APU.Go [<i><address></i>]
---------	--

Starts the real-time execution of the APU core.

An address can be specified as temporary breakpoint where execution stops. The breakpoint is deleted when it is hit.

```
APU.Go 0x2                                ; Run until address 0x2 is reached.
```

See also

- [APU](#)
- [APU.command](#)

Format:	APU.GREP <string> [<file>] [/<option>]
<option>:	Word Case ...

The command **APU.GREP** allows to search for a specific string in a (all) source files. It does basically the same as the **Data.GREP** command, but instead of opening **Data.*** windows for displaying results it will open **APU.*** windows.

Word	Search for the whole word.
Case	Perform a case sensitive search.
...	For the other options, see APU.List .

APU.GREP "i"	; search for "i" in all source files
APU.GREP "i" diabc1.c	; search for "i" in diabc1.c
APU.GREP "i" *.* /Word	; search for the word "i" in all source files

See also

- [APU](#)
- [APU.command](#)
- [Data.GREP](#)
- ▲ 'Release Information' in 'Legacy Release History'

APU.List

View program

Format:	APU.List [<address> <range>] [/<option>]
---------	---

Displays the APU program at the given address.

<address>	If <address> is omitted, the program is listed at the current PC. For more information on the parameters, see List.auto .
-----------	--

See also

- [APU](#)
- [APU.command](#)

Format: **APU.ListHll** [*<address>* | *<range>*] [*/<option>*]

Display the APU program in source format at the given address.

<address>

If *<address>* is omitted, the program is listed at the current PC. For more information on the parameters, see [List.auto](#).

See also

■ [APU](#)

■ [APU.command](#)

APU.LOAD

Load APU library

Format: **APU.LOAD** *<file>* [*<string>* | *<address>* | *<value>*]

Loads the APU library from *<file>*. Depending on the implementation, one or more parameters are passed optionally or mandatory.

See also

■ [APU](#)

■ [APU.command](#)

APU.Register

Show APU register window

Format: **APU.Register** [*<file>*] [*<tree_search_item>*]

Opens the **APU.Register** window, the content is defined in a peripheral file. For information on how to create peripheral files, see:

- [PER](#) command group
- ["Peripheral Files Programming Commands"](#) (per_prog.pdf)

Only registers that are memory mapped and accessible from the Main Core Debugger can be displayed.

<file> If *<file>* is omitted, `perapureg.per` is loaded by default. For more information on displaying peripheral files, see [PER.view](#).

See also

- [APU.Register.Set](#)
- [APU.Register.view](#)
- [APU](#)
- [APU.command](#)

APU.Register.Set

Register modification

Format: **APU.Register.Set** *<register_name>* [*<value>*]

Modifies an APU core register. This command is currently not implemented.

See also

- [APU.Register](#)

APU.Register.view

Register display

Format: **APU.Register.view** [*/<option>*]

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

Displays the APU core registers, the content is defined in a peripheral file.

<option> For a description of the options, see [PER.view](#).

See also

- [APU.Register](#)

Format: **APU.RESet**

Unloads the APU library and resets APU debugger.

See also

■ [APU](#)

■ [APU.command](#)

APU.Step

Single-stepping

Format: **APU.Step**

Executes one program step in assembler mode.

See also

■ [APU](#)

■ [APU.command](#)

APU.StepHll

HLL single-stepping

Format: **APU.StepHll**

Executes one program step in high-level language mode.

See also

■ [APU](#)

■ [APU.command](#)

Format:	APU.View [<i><file></i>] [/ <i><tree_search_item></i>]
---------	--

Opens the peripheral window for the APU core.

<file>

If *<file>* is omitted, `perapu.per` is loaded by default. For more information on displaying peripheral files, see [PER.view](#).

For more information about how to create peripheral files, see:

