

# Intel® Trace Hub (Intel® TH)

Developer's Manual

---

*Revision 2.1.2*

October 2017



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication, or disclosure is subject to restrictions stated in Intel's Software License Agreement, or in the case of software delivered to the government, in accordance with the software license agreement as defined in FAR 52.227-7013.

Intel technologies may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.

The Intel® Trace Hub may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

The code names presented in this document are only for use by Intel to identify products, technologies, or services in development that have not been made commercially available to the public, i.e., announced, launched, or shipped. They are not "commercial" names for products or services and are not intended to function as trademarks.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at <http://www.intel.com/design/literature.htm>.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See [http://www.intel.com/products/processor\\_number](http://www.intel.com/products/processor_number) for details.

Intel, the Intel® Trace Hub, Intel® Atom™, and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. MIPI® service marks and logo marks are owned by MIPI Alliance, Inc. and any use of such marks by Intel Corporation is under license.

\* Other brands and names may be claimed as the property of others.

Copyright © 2017 Intel Corporation. All rights reserved.



# Contents

<b>1</b>	<b>Introduction .....</b>	<b>20</b>
1.1	Terminology.....	20
<b>2</b>	<b>Feature Set .....</b>	<b>22</b>
2.1	Features.....	22
2.1.1	Software Trace Hub Features.....	22
2.1.2	Intel® Trace Hub PTI Port Features .....	22
2.2	Trace Sources .....	22
2.3	Trace Destinations.....	23
2.4	Primary Fabric Interface Features .....	23
2.5	Sideband Fabric Interface Features.....	23
2.6	TAP Interface Features .....	23
2.7	Consistent Programming Model.....	23
2.8	Error Handling and Survivability .....	23
<b>3</b>	<b>Functional Description.....</b>	<b>24</b>
3.1	Overview.....	24
3.1.1	Functional Blocks.....	25
3.1.2	Output Ports.....	26
3.2	MTB/CSR Memory Map .....	26
<b>4</b>	<b>Software Trace Hub .....</b>	<b>29</b>
4.1	STH MMIO Regions.....	30
4.1.1	Software Trace Memory Region (STMR) .....	31
4.1.2	Firmware Trace Memory Region.....	35
4.1.3	Intel Processor Trace Memory Region .....	36
4.2	Master Number Assignment and Allocation.....	40
4.3	Lossy Operation .....	44
4.4	Triggering and Events .....	46
4.4.1	Event Match and Mask.....	46
4.4.2	Trigger with Timestamp Packet Generation.....	46
4.5	Time Stamping.....	47
<b>5</b>	<b>DTF Bridge .....</b>	<b>48</b>
<b>6</b>	<b>Global Trace Hub .....</b>	<b>49</b>
6.1	Overview.....	49
6.2	Functional description.....	50
6.2.1	Input Buffers.....	51
6.2.2	Data Switch .....	53
6.2.3	MIPI® STP Encoder.....	55
6.2.4	Byte Packing Buffer.....	62
6.2.5	STP Maintenance Unit.....	63
6.2.6	Low Power Path.....	64
6.2.7	DTF Input.....	65
6.2.8	DCI Trace Handler.....	65
6.2.9	Trigger Unit.....	65
6.2.10	Automatic Trace Source Enable/Disable .....	69
<b>7</b>	<b>Timestamp Counter Unit.....</b>	<b>70</b>
7.1	Intel Trace Hub Timestamp .....	71
7.2	CTC Tracking Unit.....	71
<b>8</b>	<b>Memory Storage Unit.....</b>	<b>72</b>
8.1	Functional Description .....	72
8.1.1	MSU Overview.....	72



8.1.2	MSU Operational Modes.....	73
8.1.3	Multi-Block Memory Windows .....	75
8.1.4	Multi-Window Memory Buffer.....	75
8.1.5	Memory Block Header Description.....	77
8.1.6	Operational Details.....	79
8.1.7	DCI Trace Handler.....	81
<b>9</b>	<b>DCI Trace Handler.....</b>	<b>83</b>
9.1	Functional Requirements and Limitations .....	83
9.1.1	Requirements.....	83
9.1.2	Limitations .....	84
9.2	Trace Streaming Modes .....	84
9.2.1	"High Speed" USB Mode.....	84
9.2.2	"Low speed" BSSB Mode.....	84
9.3	Switching Trace Streaming Modes .....	85
9.4	End of Trace handling.....	85
9.5	Error Handling.....	85
9.5.1	DbC.Trace Device Unresponsive.....	85
9.5.2	DCI Bridge Unresponsive.....	85
9.5.3	Unexpected Reads from DbC .....	86
9.5.4	Unexpected Credit Returns from DbC.....	86
9.5.5	Unexpected Credit Returns from DCI Bridge .....	86
9.5.6	Reads to a Non-Sequential or Unexpected Address from DbC.....	86
<b>10</b>	<b>Common Trigger Sequencer .....</b>	<b>87</b>
10.1	Functional Description .....	87
10.1.1	CTS Global Counter/Timer Resources.....	89
10.1.2	External Input Events.....	92
10.1.3	Sequencer Actions.....	92
10.1.4	Sequencer Control and Status .....	95
10.1.5	Clause and State Operation .....	97
10.1.6	Global Triggering Considerations .....	98
10.2	CTS State Machine .....	100
<b>11</b>	<b>Parallel Trace Interface Port.....</b>	<b>102</b>
11.1	PTI Port Software Support.....	103
11.2	PTI Port Calibration/Training Sequence.....	103
11.3	Functional Description.....	103
11.3.1	Intel Trace Hub Trace Transfer Through the PTI Port.....	103
11.3.2	MIPI-PTI Protocol and Terminology.....	103
11.3.3	Programming Model .....	109
11.3.4	Performance/Bandwidth Analysis .....	109
<b>12</b>	<b>Host Interface .....</b>	<b>110</b>
12.1	Fabric Multi-Root Space Support.....	110
12.2	Error Handling and Reporting .....	111
12.2.1	Always Unsupported Requests .....	112
12.2.2	Programming Model Unsupported Requests .....	112
12.2.3	Erroneous Downstream Completions.....	113
12.2.4	Miscellaneous Downstream Erroneous Transactions .....	113
<b>13</b>	<b>Programming Model.....</b>	<b>115</b>
13.1	Configuring Trace Destinations .....	115
13.1.1	Trace to PTI .....	115
13.1.2	Trace to DCI, BSSB Mode .....	115
13.1.3	Trace to DCI, DbC (USB) Mode .....	116
13.1.4	Trace to System Memory, Single-Block/CSR-Driven Mode .....	116
13.1.5	Trace to System Memory, Multi-Block/Linked-List Mode .....	117
13.1.6	Internal Buffer Mode .....	117



13.2	Configuring Trace Sources.....	118
13.2.1	Visualization of Internal Signals (VIS).....	118
13.2.2	On-Die Logic Analyzer (ODLA).....	118
13.2.3	SoC CHAP (SoCHAP).....	118
13.2.4	Software Trace Hub (STH) .....	118
13.3	Mapping Trace Sources to Trace Destinations .....	118
13.4	Configuring the Trigger Unit .....	119
13.4.1	Using the Trigger Unit .....	119
13.4.2	Manual Override.....	119
13.4.3	Intel Trace Hub Software Reset .....	119
<b>14</b>	<b>Register Description .....</b>	<b>121</b>
14.1	Register Attributes Definitions .....	121
14.2	Global Trace Hub Registers.....	122
14.2.1	Global Trace Hub Registers Summary.....	122
14.2.2	GTHOPT0: GTH Output Ports 0-3 .....	124
14.2.3	GTHOPT1: GTH Output Ports 4-7 .....	126
14.2.4	SWDEST_0: Switching Destination [0] .....	129
14.2.5	SWDEST_1: Switching Destination [1].....	130
14.2.6	SWDEST_2: Switching Destination [2].....	131
14.2.7	SWDEST_3: Switching Destination [3].....	132
14.2.8	SWDEST_4: Switching Destination [4].....	133
14.2.9	SWDEST_5: Switching Destination [5].....	134
14.2.10	SWDEST_6: Switching Destination [6].....	136
14.2.11	SWDEST_7: Switching Destination [7].....	137
14.2.12	SWDEST_8: Switching Destination [8].....	138
14.2.13	SWDEST_9: Switching Destination [9].....	139
14.2.14	SWDEST_10: Switching Destination [10].....	140
14.2.15	SWDEST_11: Switching Destination [11].....	141
14.2.16	SWDEST_12: Switching Destination [12].....	142
14.2.17	SWDEST_13: Switching Destination [13].....	143
14.2.18	SWDEST_14: Switching Destination [14].....	144
14.2.19	SWDEST_15: Switching Destination [15].....	145
14.2.20	SWDEST_16: Switching Destination [16].....	146
14.2.21	SWDEST_17: Switching Destination [17].....	147
14.2.22	SWDEST_18: Switching Destination [18].....	148
14.2.23	SWDEST_19: Switching Destination [19].....	149
14.2.24	SWDEST_20: Switching Destination [20].....	150
14.2.25	SWDEST_21: Switching Destination [21].....	151
14.2.26	SWDEST_22: Switching Destination [22].....	152
14.2.27	SWDEST_23: Switching Destination [23].....	153
14.2.28	SWDEST_24: Switching Destination [24].....	154
14.2.29	SWDEST_25: Switching Destination [25].....	155
14.2.30	SWDEST_26: Switching Destination [26].....	156
14.2.31	SWDEST_27: Switching Destination [27].....	157
14.2.32	SWDEST_28: Switching Destination [28].....	158
14.2.33	SWDEST_29: Switching Destination [29].....	159
14.2.34	SWDEST_30: Switching Destination [30].....	160
14.2.35	SWDEST_31: Switching Destination [31].....	161
14.2.36	GSWDEST: General Software Trace Destination .....	162
14.2.37	LWM0: Low WaterMark for Sources 0-7 .....	162
14.2.38	LWM1: Low WaterMark for Sources 7-15 .....	163
14.2.39	GTH_INFO_1: GTH Parameter Info 1 .....	163
14.2.40	GTH_MISC: GTH Miscellaneous Register .....	163
14.2.41	SMCR0: STP Maintenance Control Register 0 .....	164
14.2.42	SMCR1: STP Maintenance Control Register 1 .....	165
14.2.43	SMCR2: STP Maintenance Control Register 2 .....	165



