LAUTERBACH
DEVELOPMENT TOOLS

Intel® Processor Trace

Release 09.2024

MANUAL

Intel® Processor Trace

Version 05-Oct-2024

Configuration

Selective Tracing (optional)

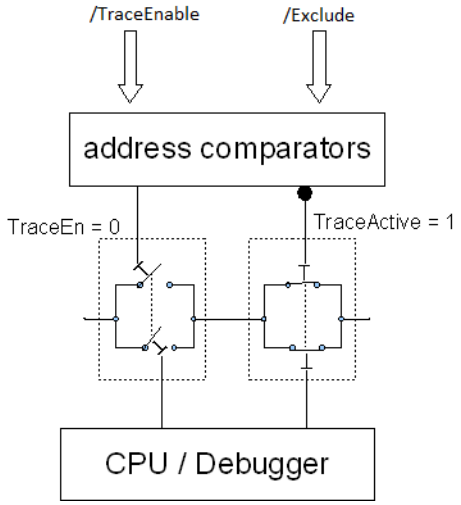
The following command is available to selectively enable or disable the IPT trace by means of the Instruction Pointer (IP):

Break.Set <*range*> [*/*<*option*>] **/Program** Filter by IP

<option>	**Description**
TraceEnable	Enables the trace whenever the IP is within the specified range.
Exclude	Disables the trace whenever the IP is within the specified range. Must be preceded by a /TraceEnable.

Without any option, this command works as described in **Break.Set**. The /Program option is mandatory!

The IPT infrastructure allows for up to two IP filter breakpoints, which can be of different or same kind. Please refer to the schematic below:



Two register bits control the generation of trace data: *TraceEnable* and *TraceActive*. Both must be set to '1' in order to enable the trace (AND operation). They can either be accessed by the IP address comparators or by the CPU / debugger.

Without any interference by the CPU, the debugger sets *TraceEnable=0* and *TraceActive=1* as defaults. As long as the IP executes code within the */TraceEnable* range, *TraceEnable* will be '1'. Otherwise it will be '0'. Correspondingly *TraceActive* becomes '0' within the */TraceEnable /Exclude* range and '1' otherwise.

This leads to the following trace scenarios:

	0 TraceEnable breakpoint	**1 TraceEnable breakpoint**	**2 TraceEnable breakpoints**
0 Exclude breakpoint	IPT is always on.	IPT is enabled while the IP is within the specified /TraceEnable range.	IPT is enabled while the IP is within one of the two specified /TraceEnable ranges.
1 Exclude breakpoint	IPT is enabled while the IP is outside the specified /TraceEnable range.	IPT is enabled while the IP is within the /TraceEnable range. The user may manually set TraceEnable=1 (via CPU or debugger) for the /Exclude option to take effect.	
2 Exclude breakpoints	IPT is enabled while the IP is outside the two specified /TraceEnable ranges.		

NOTE:	The two ranges must not overlap, otherwise the behavior is undefined!

Tracing to Memory

In order to trace to memory, the debugger has to be informed about the data format first. This is done via

Onchip.Buffer.IPT	Trace data in memory is IPT.

(See also **Onchip.Buffer.LBR** and **Onchip.Buffer.BTS**).

After that the base address in RAM must be specified to which the trace data will be dumped. Make sure the buffer is reserved memory and will not be written by any application code on the target!

Onchip.Buffer.BASE <*base*>	Memory base address for trace data.

Default: 0x0

The size of the buffer is configured via **Onchip.Buffer.SIZE**.

NOTE:	Only make use of this command when IPT is disabled (**IPT.OFF**)!

Memory Buffer Size

When in onchip mode, this command defines the size of the trace buffer in bytes. Also see **Onchip.Buffer.BASE.** The size must be in range 0x0 - 0x1000000.

Onchip.Buffer.SIZE <*size*>	Total buffer size in bytes.

Default: 0x0

NOTE:	Only make use of this command when IPT is disabled (**IPT.OFF**)!

