RAM Trace Port

MANUAL

RAM Trace Port

Version 06-Jun-2024

Overview

The main purpose of the Ram Trace Port (RTP) is to allow for non-intrusive tracing of write- or read accesses to the internal RAMs of Texas Instruments (TI) automotive microcontrollers. Besides that, also CPU controlled data tracing is supported.

In the non-intrusive mode - called TraceMode - the RTP module is capable of tracing three different RAMs as well as the peripheral bus. Tracing of each RAM (or peripheral bus) in turn is limited to two independent trace regions, which makes a total of eight different trace regions. In order to configure these trace regions, RTP commands starting with **RTP.TraceMode** must be used.

Trace data can also be generated by writing to a dedicated capture register via CPU or DMA. This mode in turn is called DirectDataMode and can be configured via the **RTP.DirectDataMode** commands. It is worth mentioning that the DirectDataMode also can be used to trace data reads from the RAMs of the microcontroller; however the TraceMode should be preferred to that kind of configuration of the DirectDataMode.

FAQ

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

Quick Start

1. Turn off the target.

2. Please follow the hardware installation instructions in **trace_arm_etm.pdf** to set up the trace hardware.

3. If you are using the AutoFocus II preprocessor, follow the instructions under 4). For the AutoFocus 600 MIPI preprocessor proceed with 5).

4. AutoFocus II: Connect the plug of the **preprocessor (LA-3905)** marked 'TRACE A' to the appropriate socket of the **RTP to Mictor38 converter (LA-3835)** using the mictor flex extension delivered with your preprocessor. Continue with 7).

5. AutoFocus 600 MIPI: Attach the header of the preprocessor (or **flex extension LA-1228**) to the **MIPI 60 to Mictor38 converter (LA-3769)**.

6. AutoFocus 600 MIPI: Fit the **MIPI 60 to Mictor 38 converter** into the **RTP to Mictor38 converter (LA-3835)**.

7. Depending on your target board and debug cable, either attach the debug cable to the target board directly or make use of the ARM-20 connector provided by the **RTP to Mictor38 converter**.

8. Attach the **RTP to Mictor38 converter** to the target.

9. Start the TRACE32 software.

10. Power up the target and connect to the target CPU.

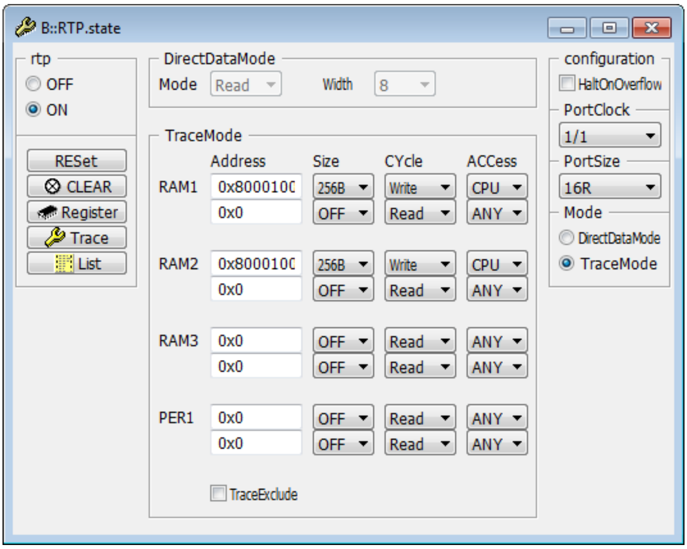
11. Run your start-up script(s).

12. Make sure your target is stopped.

13. Configure the RTP module:

    - Optional: Configure the parallel ETM and RTP asynchronous mode: **<trace>.PortType TPIU+RTP** (AutoFocus II only).

- Run **RTP.state** to display the RTP setup window, and/or



rtp: ○ OFF ● ON

DirectDataMode: Mode Read, Width 8

TraceMode:

	Address	Size	CYcle	ACCess
RAM1	0x8000100	256B	Write	CPU
	0x0	OFF	Read	ANY
RAM2	0x8000100	256B	Write	CPU
	0x0	OFF	Read	ANY
RAM3	0x0	OFF	Read	ANY
	0x0	OFF	Read	ANY
PER1	0x0	OFF	Read	ANY
	0x0	OFF	Read	ANY

□ TraceExclude

configuration: □ HaltOnOverflow

PortClock: 1/1

PortSize: 16R

Mode: ○ DirectDataMode ● TraceMode

- Make use of the RTP commands described in the ’**Commands**’ section of this manual.