14.2.44	SMCR3: STP Maintenance Control Register 3 .....	166
14.2.45	PGD0: Programmable Grant Duration Register 0.....	166
14.2.46	PGD1: Programmable Grant Duration Register 1.....	167
14.2.47	SCR: Source Control Register.....	168
14.2.48	GTHFRQ: GTH Frequency Register.....	169
14.2.49	BPB_FRAME_SIZE: Byte Packing Buffer max frame size .....	169
14.2.50	GTHSTAT: GTH Status Register .....	170
14.2.51	SCR2: Source Control Register 2 .....	172
14.2.52	DESTOVR: Destination Override Register.....	173
14.2.53	SCRPD0: ScratchPad[0] Register.....	173
14.2.54	SCRPD1: ScratchPad[1] Register.....	174
14.2.55	SCRPD2: ScratchPad[2] Register.....	175
14.2.56	SCRPD3: ScratchPad[3] Register.....	175
14.2.57	NDB_CTRL: NPK DTF Control Register .....	176
14.2.58	BPB_CSR0: BPB Control/Status Register 0.....	176
14.2.59	BPB_CSR1: BPB Control/Status Register 0.....	177
14.2.60	BPB_CSR1: BPB Control/Status Register 1.....	177
14.2.61	LPP_CTL: LPP Control Register.....	177
14.3	STH Registers .....	179
14.3.1	STH Registers Summary.....	179
14.3.2	STHCAPO: STH Capability 0 .....	180
14.3.3	STHCAPI: STH Capability 1 .....	180
14.3.4	TRIG: Create Trigger Packet.....	181
14.3.5	TRIG_TS: Create Trigger Packet with Time Stamp .....	181
14.3.6	XSYNC: Create Cross-Synchronization Packet .....	181
14.3.7	XSYNC_TS: Create Cross-Synchronization Packet w Time Stamp .....	181
14.3.8	GERR: Create General Error Packet.....	182
14.3.9	TRIGTSPP: TRIG_TS packet payload.....	182
14.3.10	BDC: Backpressure Duration Counter .....	182
14.3.11	DPLC: Data Packet Lost Counter.....	182
14.3.12	EVOAMCH: Event 0 Address Match Register .....	183
14.3.13	EVOAMSK: Event 0 Address Mask Register.....	183
14.3.14	EVODMCH: Event 0 Data Match Register .....	184
14.3.15	EVOCTRL: Event 0 Match/Mask Control Register.....	184
14.3.16	EV1AMCH: Event 1 Address Match Register .....	185
14.3.17	EV1AMSK: Event 1 Address Mask Register.....	185
14.3.18	EV1DMCH: Event 1 Data Match Register.....	185
14.3.19	EV1CTRL: Event 1 Match/Mask Control Register.....	186
14.3.20	EV2AMCH: Event 2 Address Match Register .....	187
14.3.21	EV2AMSK: Event 2 Address Mask Register.....	187
14.3.22	EV2DMCH: Event 2 Data Match Register .....	187
14.3.23	EV2CTRL: Event 2 Match/Mask Control Register.....	188
14.3.24	EV3AMCH: Event 3 Address Match Register .....	189
14.3.25	EV3AMSK: Event 3 Address Mask Register.....	189
14.3.26	EV3DMCH: Event 3 Data Match Register .....	189
14.3.27	EV3CTRL: Event 3 Match/Mask Control Register.....	190
14.3.28	STHMISC: STH MISC Control register 1 .....	191
14.3.29	STHCAPI2: STH Capability 2 .....	191
14.4	TSCU Registers .....	192
14.4.1	TSCU Registers Summary.....	192
14.4.2	TSUCTRL: TSCU Control Register.....	192
14.4.3	TSCUSTAT: TSCU Status Register.....	193
14.4.4	NPKTSCSSL: NPKTSC Snapshot Lower Register.....	194
14.4.5	NPKTSCSSU: NPKTSC Snapshot Upper Register.....	194
14.5	MSU Registers.....	195
14.5.1	MSU Registers Summary .....	195



14.5.2	MSUPARAMS: MSU Parameter Register.....	196
14.5.3	MINTCTL: MSU Interrupt Control Register.....	196
14.5.4	MSUSTATUS: MSU Status Register .....	197
14.5.5	MSUORDT: MSU OBM Residual Data Timer Register.....	198
14.5.6	MSUSPARE: MSU Spare Register.....	198
14.5.7	MSCOCTL: MSCn Control Register.....	198
14.5.8	MSCOSTS: MSCn Status Register.....	200
14.5.9	MSCOBAR: MSCn Base Address Register.....	201
14.5.10	MSCODETSZ: MSCn Memory Buffer Size .....	201
14.5.11	MSCOMWP: MSC0 Lower Memory Write Pointer Register.....	201
14.5.12	MSCOTBRP: MSC0 Trace Buffer Read Pointer .....	201
14.5.13	MSCOTBWP: MSC0 Trace Buffer Write Pointer.....	202
14.5.14	MSCONWSA: MSC0 Next Window Starting Address.....	202
14.5.15	MSCOMMCR: MSCn MMIO Mode Control Register.....	202
14.5.16	MSCOOEWM: MSCn OBM Entry Watermark .....	202
14.5.17	MSCOOXWM: MSCn OBM Exit Watermark.....	203
14.5.18	MSCOFL: MSCn Fill Level .....	203
14.5.19	MSCOUBAR: MSC0 Upper Base Address Register.....	203
14.5.20	MSCOUMWP: MSC0 Upper Memory Write Pointer.....	204
14.5.21	MSCONWUSA: MSC0 Next Window Upper Starting Address .....	204
14.5.22	MSCOSPARE: MSCn Spare Register.....	204
14.5.23	MSC1CTL: MSCn Control Register.....	204
14.5.24	MSC1STS: MSCn Status Register.....	206
14.5.25	MSC1BAR: MSCn Base Address Register.....	207
14.5.26	MSC1DESTSZ: MSCn Memory Buffer Size .....	207
14.5.27	MSC1MWP: MSC1 Lower Memory Write Pointer Register .....	208
14.5.28	MSC1TBRP: MSCn Trace Buffer Read Pointer .....	208
14.5.29	MSC1TBWP: MSCn Trace Buffer Write Pointer .....	208
14.5.30	MSC1NWSA: MSCn Next Window Starting Address.....	208
14.5.31	MSC1MMCR: MSCn MM1 Mode Control Register .....	209
14.5.32	MSC1OEWM: MSCn OBM Entry Watermark .....	209
14.5.33	MSC1OXWM: MSCn OBM Exit Watermark.....	209
14.5.34	MSC1FL: MSCn Fill Level .....	209
14.5.35	MSC1UBAR: MSC1 Upper Base Address Register.....	210
14.5.36	MSC1UMWP: MSC1 Upper Memory Write Pointer.....	210
14.5.37	MSC1NWUSA: MSC0 Next Window Upper Starting Address .....	211
14.5.38	MSC1SPARE: MSCn Spare Register.....	212
14.6	DCI Trace Handler Registers.....	213
14.6.1	DCI Trace Handler Registers Summary .....	213
14.6.2	STREAMCFG1: NPKH Streaming Configuration .....	213
14.6.3	STREAMCFG2: Streaming Configuration 2.....	214
14.6.4	DBCSTSCMD: DBC.Trace Status Command.....	214
14.6.5	DBCSTSDATA: DBC Status Data .....	215
14.6.6	EXISTSCMD: EXI Bridge Status Command .....	215
14.6.7	EXISTSDATA: Exi Status Data .....	215
14.6.8	TIMEOUT: NPKH Time out register.....	215
14.6.9	BRIDGE_BASE_ADDR_HI: EXI Bridge Base Address Hi register .....	216
14.6.10	BRIDGE_BASE_ADDR_LO: EXI Bridge Base Address Lo register .....	216
14.6.11	DBC_BASE_ADDR_HI: DBC Bridge Base Address Hi register .....	216
14.6.12	DBC_BASE_ADDR_LO: DBC Base Address Lo register .....	216
14.6.13	NPKHSTS: NPKH Status register.....	216
14.6.14	NPKH_RETRY: NPKH Retry Register .....	217
14.7	CTS Registers.....	218
14.7.1	CTS Registers Summary .....	218
14.7.2	CLAUSE_EVENT_MODE_C0S0: Clause Event Mode Register C0S0 .....	223
14.7.3	CLAUSE_EVENT_MODE_C1S0: Clause Event Mode Register C1S0 .....	223