Example Script

This example script assumes that the target is stopped and the Lauterbach sieve demo has been loaded.

```
; Trace core with STP master ID = 0x80
IPT.TraceID 0x80

IPT.ON

; Optionally: Determine traceport configuration automatically.
Analyzer.AutoArm ON
Analyzer.AutoFocus

Trace.RESet
Trace.Init
Break.RESet

; Only trace function 'sieve' in the demo.
Break.Set var.range(sieve) /TraceEnable

; Alternatively: Trace all code except 'sieve' function.
;Break.Set var.range(sieve) /TraceEnable /Exlcude

Go
;...
Break

; Display trace results.
Trace.List
```

IPT — Intel® Processor Trace (IPT)

The Intel® Processor Trace (IPT) works similar to the LBR and BTS feature of Ix86 based cores (see **"CPU specific Onchip Trace Commands"** (debugger_x86.pdf). It compresses flowtrace information in a more efficient way to be exported by an external trace port.

IPT is implemented on selected architectures only and therefore the following commands are not available to all Ix86 based cores.

IPT.CLEAR — Clear memory for new core configuration

Format: **IPT.CLEAR**

Clears and sets up onchip memory after core configuration has been changed via **Core.Number** or **Core.Assign**. Only needed if **Trace.Method** is **Onchip**.

IPT.CLOCK — CPU core clock

Format: **IPT.CLOCK** *<frequency>*

Clock frequency of the CPU(s). This value will be used to compute execution times in cycle accurate tracing mode.

IPT.CompressReturn — Compression of near return addresses

Format: **IPT.CompressReturn** [**ON** | **OFF**]

Default: OFF.

If **ON**, use LIP compression instead of an Target IP packet to indicate the return address of a near call.

Format: **IPT.CR3** *<address>*

Default: Empty

Enables tracing if CR3 (page table pointer) matches the specified address value. In order to disable selective tracing, *<address>* must be left empty: **IPT.CR3**

IPT.CycleAccurate Enable cycle accurate tracing

Format: **IPT.CycleAccurate** [**ON** | **OFF**]

Default: OFF.

If ON, every instruction in the trace will have its own cycle count or a share of accumulated cycle counts (see **IPT.CycleCountThreshold**). Otherwise the current cycle count will only be output periodically, which helps to reduce bandwidth.

IPT.CycleCountThreshold Cycle count for cycle accurate tracing

Format: **IPT.CycleCountThreshold** *<number>*

Default: 0.

Cycle accurate trace packets will only be emitted after <number> cycles. Cycle accurate traceing must be enabled by **IPT.CycleAccurate** **ON**.

Format:	**IPT.DataTrace** [/*<def>*]
<def>:	**ON** | **OFF** | **Address** | **ReadAddress** | **WriteAddress** | **Data** | **ReadData** | **WriteData** | **Read** | **Write**

Enable data tracing.

IPT.DataTraceFUP Trace IP when data trace packet is generated

Format:	**IPT.DataTraceFUP** [**ON** | **OFF**]

If enabled, aditionally add the current IP to the trace. This setting only has an effect if **IPT.DataTrace** is enabled.

IPT.EXPORTBASE IPT output region

Format:	**IPT.EXPORTBASE** *<base_address> <limit_mask>*

Default: 0xFDC00000 0x7F

<base_address>	Base address at which IPT packets will be written to.
<limit_mask>	Size in bytes of the output region. The lowest 5 (RTIT) or 6 (IPT) bits are forced to ‘1’.

Format: **IPT.IgnoreGERR** [**ON** | **OFF**]

Default: OFF.

GERR errors usually are emitted to indicate a crammed FIFO. Setting this option to **OFF** will suppress any fifo overflow errors in the **Trace.List** window.

IPT.LessPackets Do not create IPT packets on certain circumstances

Format: **IPT.LessPackets** [**ON** | **OFF**]

Default: OFF.

If enabled, certain packets will only be generate when the following criteria are met:

SuperTimeSync MiniTimeCounter	ContextEn=1 and FilterEn=1
Periodic Cycle Counter	FilterEn=1
Cycle Counter increment	FilterEn=1

IPT.MiniTimeCounter Enable "MiniTimeCounter" packets

Format: **IPT.MiniTimeCounter** [**ON** | **OFF**]

Default: OFF.

Enables the generation of MiniTimeCounter (MTC) packets.

IPT.OFF — Switch IPT off

Format: **IPT.OFF**

Disables IPT tracing.

IPT.ON — Switch IPT on

Format: **IPT.ON**

Enables IPT tracing.

IPT.PortRoute — Selection of trace HW

Format: **IPT.PortRoute [AUTO | Analyzer | CAnalyzer]**

Default: AUTO.