14. Execute some of the target’s code.

15. Stop the target.

16. Display the trace data via **RTPAnalyzer.List**

17. In order to start a new trace, clear the current trace window via **RTP.CLEAR.**

18. It further on is recommended to read the trace documentation in **general_ref_t.pdf**.

Commands

RTP — Ram trace port (RTP)

See also

- RTP.CLEAR
- RTP.OFF
- RTP.RESet
- RTP.DirectDataMode
- RTP.ON
- RTP.state
- RTP.HaltOnOverflow
- RTP.PortClock
- RTP.TraceMode
- RTP.Mode
- RTP.PortSize

RTP.CLEAR — Clear tracebuffer

Format: **RTP.CLEAR**

Clears the tracebuffer; all entries of the trace window will be removed.

See also

- RTP
- RTP.state

To activate either the complex or the simple trace mode, use the **RTP.Mode** command.

See also

- RTP.DirectDataMode.Mode
- RTP.DirectDataMode.Width
- RTP
- RTP.Mode
- RTP.state

RTP.DirectDataMode.Mode — Direct data mode read/write

Format:	**RTP.DirectDataMode.Mode** [**Read** | **Write**]

Read (default)	Traces data read operations on RAM (addresses are discarded).
Write	Writes to the Direct Data Mode Write register (RTPDDMW) are traced.

See also

- RTP.DirectDataMode

RTP.DirectDataMode.Width — Trace width in bits

Format:	**RTP.DirectDataMode.Width** [**8** | **16** | **32**]

Default: 8

Sets the number of bits to be traced per read- or write access. If the access width is greater than the configured trace width, the upper bits will be truncated. If on the other hand the access width is smaller than the configured trace width, the upper bits will be filled with indeterminate data.

See also

- RTP.DirectDataMode

RTP.HaltOnOverflow — Halt system on RTP FIFO overflow

Format:	**RTP.HaltOnOverflow** [**ON** | **OFF**]

Default: OFF

If [OFF], data written to an already full RTP FIFO will be discarded. Otherwise the initiator of a data write (CPU or DMA) will be kept in a halted state until the FIFO is ready to receive new data.

See also

- RTP
- RTP.state

RTP.Mode — Select the trace mode

Format:	**RTP.Mode** [**TraceMode** | **DirectDataMode**]

Default: TraceMode

TraceMode **DirectDataMode**	Selects one of the two possible trace modes (**RTP.TraceMode** and **RTP.DirectDataMode** commands).

See also

- RTP
- RTP.DirectDataMode
- RTP.state
- RTP.TraceMode

RTP.OFF — Disables the RTP module

Format:	**RTP.OFF**

Simply turns off the RTP module.

See also

- RTP
- RTP.state

RTP.ON

Activates the RTP module

Format: **RTP.ON**

The RTP module is enabled.

See also

- ■ RTP
- ■ RTP.state
- ▲ 'Commands' in 'RAM Trace Port'

RTP.PortSize

Size of RTP data port

Format: **RTP.PortSize** [**2** | **4** | **8** | **16** | **2R** | **4R** | **8R** | **16R**]

Default: 2

Defines the number of parallel RTP data pins. Suffix 'R' denotes a remapped pin configuration that matches the pinout of the **RTP to Mictor38 converter (LA-3835)**.

See also

- ■ RTP
- ■ RTP.state

RTP.PortClock

Configure RTPCLK

Format: **RTP.PortClock** [**1/1** | **1/2** | **1/3** | **1/4** | **1/5** | **1/6** | **1/7** | **1/8**]

Default: 1/1

Sets the resulting frequency of RTPCLK, which is derived from HCLK: RTPCLK = [PortClock] * HCLK. HCLK must not exceed 100MHz.

See also

- ■ RTP
- ■ RTP.state

RTP.RESet Resets RTP settings

Format: **RTP.RESet**

Resets the RTP settings and the RTP module.

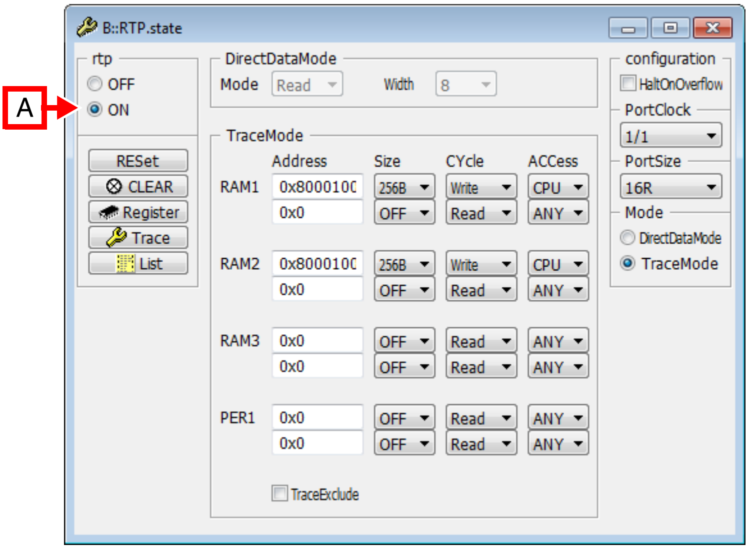
See also

- RTP
- RTP.state

RTP.state Display RTP setup

Format: **RTP.state**

Displays the RTP setup window.