14.7.4	CLAUSE_EVENT_MODE_C2S0: Clause Event Mode Register C2S0 .....	224
14.7.5	CLAUSE_EVENT_MODE_C3S0: Clause Event Mode Register C3S0 .....	225
14.7.6	CLAUSE_EVENT_MODE_C0S1: Clause Event Mode Register C0S1 .....	226
14.7.7	CLAUSE_EVENT_MODE_C1S1: Clause Event Mode Register C1S1 .....	227
14.7.8	CLAUSE_EVENT_MODE_C2S1: Clause Event Mode Register C2S1 .....	228
14.7.9	CLAUSE_EVENT_MODE_C3S1: Clause Event Mode Register C3S1 .....	228
14.7.10	CLAUSE_EVENT_MODE_C0S2: Clause Event Mode Register C0S2 .....	229
14.7.11	CLAUSE_EVENT_MODE_C1S2: Clause Event Mode Register C1S2 .....	230
14.7.12	CLAUSE_EVENT_MODE_C2S2: Clause Event Mode Register C2S2 .....	231
14.7.13	CLAUSE_EVENT_MODE_C3S2: Clause Event Mode Register C3S2 .....	232
14.7.14	CLAUSE_EVENT_MODE_C0S3: Clause Event Mode Register C0S3 .....	233
14.7.15	CLAUSE_EVENT_MODE_C1S3: Clause Event Mode Register C1S3 .....	233
14.7.16	CLAUSE_EVENT_MODE_C2S3: Clause Event Mode Register C2S3 .....	234
14.7.17	CLAUSE_EVENT_MODE_C3S3: Clause Event Mode Register C3S3 .....	235
14.7.18	CLAUSE_EVENT_MODE_C0S4: Clause Event Mode Register C0S4 .....	236
14.7.19	CLAUSE_EVENT_MODE_C1S4: Clause Event Mode Register C1S4 .....	237
14.7.20	CLAUSE_EVENT_MODE_C2S4: Clause Event Mode Register C2S4 .....	238
14.7.21	CLAUSE_EVENT_MODE_C3S4: Clause Event Mode Register C3S4 .....	239
14.7.22	CLAUSE_EVENT_ENABLE_C0S0: Clause Event Enable Register C0S0.....	240
14.7.23	CLAUSE_EVENT_ENABLE_C1S0: Clause Event Enable Register C1S0.....	241
14.7.24	CLAUSE_EVENT_ENABLE_C2S0: Clause Event Enable Register C2S0.....	242
14.7.25	CLAUSE_EVENT_ENABLE_C3S0: Clause Event Enable Register C3S0.....	243
14.7.26	CLAUSE_EVENT_ENABLE_C0S1: Clause Event Enable Register C0S1.....	244
14.7.27	CLAUSE_EVENT_ENABLE_C1S1: Clause Event Enable Register C1S1.....	245
14.7.28	CLAUSE_EVENT_ENABLE_C2S1: Clause Event Enable Register C2S1.....	246
14.7.29	CLAUSE_EVENT_ENABLE_C3S1: Clause Event Enable Register C3S1.....	247
14.7.30	CLAUSE_EVENT_ENABLE_C0S2: Clause Event Enable Register C0S2.....	248
14.7.31	CLAUSE_EVENT_ENABLE_C1S2: Clause Event Enable Register C1S2.....	249
14.7.32	CLAUSE_EVENT_ENABLE_C2S2: Clause Event Enable Register C2S2.....	250
14.7.33	CLAUSE_EVENT_ENABLE_C3S2: Clause Event Enable Register C3S2.....	251
14.7.34	CLAUSE_EVENT_ENABLE_C0S3: Clause Event Enable Register C0S3.....	252
14.7.35	CLAUSE_EVENT_ENABLE_C1S3: Clause Event Enable Register C1S3.....	253
14.7.36	CLAUSE_EVENT_ENABLE_C2S3: Clause Event Enable Register C2S3.....	254
14.7.37	CLAUSE_EVENT_ENABLE_C3S3: Clause Event Enable Register C3S3.....	255
14.7.38	CLAUSE_EVENT_ENABLE_C0S4: Clause Event Enable Register C0S4.....	256
14.7.39	CLAUSE_EVENT_ENABLE_C1S4: Clause Event Enable Register C1S4.....	257
14.7.40	CLAUSE_EVENT_ENABLE_C2S4: Clause Event Enable Register C2S4.....	258
14.7.41	CLAUSE_EVENT_ENABLE_C3S4: Clause Event Enable Register C3S4.....	259
14.7.42	CLAUSE_ACTION_CONTROL_C0S0: Clause Action Control Registers C0S0260	
14.7.43	CLAUSE_ACTION_CONTROL_C2S2: Clause Action Control Registers C2S2261	
14.7.44	CLAUSE_ACTION_CONTROL_C1S0: Clause Action Control Registers C1S0262	
14.7.45	CLAUSE_ACTION_CONTROL_C1S2: Clause Action Control Registers C1S2263	
14.7.46	CLAUSE_ACTION_CONTROL_C2S0: Clause Action Control Registers C2S0263	
14.7.47	CLAUSE_ACTION_CONTROL_C3S0: Clause Action Control Registers C3S0266	
14.7.48	CLAUSE_ACTION_CONTROL_C0S1: Clause Action Control Registers C0S1268	
14.7.49	CLAUSE_ACTION_CONTROL_C1S1: Clause Action Control Registers C1S1271	
14.7.50	CLAUSE_ACTION_CONTROL_C2S1: Clause Action Control Registers C2S1273	
14.7.51	CLAUSE_ACTION_CONTROL_C3S1: Clause Action Control Registers C3S1276	
14.7.52	CLAUSE_ACTION_CONTROL_C0S2: Clause Action Control Registers C0S2278	
14.7.53	CLAUSE_ACTION_CONTROL_C1S2: Clause Action Control Registers C1S2281	
14.7.54	CLAUSE_ACTION_CONTROL_C2S2: Clause Action Control Registers C2S2283	
14.7.55	CLAUSE_ACTION_CONTROL_C3S2: Clause Action Control Registers C3S2286	
14.7.56	CLAUSE_ACTION_CONTROL_C0S3: Clause Action Control Registers C0S3288	
14.7.57	CLAUSE_ACTION_CONTROL_C1S3: Clause Action Control Registers C1S3291	
14.7.58	CLAUSE_ACTION_CONTROL_C2S3: Clause Action Control Registers C2S3293	
14.7.59	CLAUSE_ACTION_CONTROL_C3S3: Clause Action Control Registers C3S3296	