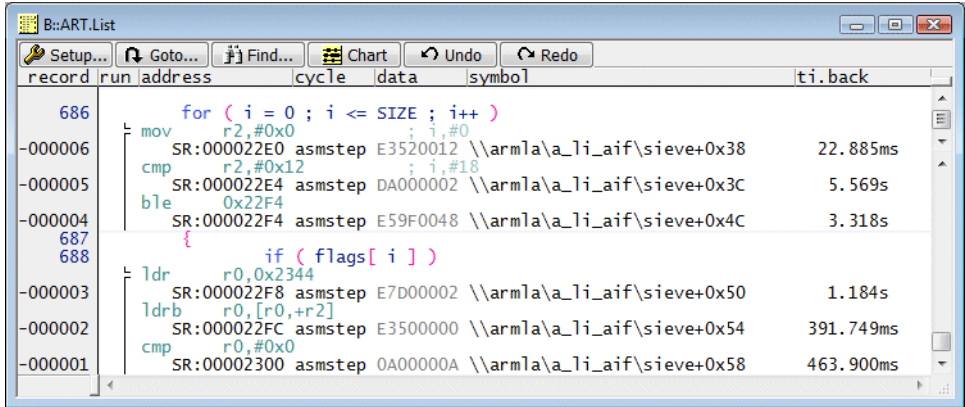
- [PER](#) command group
- ["Peripheral Files Programming Commands"](#) (`per_prog.pdf`)

Only registers that are memory mapped and accessible from the Main Core Debugger can be displayed.

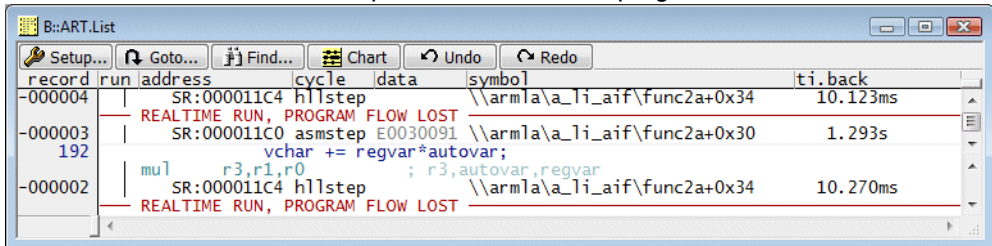
See also

- [APU](#)
- [APU.command](#)

ART (Advanced Register Trace) is the most basic TRACE32 trace method. When debugging in mixed (**Mode.Mix**) or assembly mode (**Mode.Asm**), ART logs all CPU registers when single stepping on assembler level. This information is used to provide a single step trace.

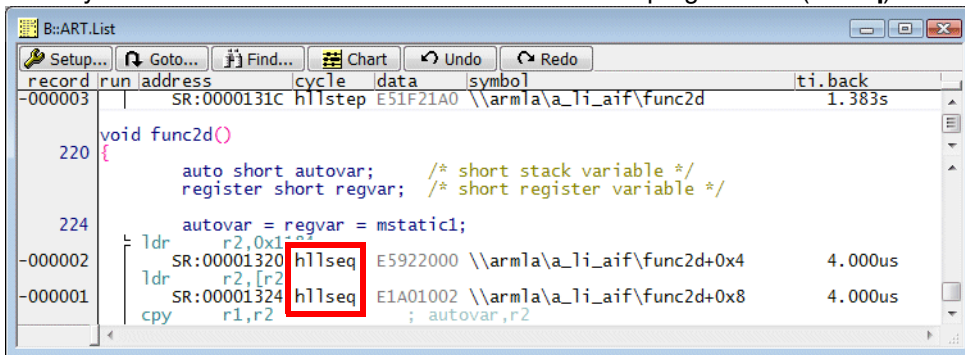


When performing high-level single steps, the program execution is however started and stopped at the next high-level code line. Due to this proceeding, the trace method ART cannot log the CPU registers for each instruction executed. As a result part of the assembler program flow is lost in the ART trace.