Used to configure Lauterbach trace HW. Only required if PowerTrace(Analyzer) and Combiprobe (CAnalyzer) are both attached to the debugger.

IPT.Register — Show IPT registers

Format: **IPT.Register**

Displays IPT registers in a new **PER.view** window.

Format:	**IPT.PacketCount** *<value>*
<value>:	**2047** **4095** **8191** **16383**

Default: 8191

Used to define the minimum packet byte count after which a PSB (Packet Stream Boundary) is output. PSBs help an external decoder to synchronize with the IPT trace by inserting a unique pattern into the stream.

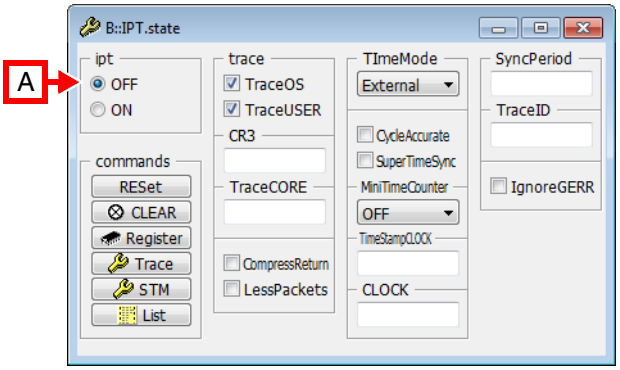
IPT.RESet Reset IPT settings

Format:	**IPT.RESet**

All IPT settings are reset to their defaults.

Format: **IPT.state**

Opens the IPT configuration window:



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

IPT.SuperTimeSync Enable "SuperTimeSync" packets

Format: **IPT.SuperTimeSync** [**OFF** | **14-7** | **16-9** | **18-11** | **20-13**]

Default: OFF.

Enables the generation of SuperTimeSync (STS) packets.

OFF	No packets will be generated.
14-7	Bits 14:7 of the HW timestamp counter will be exported.
16-9	Bits 16:9 of the HW timestamp counter will be exported.
18-11	Bits 18:11 of the HW timestamp counter will be exported.
20-13	Bits 20:13 of the HW timestamp counter will be exported.

Format: **IPT.SyncPeriod** *<value>*

Deprecated. Use **IPT.PacketCount**.

IPT.TImeMode Timestamp mode

Format: **IPT.TImeMode** *<mode>*

<mode>: **OFF** | **External** | **ExternalInterpolated** |
MiniTimeCounter
TimeStampCounter
CycleAccurate
CycleAccurate+External
CycleAccurate+ExternalTrack
CycleAccurate+TimeStampCounter

See **ETM.TImeMode**.

IPT.TimeStampCLOCK Timestamp frequency

Format: **IPT.TimeStampCLOCK** *<frequency>*

Timing information in the **Trace.List** window will be displayed in SI units instead of cycles.

IPT.TraceCORE Trace selected cores only

Format: **IPT.TraceCORE** *<core1> <core2>*

Default: All on

All core traces except the stated one(s) will be switched off. In case of off-chip traces this command helps to reduce bandwidth.

Format:	**IPT.TraceID** *<value>* | *<bitmask>*

If Intel® PT information is wrapped in STP, TRACE32 needs to know which master/channel IDs are used.

<value>	*<value>* is a 32-bit number. The first 16 bits represent the master ID, the last 16 bits represent the channel ID
<bitmask>	bitmask representation of *<value>*

Example 1: Each core has its own master ID.

```
IPT.Trace 0x00800000    ; master ID 0x80 is used to export PT trace
                        ; information for core 0

IPT.Trace 0x008x0000    ; master ID 0x80, 0x81, 0x82 … are used to export
                        ; PT trace information
                        ; master ID 0x80 represents core 0
                        ; the other master IDs consecutively represent
                        ; core 1 to core 15
```

Example 2: Each core has its own channel ID, all cores use the same master ID.

```
IPT.Trace 0x0080000x    ; master ID 0x80 is used to export PT trace
                        ; information
                        ; channel ID 0 represents core 0
                        ; the other channel IDs consecutively represent
                        ; core 1 to core 15
```

IPT.TraceOS Filtering by current privilege level 0

Format:	**IPT.TraceOS** [**ON** | **OFF**]

Default: ON.

Enables or disables tracing when CPU is in Current Privilege Level 0.

Format: **IPT.TraceUSER** [**ON** | **OFF**]

Default: ON.