14.7.60	CLAUSE_ACTION_CONTROL_C0S4: Clause Action Control Registers C0S4	298
14.7.61	CLAUSE_ACTION_CONTROL_C1S4: Clause Action Control Registers C1S4	301
14.7.62	CLAUSE_ACTION_CONTROL_C2S4: Clause Action Control Registers C2S4	303
14.7.63	CLAUSE_ACTION_CONTROL_C3S4: Clause Action Control Registers C3S4	306
14.7.64	SCT0_CONTROL: SCT0 Control Register .....	308
14.7.65	SCT1_CONTROL: SCT1 Control Register .....	309
14.7.66	SCT2_CONTROL: SCT2 Control Register .....	310
14.7.67	LCT0_LOWER_CONTROL: LCT0 Lower Control Register.....	311
14.7.68	LCT0_UPPER_CONTROL: LCT0 Upper Control Register .....	311
14.7.69	LCT1_LOWER_CONTROL: LCT1 Lower Control Register.....	312
14.7.70	LCT1_UPPER_CONTROL: LCT1 Upper Control Register .....	312
14.7.71	SCT0_CURRENT_STATUS: SCT0 Current Status Register .....	313
14.7.72	SCT1_CURRENT_STATUS: SCT1 Current Status Register .....	313
14.7.73	SCT2_CURRENT_STATUS: SCT2 Current Status Register .....	314
14.7.74	LCT0_LOWER_CURRENT_STATUS: LCT0 Lower Current Status Register ...	314
14.7.75	LCT0_UPPER_CURRENT_STATUS: LCT0 Upper Current Status Register ...	314
14.7.76	LCT1_LOWER_CURRENT_STATUS: LCT1 Lower Current Status Register...	315
14.7.77	LCT1_UPPER_CURRENT_STATUS: LCT1 Upper Current Status Register ...	315
14.7.78	CTS_STATUS: CTS Status Register.....	316
14.7.79	CTS_CONTROL: CTS Control Register.....	317
14.7.80	LCT0_PEAK_TIMER_LOWER_STATUS: LCT0 Peak Timer Lower Status Register .....	318
14.7.81	LCT0_PEAK_TIMER_UPPER_STATUS: LCT0 Peak Timer Upper Status Register .....	319
14.7.82	LCT1_PEAK_TIMER_LOWER_STATUS: LCT1 Peak Timer Lower Status Register .....	319
14.7.83	LCT1_PEAK_TIMER_UPPER_STATUS: LCT1 Peak Timer Upper Status Register .....	320
14.7.84	PARAMETER_STATUS: Parameter Status Register .....	320
14.7.85	EXTENDED_CLAUSE_ACTION_CONTROL_C0S0: Extended Clause Action Control Registers C0S0.....	321
14.7.86	EXTENDED_CLAUSE_ACTION_CONTROL_C1S0: Extended Clause Action Control Registers C1S0.....	322
14.7.87	EXTENDED_CLAUSE_ACTION_CONTROL_C2S0: Extended Clause Action Control Registers C2S0.....	322
14.7.88	EXTENDED_CLAUSE_ACTION_CONTROL_C3S0: Extended Clause Action Control Registers C3S0.....	322
14.7.89	EXTENDED_CLAUSE_ACTION_CONTROL_C0S1: Extended Clause Action Control Registers C0S1.....	323
14.7.90	EXTENDED_CLAUSE_ACTION_CONTROL_C1S1: Extended Clause Action Control Registers C1S1.....	323
14.7.91	EXTENDED_CLAUSE_ACTION_CONTROL_C2S1: Extended Clause Action Control Registers C2S1.....	323
14.7.92	EXTENDED_CLAUSE_ACTION_CONTROL_C3S1: Extended Clause Action Control Registers C3S1.....	324
14.7.93	EXTENDED_CLAUSE_ACTION_CONTROL_C0S2: Extended Clause Action Control Registers C0S2.....	324
14.7.94	EXTENDED_CLAUSE_ACTION_CONTROL_C1S2: Extended Clause Action Control Registers C1S2.....	324
14.7.95	EXTENDED_CLAUSE_ACTION_CONTROL_C2S2: Extended Clause Action Control Registers C2S2.....	325
14.7.96	EXTENDED_CLAUSE_ACTION_CONTROL_C3S2: Extended Clause Action Control Registers C3S2.....	325
14.7.97	EXTENDED_CLAUSE_ACTION_CONTROL_C0S3: Extended Clause Action Control Registers C0S3.....	325



14.7.98	EXTENDED_CLAUSE_ACTION_CONTROL_C1S3: Extended Clause Action Control Registers C1S3.....	326
14.7.99	EXTENDED_CLAUSE_ACTION_CONTROL_C2S3: Extended Clause Action Control Registers C2S3.....	326
14.7.100	EXTENDED_CLAUSE_ACTION_CONTROL_C3S3: Extended Clause Action Control Registers C3S3.....	326
14.7.101	EXTENDED_CLAUSE_ACTION_CONTROL_C0S4: Extended Clause Action Control Registers C0S4.....	327
14.7.102	EXTENDED_CLAUSE_ACTION_CONTROL_C1S4: Extended Clause Action Control Registers C1S4.....	327
14.7.103	EXTENDED_CLAUSE_ACTION_CONTROL_C2S4: Extended Clause Action Control Registers C2S4.....	327
14.7.104	EXTENDED_CLAUSE_ACTION_CONTROL_C3S4: Extended Clause Action Control Registers C3S4.....	328
14.8	PCI Configuration Registers .....	329
14.8.1	PCI Configuration Registers Summary .....	329
14.8.2	VID: Device ID Register .....	330
14.8.3	VID: Vendor ID Register .....	330
14.8.4	CMD: Command Register .....	330
14.8.5	RID: Class Code Register .....	332
14.8.6	RID: Revision ID Register .....	332
14.8.7	HT: Header Type Register .....	332
14.8.8	MTB_LBAR: MSC Trace Buffer Lower BAR .....	332
14.8.9	MTB_UBAR: MSC Trace Buffer Upper BAR .....	333
14.8.10	SW_LBAR: Software Lower BAR .....	333
14.8.11	SW_UBAR: Software Upper BAR .....	334
14.8.12	RTIT_LBAR: RTIT Lower BAR .....	334
14.8.13	RTIT_UBAR: RTIT Upper BAR .....	335
14.8.14	SVID: Subsystem Vendor ID .....	335
14.8.15	CAP: Capabilities Pointer .....	335
14.8.16	INTL: Interrupt Line .....	335
14.8.17	MSICID: MSI Capability ID .....	336
14.8.18	MSILMA: MSI Lower Message Address .....	336
14.8.19	MSIUMA: MSI Upper Message Address .....	336
14.8.20	MSIMD: MSI Message Data .....	337
14.8.21	FW_LBAR: Firmware Lower Bar .....	337
14.8.22	FW_UBAR: Firmware Upper Bar .....	337
14.8.23	NPKDSC: NPK Device Specific Control .....	338
14.8.24	NPKDSD: NPK Device Specific Defeature .....	339
14.9	PTI Registers .....	340
14.10	Power Control Registers .....	340
14.10.1	Power Control Registers Summary .....	340
14.10.2	PMTCS: Power Mode Transition Control and status .....	340
14.10.3	PMTTL: Power Mode Transition Time Limit Register .....	340
14.11	SoCHAP Registers .....	341
14.11.1	SoCHAP Registers Summary .....	341
14.11.2	SDC: SoCHAP Device Capabilities .....	343
14.11.3	SSS: SoCHAP Software Register .....	344
14.11.4	CEC_0: Customizable Event Creation 0 .....	345
14.11.5	CEC_1: Customizable Event Creation 1 .....	347
14.11.6	CEC_2: Customizable Event Creation 2 .....	349
14.11.7	CEC_3: Customizable Event Creation 3 .....	351
14.11.8	CEC_4: Customizable Event Creation 4 .....	353
14.11.9	CEC_5: Customizable Event Creation 5 .....	355
14.11.10	CEC_6: Customizable Event Creation 6 .....	357
14.11.11	CEC_7: Customizable Event Creation 7 .....	359