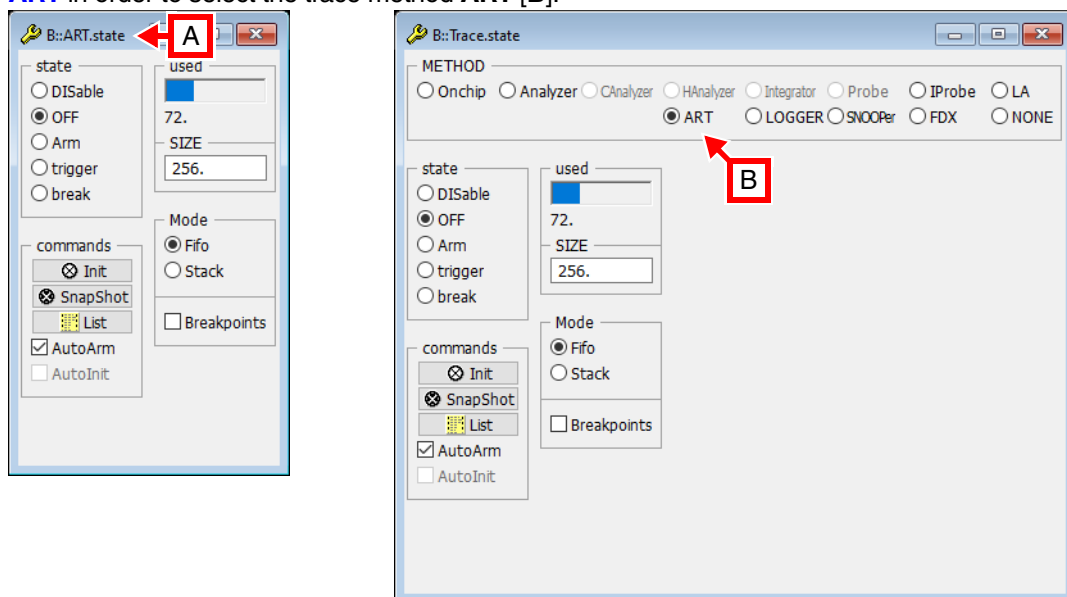
The default size of the ART is 256 single steps (records). The maximum size is 65536.

In many cases TRACE32 can reconstruct the assembler program flow (**hllseq**).



For selecting and configuring the trace method ART, use the TRACE32 command line of a PRACTICE script (*.cmm) or the **ART.state** window [A].

Alternatively, use the **Trace.state** window: Click the option **ART** or execute the command **Trace.METHOD ART** in order to select the trace method **ART** [B].



ART trace can also be used for code coverage. Please refer to “**ART Mode Code Coverage**” in Application Note for Trace-Based Code Coverage, page 77 (app_code_coverage.pdf) for more information.

The chapter “**ART-specific Trace Commands**”, page 76 describes the ART-specific configuration commands. While the chapter “**ART Trace Commands**”, page 77 lists the ART trace analysis and display commands, which are shared with other TRACE32 trace methods.

See also

■ [COVErage.METHOD](#) ■ [Trace.METHOD](#)

▲ ['Release Information' in 'Legacy Release History'](#)

ART.Mode

Set the trace operation mode

Format:	ART.Mode [<i><mode></i>]
<i><mode></i> :	Fifo Stack

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.

See also

■ [<trace>.Mode](#)

ART.Arm

Arm the trace

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

ART.AutoArm

Arm automatically

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

ART.AutoInit

Automatic initialization

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

ART.BookMark

Set a bookmark in trace listing

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

ART.Chart

Display trace contents graphically

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

ART.ComPare

Compare trace contents

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

See command [`<trace>.PROTOcol`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 339).

See command [`<trace>.PROTOcol.Chart`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 339).

See command [`<trace>.PROTOcol.Draw`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 341).

See command [`<trace>.PROTOcol.EXPORT`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 342).

See command [`<trace>.PROTOcol.Find`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 343).

See command [`<trace>.PROTOcol.list`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 344).

See command [`<trace>.PROTOcol.PROfileChart`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 347).

See command [`<trace>.PROTOcol.PROfileSTATistic`](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 348).

See command [**<trace>.PROTOcol.STATistic**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 350).

ART.REFSet reference point for time measurement

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

ART.RESetReset command

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

ART.SAVESave trace for postprocessing in TRACE32

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

ART.SelfArmAutomatic restart of trace recording

See command [**<trace>.SelfArm**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 362).

ART.SIZEDefine buffer size

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

ART.SnapShotRestart trace capturing once

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

ART.stateDisplay trace configuration window

See command [**<trace>.state**](#) in 'General Commands Reference Guide T' (general_ref_t.pdf, page 376).

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

ART.Timing

Waveform of trace buffer

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

ART.TRACK

Set tracking record

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

ART.View

Display single record

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

ART.ZERO

Align timestamps of trace and timing analyzers

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

AutoSTOre

Save and restore settings (history, GUI, etc.) automatically

Format:	AutoSTOre <file> [<item> ...] [/<option>]
<item>:	default ALL SYStem ... <history_and_gui_settings>
<option>:	NoDate

Restores settings from the previous TRACE32 session and stores specified settings automatically at the end of a TRACE32 session.

When **AutoSTOre** is executed, the following happens:

- **AutoSTOre** calls the PRACTICE script specified by <file>. The script is executed as if it was executed by the **DO** command.
- **AutoSTOre** registers the specified items to be stored when the TRACE32 session ends. The settings will be stored to the PRACTICE script specified by <file>.