rtp
○ OFF
● ON

DirectDataMode
Mode: Read Width: 8

TraceMode

	Address	Size	CYcle	ACCess
RAM1	0x800010C	256B	Write	CPU
	0x0	OFF	Read	ANY
RAM2	0x800010C	256B	Write	CPU
	0x0	OFF	Read	ANY
RAM3	0x0	OFF	Read	ANY
	0x0	OFF	Read	ANY
PER1	0x0	OFF	Read	ANY
	0x0	OFF	Read	ANY

□ TraceExclude

configuration
□ HaltOnOverflow
PortClock: 1/1
PortSize: 16R
Mode
○ DirectDataMode
● TraceMode

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

See also

- RTP
- RTP.CLEAR
- RTP.DirectDataMode
- RTP.HaltOnOverflow
- RTP.Mode
- RTP.OFF
- RTP.ON
- RTP.PortClock
- RTP.PortSize
- RTP.RESet
- RTP.TraceMode

To activate either the complex or the simple trace mode, use the **RTP.Mode** command.

See also

- RTP.TraceMode.TraceExclude
- RTP.Mode
- RTP
- RTP.state

RTP.TraceMode.RAM<x>.SECTion<y> Configures a trace region

Format:	**RTP.TraceMode.RAM**<*x*>**.SECTion**<*y*> <*parameter*>
<*x*>:	[**1** | **2** | **3**]
<*y*>:	[**1** | **2**]
<*parameter*>:	**ACCess** [**ANY** | **CPU** | **DMA**] **Address** <*address*> **CYcle** [**Read** | **Write**] **Size** [**OFF** | **256** | **512** | **1k** | **2k** | **4k** | **8k** | **16k** | **32k** | **64k** | **128k** | **256k**]

Configures section #<*y*> of RAM #<*x*>. In case a peripheral region instead of a RAM shall be traced, RTP.TraceMode.PER1.SECTion<*y*> must be used.

ACCess [**ANY** | **CPU** | **DMA**]	Defined whether only CPU, only DMA or both types of accesses to the specified RAM region are traced. (default: ANY)
Address	Base address of trace region. (default: 0)
CYcle [**Read** | **Write**]	Either read- or write accesses to the specified RAM region are traced. (default: Read)
Size	Sets the size of the trace region. All data accesses to the RAM from **RTP.TraceMode.RAM<x>.SECTion<y>.Address** to **RTP.TraceMode.RAM<x>.SECTion<y>.Address** + **RTP.TraceMode.RAM<x>.SECTion<y>.Size** are traced. (default: OFF) **NOTE:** The size of the peripheral trace regions is limited to 128k.

```
;Example configuration:
;CPU write accesses to RAM1 region 0x200 to 0x600 to be traced.

RTP.TraceMode.RAM1.SECTion1.ACCess CPU         ; Only trace CPU
                                               ; accesses. Instead of
                                               ; SECTion1 also
                                               ; SECTion2 could be used.

RTP.TraceMode.RAM1.SECTion1.CYcle Write        ; Only trace write
                                               ; accesses.

RTP.TraceMode.RAM1.Section1.Address 0x200      ; Base address = 0x200

RTP.TraceMode.RAM1.SECTion1.Size 1k            ; Trace region = 0x200 to
                                               ; 0x600

RTP.ON                                         ; Turn on RTP module.
```

RTP.TraceMode.TraceExclude   Invert all trace regions

Format: **RTP.TraceMode.INVert** [**ON** | **OFF**]

OFF (default)	Data accesses to regions defined by **RTP.TraceMode.RAM<x>.SECTion<x>.Address** and **RTP.TraceMode.RAM<x>.SECTion<x>.Size** are traced.
ON	Data accesses to the area outside the regions defined by **RTP.TraceMode.RAM<x>.SECTion<x>.Address** and **RTP.TraceMode.RAM<x>.SECTion<x>.Size** are traced.

See also

- RTP.TraceMode