14.11.12	CEC_8: Customizable Event Creation 8.....	361
14.11.13	CEC_9: Customizable Event Creation 9.....	363
14.11.14	CEC_10: Customizable Event Creation 10.....	365
14.11.15	CEC_11: Customizable Event Creation 11.....	367
14.11.16	CEC_12: Customizable Event Creation 12.....	369
14.11.17	CEC_13: Customizable Event Creation 13.....	371
14.11.18	CEC_14: Customizable Event Creation 14.....	373
14.11.19	CEC_15: Customizable Event Creation 15.....	375
14.11.20	SOCHAPCMD_0: SoCHAP Command Register 0.....	377
14.11.21	SOCHAPEV_0: SoCHAP Events Register 0.....	379
14.11.22	SOCHAPSTAT_0: SoCHAP Status Register 0.....	381
14.11.23	SOCHAPDATA_0: SoCHAP Data Register 0.....	382
14.11.24	SOCHAPCMD_1: SoCHAP Command Register 1.....	383
14.11.25	SOCHAPEV_1: SoCHAP Events Register 1.....	385
14.11.26	SOCHAPSTAT_1: SoCHAP Status Register 1.....	387
14.11.27	SOCHAPDATA_1: SoCHAP Data Register 1.....	388
14.11.28	SOCHAPCMD_2: SoCHAP Command Register 2.....	389
14.11.29	SOCHAPEV_2: SoCHAP Events Register 2.....	391
14.11.30	SOCHAPSTAT_2: SoCHAP Status Register 2.....	393
14.11.31	SOCHAPDATA_2: SoCHAP Data Register 2.....	394
14.11.32	SOCHAPCMD_3: SoCHAP Command Register 3.....	395
14.11.33	SOCHAPEV_3: SoCHAP Events Register 3.....	397
14.11.34	SOCHAPSTAT_3: SoCHAP Status Register 3.....	399
14.11.35	SOCHAPDATA_3: SoCHAP Data Register 3.....	400
14.11.36	SOCHAPCMD_4: SoCHAP Command Register 4.....	401
14.11.37	SOCHAPEV_4: SoCHAP Events Register 4.....	403
14.11.38	SOCHAPSTAT_4: SoCHAP Status Register 4.....	405
14.11.39	SOCHAPDATA_4: SoCHAP Data Register 4.....	406
14.11.40	SOCHAPCMD_5: SoCHAP Command Register 5.....	407
14.11.41	SOCHAPEV_5: SoCHAP Events Register 5.....	409
14.11.42	SOCHAPSTAT_5: SoCHAP Status Register 5.....	411
14.11.43	SOCHAPDATA_5: SoCHAP Data Register 5.....	412
14.11.44	SOCHAPCMD_6: SoCHAP Command Register 6.....	413
14.11.45	SOCHAPEV_6: SoCHAP Events Register 6.....	415
14.11.46	SOCHAPSTAT_6: SoCHAP Status Register 6.....	417
14.11.47	SOCHAPDATA_6: SoCHAP Data Register 6.....	418
14.11.48	SOCHAPCMD_7: SoCHAP Command Register 7.....	419
14.11.49	SOCHAPEV_7: SoCHAP Events Register 7.....	421
14.11.50	SOCHAPSTAT_7: SoCHAP Status Register 7.....	423
14.11.51	SOCHAPDATA_7: SoCHAP Data Register 7.....	424
14.11.52	SOCHAPCMD_8: SoCHAP Command Register 8.....	425
14.11.53	SOCHAPEV_8: SoCHAP Events Register 8.....	427
14.11.54	SOCHAPSTAT_8: SoCHAP Status Register 8.....	429
14.11.55	SOCHAPDATA_8: SoCHAP Data Register 8.....	430
14.11.56	SOCHAPCMD_9: SoCHAP Command Register 9.....	431
14.11.57	SOCHAPEV_9: SoCHAP Events Register 9.....	433
14.11.58	SOCHAPSTAT_9: SoCHAP Status Register 9.....	435
14.11.59	SOCHAPDATA_9: SoCHAP Data Register 9.....	436
14.11.60	SOCHAPCMD_10: SoCHAP Command Register 10.....	437
14.11.61	SOCHAPEV_10: SoCHAP Events Register 10.....	439
14.11.62	SOCHAPSTAT_10: SoCHAP Status Register 10.....	441
14.11.63	SOCHAPDATA_10: SoCHAP Data Register 10.....	442
14.11.64	SOCHAPCMD_11: SoCHAP Command Register 11.....	443
14.11.65	SOCHAPEV_11: SoCHAP Events Register 11 .....	445
14.11.66	SOCHAPSTAT_11: SoCHAP Status Register 11.....	447
14.11.67	SOCHAPDATA_11: SoCHAP Data Register 11 .....	448



14.11.68	SOCHAPCMD_12: SoCHAP Command Register 12 .....	449
14.11.69	SOCHAPEV_12: SoCHAP Events Register 12 .....	451
14.11.70	SOCHAPSTAT_12: SoCHAP Status Register 12 .....	453
14.11.71	SOCHAPDATA_12: SoCHAP Data Register 12 .....	454
14.11.72	SOCHAPCMD_13: SoCHAP Command Register 13 .....	455
14.11.73	SOCHAPEV_13: SoCHAP Events Register 13 .....	457
14.11.74	SOCHAPSTAT_13: SoCHAP Status Register 13 .....	459
14.11.75	SOCHAPDATA_13: SoCHAP Data Register 13 .....	460
14.11.76	SOCHAPCMD_14: SoCHAP Command Register 14 .....	461
14.11.77	SOCHAPEV_14: SoCHAP Events Register 14 .....	463
14.11.78	SOCHAPSTAT_14: SoCHAP Status Register 14 .....	465
14.11.79	SOCHAPDATA_14: SoCHAP Data Register 14 .....	466
14.11.80	SOCHAPCMD_15: SoCHAP Command Register 15 .....	467
14.11.81	SOCHAPEV_15: SoCHAP Events Register 15 .....	469
14.11.82	SOCHAPSTAT_15: SoCHAP Status Register 15 .....	471
14.11.83	SOCHAPDATA_15: SoCHAP Data Register 15 .....	472
14.11.84	SOCHAPPKTCTRL: SOCHAP Packet Control Register .....	473
14.11.85	SOCHAPPKTCAP: SOCHAP Packet Capabilities Register .....	473
14.12	ODLA Registers .....	474
14.12.1	ODLA Registers Summary .....	474
14.12.2	ODLACAP: ODLA Capability Register .....	475
14.12.3	OSMC: ODLA Storage Mode Control .....	476
14.12.4	TTSS: Timestamp Trigger Snap Shot .....	477
14.12.5	OSTAT: ODLA Status Register .....	478
14.12.6	ARBGNTCRED: Arbiter Grant Credit Register .....	479
14.12.7	TTSSEXT: Timestamp Trigger Snap Shot Extended Register .....	480
14.12.8	CTSS: Current Timestamp Snap Shot Register .....	480
14.12.9	CTSSEXT: Current Timestamp Snap Shot Extended Register .....	480
14.12.10	VCAPCTL: VISA Capture Control Register .....	480
14.12.11	VCAPREG_0: VISA Capture Register 0 .....	481
14.12.12	LNSTORECTRL_0: Lane Storage Control Register 0 .....	482
14.12.13	LNSTORECTRL_1: Lane Storage Control Register 1 .....	482
14.12.14	LNSTORECTRL_2: Lane Storage Control Register 2 .....	483
14.12.15	LNSTORECTRL_3: Lane Storage Control Register 3 .....	483
14.12.16	LNFTLCTL_0: Lane Filter Control Register 0 .....	484
14.12.17	LNFTLCTL_1: Lane Filter Control Register 1 .....	485
14.12.18	LNFTLCTL_2: Lane Filter Control Register 2 .....	486
14.12.19	LNFTLCTL_3: Lane Filter Control Register 3 .....	487
14.13	TAP Registers .....	488
14.13.1	TAP Registers Summary .....	488
14.13.2	SLVIDCODE: Slave TAP Identification Register .....	488
14.13.3	REQMSG: Request Message Register .....	488
14.13.4	RSPMSG: Response Message Register .....	489
14.13.5	CLKOVR: Clock Override Register .....	489
14.13.6	REQMSG: Request Message Register .....	489
14.13.7	RSPMSG: Response Message Register .....	490
14.13.8	CLKOVR: Clock Override Register .....	490
14.14	VIS Event Recognizer Registers .....	491
14.14.1	VIS Event Recognizer Registers Summary .....	491
14.14.2	VERCAP: VISA Event Recognizer Capabilities Register .....	492
14.14.3	VERCTL: VISA Event Recognizer Control Register .....	492
14.14.4	VERSTAT: VISA Event Recognizer Status Register .....	493
14.14.5	EVRCTL0: Event Recognizer Control Register 0 .....	493
14.14.6	EVRCTL1: Event Recognizer Control Register 1 .....	493
14.14.7	EVRCTL2: Event Recognizer Control Register 2 .....	494
14.14.8	EVRCTL3: Event Recognizer Control Register 3 .....	494