Enables or disables tracing when CPU is in Current Privilege Levels 1-3.

IPT.TSC Enable timestamp counter packets

Format: **IPT.TSC** [**ON** | **OFF**]

Default: OFF.

Enables or disables Timestamp Counter (TSC) packets.

Intel® MIPI60 Connector

Target pinout specified by Intel®.

Signal	Pin	Pin	Signal
VREF_DEBUG	1	2	TMS
TCK0	3	4	TDO
TDI	5	6	No Connect
HOOK[6]=Reset In	7	8	10 kOhm to GND
TRST_N	9	10	PREQ_N
PRDY_N	11	12	VTREF_TRACE
PTI_0_CLK	13	14	PTI_1_CLK
POD_PRESENT1_N	15	16	GND
POD_PRESENT2_N	17	18	PTI_1_DATA[0]
PTI_0_DATA[0]	19	20	PTI_1_DATA[1]
PTI_0_DATA[1]	21	22	PTI_1_DATA[2]
PTI_0_DATA[2]	23	24	PTI_1_DATA[3]
PTI_0_DATA[3]	25	26	PTI_1_DATA[4]/PTI_2_DATA[0]
PTI_0_DATA[4]	27	28	PTI_1_DATA[5]/PTI_2_DATA[1]
PTI_0_DATA[5]	29	30	PTI_1_DATA[6]/PTI_2_DATA[2]
PTI_0_DATA[6]	31	32	PTI_1_DATA[7]/PTI_2_DATA[3]
PTI_0_DATA[7]	33	34	HOOK[7]=Reset Out
PTI_0_DATA[8]/PTI_3_DATA[0]	35	36	HOOK[3]=Boot Stall
PTI_0_DATA[9]/PTI_3_DATA[1]	37	38	HOOK[2]=CPU Boot Stall
PTI_0_DATA[10]/PTI_3_DATA[2]	39	40	HOOK[1]=Power Button
PTI_0_DATA[11]/PTI_3_DATA[3]	41	42	HOOK[0]=PWRGOOD
PTI_0_DATA[12]/PTI_3_DATA[4]	43	44	HOOK[5]
PTI_0_DATA[13]/PTI_3_DATA[5]	45	46	HOOK[4]
PTI_0_DATA[14]/PTI_3_DATA[6]	47	48	I2C_SCL
PTI_0_DATA[15]/PTI_3_DATA[7]	49	50	I2C_SDA
TCK1	51	52	GND
TRIG_INOUT	53	54	DBG_UART_TX
TRIG_IN	55	56	DBG_UART_RX
GND	57	58	GND
PTI_3_CLK	59	60	PTI_2_CLK

MIPI60-P Connector

Not all pins of the Intel® MIPI60 connector are connected to the *Prepro. for Intel® PT AUTOFOCUS MIPI60-P*. The connected pins are displayed with their name on a gray background in the picture below.

Signal	Pin	Pin	Signal
VREF_DEBUG	1	2	TMS
TCK0	3	4	TDO
TDI	5	6	No Connect
No Connect	7	8	No Connect
TRST_N	9	10	PREQ_N
PRDY_N	11	12	VREF_TRACE
PTI_0_CLK	13	14	No Connect
No Connect	15	16	No Connect
No Connect	17	18	No Connect
PTI_0_DATA[0]	19	20	No Connect
PTI_0_DATA[1]	21	22	No Connect
PTI_0_DATA[2]	23	24	No Connect
PTI_0_DATA[3]	25	26	No Connect
PTI_0_DATA[4]	27	28	No Connect
PTI_0_DATA[5]	29	30	No Connect
PTI_0_DATA[6]	31	32	No Connect
PTI_0_DATA[7]	33	34	No Connect
PTI_0_DATA[8]	35	36	No Connect
PTI_0_DATA[9]	37	38	No Connect
PTI_0_DATA[10]	39	40	No Connect
PTI_0_DATA[11]	41	42	No Connect
PTI_0_DATA[12]	43	44	No Connect
PTI_0_DATA[13]	45	46	No Connect
PTI_0_DATA[14]	47	48	No Connect
PTI_0_DATA[15]	49	50	No Connect
No Connect	51	52	No Connect
No Connect	53	54	No Connect
No Connect	55	56	No Connect
No Connect	57	58	No Connect
No Connect	59	60	No Connect