The **AutoSTOre** command should be used only once per TRACE32 session. Usually it is used within the PRACTICE script file autostore.cmm (which you should not edit), but you can also use it again in the PRACTICE script files system-settings.cmm (in the TRACE32 system directory) or user-settings.cmm (in the user settings directory, on Windows %APPDATA%\TRACE32 or ~/.trace32 otherwise).

Alternatively, you can save settings manually with the **STOre** command and restore them with the **DO** command. Therefore you might want to use **SETUP.QUITDO** to execute **STOre** at the end of a TRACE32 session.

<file> or ,	User-defined path and file name. If a comma is used instead, TRACE32 saves the file in the temporary directory of TRACE32. See example . The auto-generated file name consists of the return value of the OS.ID() function and the string <code>store.cmm</code> .
<item>, <option>	For a detailed description of <item> and <option>, refer to the STOre command.
<history_and_gui_settings>	See ' AutoSTOre ' in ' PowerView Command Reference '.
default	When executing the AutoSTOre command without parameters (default), all settings will be stored, except for the window setting .

ALL	Stores all settings except the Break and FLAG information. For storing Break and FLAG information, see example .
SYStem ...	All other keywords refer to the command settings of the same name.

Example 1: Restore settings saved by **AutoSTOre** in the previous TRACE32 session and register the saving of the following items when TRACE32 gets closed: Command history (**HISTORY**), the address and trace bookmarks (**BOOKMARK**) and the help bookmarks (**HELP**).

```
AutoSTOre , HISTORY BOOKMARK HELP
```

Example 2: For storing **Break** and **FLAG** information, please use the command:

```
STOre test.cmm ALL Break FLAG
```

See also

- [BookMark.List](#)
- [ClipSTOre](#)
- [STOre](#)
- [SETUP.STOre](#)

AVX

AVX registers (Advanced Vector Extension)

Intel® x86

The **AVX** command group is used to display and modify AVX (Advanced Vector Extension) registers of Intel® x86.

See also

- [■ AVX.Init](#)[■ AVX.Set](#)[■ AVX.view](#)[■ AVX512](#)
- [□ AVX\(\)](#)
- [▲ 'Command Groups for Special Registers' in 'Intel® x86/x64 Debugger'](#)
- [▲ 'AVX Functions' in 'General Function Reference'](#)

AVX.Init

Initialize AVX registers

Intel® x86

Format:

AVX.Init

Sets the AVX registers to their default values.

See also

- [■ AVX](#)[■ AVX.view](#)

AVX.Set

Modify AVX registers

Intel® x86

Format:

AVX.Set <register> <value> ... [/<option>]

Modifies the AVX registers.

<option>

For a description of the options, see [Register.view](#).

See also

- [■ AVX](#)[■ AVX.view](#)

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

Displays the AVX registers.

<option> For a description of the options, see [Register.view](#).

See also

- [AVX](#) ■ [AVX.Init](#) ■ [AVX.Set](#)
- ▲ 'Release Information' in 'Legacy Release History'

AVX512

AVX512 registers (Advanced Vector Extension)

Intel® x86

The **AVX512** command group is used to display and modify AVX512 (Advanced Vector Extension) registers of Intel® x86.

See also

- [■ AVX512.Init](#)[■ AVX512.Set](#)[■ AVX512.view](#)[■ AVX](#)
- [□ AVX512\(\)](#)
- [▲ 'Command Groups for Special Registers' in 'Intel® x86/x64 Debugger'](#)
- [▲ 'AVX Functions' in 'General Function Reference'](#)

AVX512.Init

Initialize AVX512 registers

Intel® x86

Format:

AVX512.Init

Sets the AVX512 registers to their default values.

See also

- [■ AVX512](#)[■ AVX512.view](#)

AVX512.Set

Modify AVX512 registers

Intel® x86

Format:

AVX512.Set *<register>* *<value>* ... [*<option>*]

Modifies the AVX512 registers.

<option>

For a description of the options, see [Register.view](#).

See also

- [■ AVX512](#)[■ AVX512.view](#)

Format: **AVX512.view** [/<option>]

Displays the AVX512 registers.

<option> For a description of the options, see [Register.view](#).

See also

- [■ AVX512](#)
- [■ AVX512.Init](#)
- [■ AVX512.Set](#)
- [▲ 'Release Information' in 'Legacy Release History'](#)