14.14.9	EVRCTL4: Event Recognizer Control Register 4 .....	494
14.14.10	EVRCTL5: Event Recognizer Control Register 5 .....	495
14.14.11	EVRCTL6: Event Recognizer Control Register 2 .....	495
14.14.12	EVRCTL7: Event Recognizer Control Register 7 .....	495
14.14.13	EVRMCH0: Event Recognizer Match Register 0 .....	496
14.14.14	EVRMCH1: Event Recognizer Match Register 1 .....	496
14.14.15	EVRMCH2: Event Recognizer Match Register 2 .....	496
14.14.16	EVRMCH3: Event Recognizer Match Register 3 .....	496
14.14.17	EVRMCH4: Event Recognizer Match Register 0 .....	497
14.14.18	EVRMCH5: Event Recognizer Match Register 1 .....	497
14.14.19	EVRMCH6: Event Recognizer Match Register 2 .....	497
14.14.20	EVRMCH7: Event Recognizer Match Register 3 .....	497
14.14.21	EVRMSK0: Event Recognizer Mask Register 0 .....	498
14.14.22	EVRMSK1: Event Recognizer Mask Register 1 .....	498
14.14.23	EVRMSK2: Event Recognizer Mask Register 2 .....	498
14.14.24	EVRMSK3: Event Recognizer Mask Register 3 .....	498
14.14.25	EVRMSK4: Event Recognizer Mask Register 0 .....	499
14.14.26	EVRMSK5: Event Recognizer Mask Register 1 .....	499
14.14.27	EVRMSK6: Event Recognizer Mask Register 2 .....	499
14.14.28	EVRMSK7: Event Recognizer Mask Register 3 .....	499
14.14.29	EVRFBMCH0: Event Recognizer Feedback Match Register 0 .....	500
14.14.30	EVRFBMCH1: Event Recognizer Feedback Match Register 1 .....	500
14.14.31	EVRFBMCH2: Event Recognizer Feedback Match Register 2 .....	500
14.14.32	EVRFBMCH3: Event Recognizer Feedback Match Register 3 .....	500
14.14.33	EVRFBMCH4: Event Recognizer Feedback Match Register 4 .....	501
14.14.34	EVRFBMCH5: Event Recognizer Feedback Match Register 5 .....	501
14.14.35	EVRFBMCH6: Event Recognizer Feedback Match Register 6 .....	501
14.14.36	EVRFBMCH7: Event Recognizer Feedback Match Register 7 .....	501
14.14.37	EVRFBMSK0: Event Recognizer Feedback Mask Register 0 .....	502
14.14.38	EVRFBMSK1: Event Recognizer Feedback Mask Register 1 .....	502
14.14.39	EVRFBMSK2: Event Recognizer Feedback Mask Register 2 .....	502
14.14.40	EVRFBMSK3: Event Recognizer Feedback Mask Register 3 .....	502
14.14.41	EVRFBMSK4: Event Recognizer Feedback Mask Register 0 .....	503
14.14.42	EVRFBMSK5: Event Recognizer Feedback Mask Register 1 .....	503
14.14.43	EVRFBMSK6: Event Recognizer Feedback Mask Register 2 .....	503
14.14.44	EVRFBMSK7: Event Recognizer Feedback Mask Register 3 .....	503
14.15	VIS Register Controller Registers .....	504
14.15.1	VIS Register Controller Registers Summary .....	504
14.15.2	VISACTLADDR: VISA2 Controller Address/Index Register .....	510
14.15.3	VISACTLDATA: VISA2 Controller Data Register .....	510
14.15.4	VISARPLYSEQ0: VISA2 Controller Replay Sequence Control Register 0 .....	510
14.15.5	VISACTLCAP: VISA2 Controller Capabilities Register .....	511
14.15.6	VISACTLCLM0: VISACTLCLM0 .....	511
14.15.7	VISACTLCLM1: VISACTLCLM1 .....	512
14.15.8	VISACTLCLM2: VISACTLCLM2 .....	512
14.15.9	VISACTLCLM3: VISACTLCLM3 .....	512
14.15.10	VISACTLCLM4: VISACTLCLM4 .....	512
14.15.11	VISACTLCLM5: VISACTLCLM5 .....	512
14.15.12	VISACTLCLM6: VISACTLCLM6 .....	513
14.15.13	VISACTLCLM7: VISACTLCLM7 .....	513
14.15.14	VISACTLCLM8: VISACTLCLM8 .....	513
14.15.15	VISACTLCLM9: VISACTLCLM9 .....	513
14.15.16	VISACTLULM0: VISACTLULM0 .....	513
14.15.17	VISACTLULM1: VISACTLULM1 .....	514
14.15.18	VISACTLULM2: VISACTLULM2 .....	514
14.15.19	VISACTLULM3: VISACTLULM3 .....	514



14.15.20	VISACTLULM4: VISACTLULM4.....	514
14.15.21	VISACTLULM5: VISACTLULM5.....	514
14.15.22	VISACTLULM6: VISACTLULM6.....	515
14.15.23	VISACTLULM7: VISACTLULM7.....	515
14.15.24	VISACTLULM8: VISACTLULM8.....	515
14.15.25	VISACTLULM9: VISACTLULM9.....	515
14.15.26	RRL_Data: VISA Replay RAM Lower.....	515
14.15.27	RSVD: VISA Replay RAM Upper .....	516
14.15.28	RRL_Data: VISA Replay RAM Lower.....	516
14.15.29	RSVD: VISA Replay RAM Upper .....	516
14.15.30	RRL_Data: VISA Replay RAM Lower.....	516
14.15.31	RSVD: VISA Replay RAM Upper .....	516
14.15.32	RRL_Data: VISA Replay RAM Lower.....	517
14.15.33	RSVD: VISA Replay RAM Upper .....	517
14.15.34	RRL_Data: VISA Replay RAM Lower.....	517
14.15.35	RSVD: VISA Replay RAM Upper .....	517
14.15.36	RRL_Data: VISA Replay RAM Lower.....	518
14.15.37	RSVD: VISA Replay RAM Upper .....	518
14.15.38	RRL_Data: VISA Replay RAM Lower.....	518
14.15.39	RSVD: VISA Replay RAM Upper .....	518
14.15.40	RRL_Data: VISA Replay RAM Lower.....	518
14.15.41	RSVD: VISA Replay RAM Upper .....	519
14.15.42	RRL_Data: VISA Replay RAM Lower.....	519
14.15.43	RSVD: VISA Replay RAM Upper .....	519
14.15.44	RRL_Data: VISA Replay RAM Lower.....	519
14.15.45	RSVD: VISA Replay RAM Upper .....	520
14.15.46	RRL_Data: VISA Replay RAM Lower.....	520
14.15.47	RSVD: VISA Replay RAM Upper .....	520
14.15.48	RRL_Data: VISA Replay RAM Lower.....	520
14.15.49	RSVD: VISA Replay RAM Upper .....	520
14.15.50	RRL_Data: VISA Replay RAM Lower.....	521
14.15.51	RSVD: VISA Replay RAM Upper .....	521
14.15.52	RRL_Data: VISA Replay RAM Lower.....	521
14.15.53	RSVD: VISA Replay RAM Upper .....	521
14.15.54	RRL_Data: VISA Replay RAM Lower.....	522
14.15.55	RSVD: VISA Replay RAM Upper .....	522
14.15.56	RRL_Data: VISA Replay RAM Lower.....	522
14.15.57	RSVD: VISA Replay RAM Upper .....	522
14.15.58	RRL_Data: VISA Replay RAM Lower.....	522
14.15.59	RSVD: VISA Replay RAM Upper .....	523
14.15.60	RRL_Data: VISA Replay RAM Lower.....	523
14.15.61	RSVD: VISA Replay RAM Upper .....	523
14.15.62	RRL_Data: VISA Replay RAM Lower.....	523
14.15.63	RSVD: VISA Replay RAM Upper .....	524
14.15.64	RRL_Data: VISA Replay RAM Lower.....	524
14.15.65	RSVD: VISA Replay RAM Upper .....	524
14.15.66	RRL_Data: VISA Replay RAM Lower.....	524
14.15.67	RSVD: VISA Replay RAM Upper .....	524
14.15.68	RRL_Data: VISA Replay RAM Lower.....	525
14.15.69	RSVD: VISA Replay RAM Upper .....	525
14.15.70	RRL_Data: VISA Replay RAM Lower.....	525
14.15.71	RSVD: VISA Replay RAM Upper .....	525
14.15.72	RRL_Data: VISA Replay RAM Lower.....	526
14.15.73	RSVD: VISA Replay RAM Upper .....	526
14.15.74	RRL_Data: VISA Replay RAM Lower.....	526
14.15.75	RSVD: VISA Replay RAM Upper .....	526



14.15.76	RRL_Data: VISA Replay RAM Lower.....	526
14.15.77	RSVD: VISA Replay RAM Upper.....	527
14.15.78	RRL_Data: VISA Replay RAM Lower.....	527
14.15.79	RSVD: VISA Replay RAM Upper.....	527
14.15.80	RRL_Data: VISA Replay RAM Lower.....	527
14.15.81	RSVD: VISA Replay RAM Upper.....	528
14.15.82	RRL_Data: VISA Replay RAM Lower.....	528
14.15.83	RSVD: VISA Replay RAM Upper.....	528
14.15.84	RRL_Data: VISA Replay RAM Lower.....	528
14.15.85	RSVD: VISA Replay RAM Upper.....	528
14.15.86	RRL_Data: VISA Replay RAM Lower.....	529
14.15.87	RSVD: VISA Replay RAM Upper.....	529
14.15.88	RRL_Data: VISA Replay RAM Lower.....	529
14.15.89	RSVD: VISA Replay RAM Upper.....	529
14.15.90	RRL_Data: VISA Replay RAM Lower.....	530
14.15.91	RSVD: VISA Replay RAM Upper.....	530
14.15.92	RRL_Data: VISA Replay RAM Lower.....	530
14.15.93	RSVD: VISA Replay RAM Upper.....	530
14.15.94	RRL_Data: VISA Replay RAM Lower.....	530
14.15.95	RSVD: VISA Replay RAM Upper.....	531
14.15.96	RRL_Data: VISA Replay RAM Lower.....	531
14.15.97	RSVD: VISA Replay RAM Upper.....	531
14.15.98	RRL_Data: VISA Replay RAM Lower.....	531
14.15.99	RSVD: VISA Replay RAM Upper.....	532
14.15.100	RRL_Data: VISA Replay RAM Lower.....	532
14.15.101	RSVD: VISA Replay RAM Upper.....	532
14.15.102	RRL_Data: VISA Replay RAM Lower.....	532
14.15.103	RSVD: VISA Replay RAM Upper.....	532
14.15.104	RRL_Data: VISA Replay RAM Lower.....	533
14.15.105	RSVD: VISA Replay RAM Upper.....	533
14.15.106	RRL_Data: VISA Replay RAM Lower.....	533
14.15.107	RSVD: VISA Replay RAM Upper.....	533
14.15.108	RRL_Data: VISA Replay RAM Lower.....	534
14.15.109	RSVD: VISA Replay RAM Upper.....	534
14.15.110	RRL_Data: VISA Replay RAM Lower.....	534
14.15.111	RSVD: VISA Replay RAM Upper.....	534
14.15.112	RRL_Data: VISA Replay RAM Lower.....	534
14.15.113	RSVD: VISA Replay RAM Upper.....	535
14.15.114	RRL_Data: VISA Replay RAM Lower.....	535
14.15.115	RSVD: VISA Replay RAM Upper.....	535
14.15.116	RRL_Data: VISA Replay RAM Lower.....	535
14.15.117	RSVD: VISA Replay RAM Upper.....	536
14.15.118	RRL_Data: VISA Replay RAM Lower.....	536
14.15.119	RSVD: VISA Replay RAM Upper.....	536
14.15.120	RRL_Data: VISA Replay RAM Lower.....	536
14.15.121	RSVD: VISA Replay RAM Upper.....	536
14.15.122	RRL_Data: VISA Replay RAM Lower.....	537
14.15.123	RSVD: VISA Replay RAM Upper.....	537
14.15.124	RRL_Data: VISA Replay RAM Lower.....	537
14.15.125	RSVD: VISA Replay RAM Upper.....	537
14.15.126	RRL_Data: VISA Replay RAM Lower.....	538
14.15.127	RSVD: VISA Replay RAM Upper.....	538
14.15.128	RRL_Data: VISA Replay RAM Lower.....	538
14.15.129	RSVD: VISA Replay RAM Upper.....	538
14.15.130	RRL_Data: VISA Replay RAM Lower.....	538
14.15.131	RSVD: VISA Replay RAM Upper .....	539



14.15.132 RRL_Data: VISA Replay RAM Lower.....	539
14.15.133 RSVD: VISA Replay RAM Upper.....	539
14.15.134 RRL_Data: VISA Replay RAM Lower.....	539
14.15.135 RSVD: VISA Replay RAM Upper.....	539
14.15.136 RRL_Data: VISA Replay RAM Lower.....	540
14.15.137 RSVD: VISA Replay RAM Upper.....	540
14.15.138 RRL_Data: VISA Replay RAM Lower.....	540
14.15.139 RSVD: VISA Replay RAM Upper.....	540
14.15.140 RRL_Data: VISA Replay RAM Lower.....	540
14.15.141 RSVD: VISA Replay RAM Upper.....	541
14.15.142 RRL_Data: VISA Replay RAM Lower.....	541
14.15.143 RSVD: VISA Replay RAM Upper.....	541
14.15.144 RRL_Data: VISA Replay RAM Lower.....	541
14.15.145 RSVD: VISA Replay RAM Upper.....	541
14.15.146 RRL_Data: VISA Replay RAM Lower.....	542
14.15.147 RSVD: VISA Replay RAM Upper.....	542
14.15.148 RRL_Data: VISA Replay RAM Lower.....	542
14.15.149 RSVD: VISA Replay RAM Upper.....	542
14.15.150 RRL_Data: VISA Replay RAM Lower.....	542
14.15.151 RSVD: VISA Replay RAM Upper.....	543
14.15.152 RRL_Data: VISA Replay RAM Lower.....	543
14.15.153 RSVD: VISA Replay RAM Upper.....	543
14.15.154 RRL_Data: VISA Replay RAM Lower.....	543
14.15.155 RSVD: VISA Replay RAM Upper.....	543
14.15.156 RRL_Data: VISA Replay RAM Lower.....	544
14.15.157 RSVD: VISA Replay RAM Upper.....	544
14.15.158 RRL_Data: VISA Replay RAM Lower.....	544
14.15.159 RSVD: VISA Replay RAM Upper.....	544
14.15.160 RRL_Data: VISA Replay RAM Lower.....	544
14.15.161 RSVD: VISA Replay RAM Upper.....	545
14.15.162 RRL_Data: VISA Replay RAM Lower.....	545
14.15.163 RSVD: VISA Replay RAM Upper.....	545
14.15.164 RRL_Data: VISA Replay RAM Lower.....	545
14.15.165 RSVD: VISA Replay RAM Upper.....	545
14.15.166 RRL_Data: VISA Replay RAM Lower.....	546
14.15.167 RSVD: VISA Replay RAM Upper.....	546
14.15.168 RRL_Data: VISA Replay RAM Lower.....	546
14.15.169 RSVD: VISA Replay RAM Upper.....	546
14.15.170 RRL_Data: VISA Replay RAM Lower.....	546
14.15.171 RSVD: VISA Replay RAM Upper.....	547
14.15.172 RRL_Data: VISA Replay RAM Lower.....	547
14.15.173 RSVD: VISA Replay RAM Upper.....	547
14.15.174 RRL_Data: VISA Replay RAM Lower.....	547
14.15.175 RSVD: VISA Replay RAM Upper.....	547
14.15.176 RRL_Data: VISA Replay RAM Lower.....	548
14.15.177 RSVD: VISA Replay RAM Upper.....	548
14.15.178 RRL_Data: VISA Replay RAM Lower.....	548
14.15.179 RSVD: VISA Replay RAM Upper.....	548
14.15.180 RRL_Data: VISA Replay RAM Lower.....	548
14.15.181 RSVD: VISA Replay RAM Upper.....	549
14.15.182 RRL_Data: VISA Replay RAM Lower.....	549
14.15.183 RSVD: VISA Replay RAM Upper.....	549
14.15.184 RRL_Data: VISA Replay RAM Lower.....	549



## Figures

Figure 1: Intel Trace Hub architecture high-level diagram.....	24
Figure 2: CSR_MTB_BAR MMIO Address Space Map .....	28
Figure 3: Intel Trace Hub Architecture .....	29
Figure 4: Intel Trace Hub/STH MMIO Regions .....	31
Figure 5: Software Trace Memory Region Memory Map .....	32
Figure 6: Software Master/Channel Memory Map Detail .....	33
Figure 7: Special Packet Generation CSRs Map .....	34
Figure 8: Intel PT Trace Memory Region Memory Map.....	37
Figure 9: STH Master Allocation.....	41
Figure 10: Intel Trace Hub Master Allocation.....	42
Figure 11: Master Allocation Example 1 .....	43
Figure 12: Master Allocation Example 2 .....	44
Figure 13: STH Lossy Mode Recovery FSM.....	45
Figure 14: Data Loss Special Packets Format .....	46
Figure 15: GTH in Intel Trace Hub .....	49
Figure 16: Global Trace Hub Architecture.....	51
Figure 17: Data Switch.....	54
Figure 18: Nibble order (1 of 5).....	56
Figure 19: Nibble order (2 of 5).....	57
Figure 20: Nibble order (3 of 5).....	58
Figure 21: Nibble order (4 of 5).....	59
Figure 22: Nibble order (5 of 5).....	60
Figure 23: Encoder Nibble Ordering in Memory.....	61
Figure 24: STPv2.1 USER packet format.....	61
Figure 25: Start of Data Loss Packet.....	62
Figure 26: End of Data Loss Packet.....	62
Figure 27: Trigger Unit.....	66
Figure 28: Trigger Unit Manual Override Logic .....	68
Figure 29: SoC timestamp correlation architecture – system view.....	70
Figure 30: Memory Storage Unit in Intel Trace Hub.....	72
Figure 31: MSC Single Block or CSR-Driven Mode .....	73
Figure 32: Single Memory Window with Multiple Blocks.....	75
Figure 33: Multiple Memory Windows.....	76
Figure 34: Memory Block Description Showing Header .....	77
Figure 35 : CTS Block Diagram.....	87
Figure 36: Textual Example of Sequencer Terms.....	88
Figure 37: Timing Diagrams Showing the Signal_Out Mode Behavior .....	93
Figure 38: Trigger Locations.....	99
Figure 39: Sequencer Block Diagram.....	101
Figure 40: Intel Trace Hub Architecture High-Level Diagram.....	102
Figure 41: PTI Packets (4-bit mode).....	107
Figure 42: PTI Packets (8-bit mode) .....	108
Figure 43: PTI Packets (16-bit mode) .....	108



## Tables

Table 3-1: Intel Trace Hub MMIO Offset/Size .....	27
Table 4-1: Software Re-Ordered Intel Processor Trace Header Definition.....	38
Table 4-2: STH Master Allocation Registers.....	41
Table 6-1: Standard Trace Source Assignments .....	52
Table 6-2: SMU Packet Data.....	64
Table 6-3: Store_Qual to Trace Source Mapping .....	66
Table 6-4: Trigger Unit Event Connections.....	68
Table 11-1: PTI Port Modes and Signal Mapping.....	105
Table 11-2: MIPI-PTI Training/Calibration Patterns .....	106
Table 12-1: Host Space Base Address Registers.....	111
Table 12-2: Root-Space 1 (RS1) Base Address Registers .....	111
Table 12-3: Unsupported Request Error Handling.....	112
Table 12-4: Completer Abort Error Handling .....	112
Table 12-5: Unsuccessful Completion Error Handling as a Requestor.....	113
Table 14-1: Register Attributes.....	121



# Revision History

Revision	Description	Date
1.0	Initial release.	January 2015
1.0.1	<ul style="list-style-type: none"><li>Section <a href="#">10.1</a>: Block diagram no longer shows External Mask and Match.</li><li>Section 14.2.53: Additions to the ScratchPad 0 register description.</li><li>Sections 14.2.55 and 14.2.56: Change to the bit description for ScratchPad 2 and ScratchPad 3 registers.</li><li>Section 14.4.2: TSCU Control Register updated.</li></ul>	September 2015
1.0.2	<ul style="list-style-type: none"><li>Section <a href="#">13.1.4</a> and <a href="#">13.1.5</a>: Added clarifying notes on the Trace to System Memory programming model.</li><li>Added SoCHAPPKTCAP register.</li><li>Removed IOSF Prim Agent Defeature register documentation.</li><li>Changes to registers:<ul style="list-style-type: none"><li>GTH_FREQ: default value changed from "varies" to 357.14 MHz</li><li>SCR.FCD: removed comment about forcing new window in MSU multi-block mode</li><li>STH.EVENT2DMSK: "event 2" added to description, clarifying that the EV2DMSK bits are for event 2</li><li>EV2CTRL: changed "event 1" to "event 2" in description</li><li>STH.EVENT3DMSK: "event 3" added to description, clarifying that the EV3DMSK bits are for event 3</li><li>EV3CTRL: changed "event 1" to "event 3" in description</li><li>SOCCHAPCMD_n: Select All Counters bit description updated for clarity</li><li>VID: value for DID field changed from fixed value to "varies"</li><li>SVID: Values for SID and SVID fields changed from fixed value to "varies".</li></ul></li></ul>	February 2016
2.1.0	<ul style="list-style-type: none"><li>Initial release of 2<sup>nd</sup> generation of Intel® Trace Hub Developer's Manual</li></ul>	September 2016
2.1.1	<ul style="list-style-type: none"><li>Added missing registers: STH, SoCHAPDATA_9, SoCHAPPKTCAP, PMTTL, PMTCS.</li></ul>	December 2016
2.1.2	<ul style="list-style-type: none"><li>Added bit-level descriptions of IAFW_CTRL field in SCRPD2 register.</li></ul>	October 2017

# 1 Introduction

The Intel® Trace Hub (Intel® TH) hardware is a set of functional blocks with the ability to perform full system debugging. That is, the Intel Trace Hub is not intended for hardware only or software only, but for full *system* debug. Specifically, it allows for testing the interaction of hardware and software as they produce complex system behaviors. Because the Intel Trace Hub is intended for use by third-party debug tools, it is purposely designed and aligned with industry-standard debug methods and tools.

To accomplish these goals, the Intel Trace Hub defines new features and leverages existing debug technologies to provide a complete framework for hardware/software co-debug, software development and tuning, and overall system performance optimization. Moreover, the Intel Trace Hub ensures a consistent programming model by working as a Peripheral Component Interconnect (PCI) device that is visible to software.

The Intel Trace Hub is composed of trace sources, a global hub, trace destinations, and a trigger unit. Trace sources include internal hardware signals from the Visualization of Internal Signals Architecture, performance data from the SoC Hardware and Performance counters, and software/firmware trace and debug data from the Software Trace Hub. Trace destinations include system memory, a MIPI® PTI port, and USB. The Global Trace Hub routes the data from the trace sources to the trace destinations according to the user's configuration (via software), and the Trigger Unit controls the starting and stopping of tracing operations. Finally, the Time Stamp Correlation Unit provides a timestamp to the trace sources, and also allows correlating of time-stamped data from the CPU's time domain to the Intel Trace Hub time domain.

## 1.1 Terminology

Term	Description
ART	Always-Running Timer
BPB	Byte-Packing-Buffer
Channel	STPv2 Channel: The lower-level data source identifier in a hierarchical STP data stream. Each Master has an independent Channel space.
CSR	Registers in a functional block that control an operation, or report the status of an operation. This single term is used to refer to both control registers and status registers.
CTC	Common Timer Copy
CTS	Common Trigger Sequencer
DbC or DBC	The host mode of an xHCI controller.
DCI	Intel® Direct Connect Interface (Intel® DCI)
DbC.Trace	The endpoint within the xHCI controller that transports trace data directly from the Intel Trace Hub to the USB port/connector
FIFO	First-In First-Out
GTH	Global Trace Hub

Term	Description
HAS	High-Level Architecture Specification
IA	Intel® architecture
IO	Input/Output
IP	Intellectual Property
Master	STPv2 Master: The higher-level data source identifier in a hierarchical STP data stream.
MMI	MSC Memory Interface
MMIO	Memory Mapped IO
MSC	Memory Storage Controller
MSI	Message Signaled Interrupt
MSU	Memory Storage Unit
MTB	MSC Trace Buffer
ODLA	On-Die Logic Analyzer
OS	Operating System
PCI	Peripheral Component Interconnect
PHY	Physical Layer
PTI	Parallel Trace Interface
RTIT	Real Time Instruction Tracing. An older version of Intel® Processor Trace.
SoC	System-on-Chip
SoCHAP	System-on-Chip Performance Counters
SSC	Spread-Spectrum-Clocking
STH	Software Trace Hub
STP	MIPI® System Trace Protocol
STPv2	MIPI System Trace Protocol, version 2
TSC	Timestamp Counter
TSCU	Time Stamp Counter Unit
UC	Uncacheable: A type of memory that cannot be stored in a CPUs cache. UC memory is strongly ordered.
USWC	Uncacheable Speculative Write Combining: A type of memory that allows for multiple individual writes to be combined into a single burst write. USWC memory is weakly ordered.
VER	VIS Event Recognition
VIS	Visualization of Internal Signals



## 2 Feature Set

This chapter includes a brief list of the requirements and features of the Intel Trace Hub architecture host interface and shared blocks.

### 2.1 Features

#### 2.1.1 Software Trace Hub Features

- **Support for industry standard MIPI STPv2.1:** Support for latest industry standard tracing protocol developed by the MIPI\* Alliance and known as STPv2.1 (System Trace Protocol version 2.1) with up  $2^{16}$  Masters and up to  $2^8$  Channels per Master.
- **Software/firmware tracing through primary and sideband fabric**
- **Intel® Processor Trace (Intel® PT) support through primary fabric with bandwidth up to 2GB/s:** Support for Intel PT through primary fabric with bandwidth up to 2GB/s<sup>1</sup>.
- **Uncacheable Speculative Write Combining (USWC) MMIO region for PT trace data:** Offers higher performance than Un-Cacheable (UC) memory, maintaining IA core performance for Intel PT.
- **Un-Cacheable (UC) MMIO region for general-purpose software trace data:** Guarantees the strongly-ordered memory model for software/firmware tracing and debug.
- **Time stamping Support:** Supports time-stamping of incoming debug data via the Time Stamp Counter Unit (TSCU), another component of the Intel Trace Hub architecture.
- **System Deadlock avoidance function:** STH supports a deadlock avoidance counter to eliminate input-to-out dependencies that could result in a deadlock on the primary fabric.
- **Supports up to 4 Match/Mask Events:** Built-in support for simultaneously detecting up to 4 configurable software events.

#### 2.1.2 Intel® Trace Hub PTI Port Features

- Compliant with MIPI-PTIv1
- Configurable width output data of 2 bits, 4 bits, 8 bits, and 16 bits.
- No explicit constraints on the trace clock speed as long as the required setup and hold times are met (see MIPI-PTIv1 specification).

### 2.2 Trace Sources

- Software/firmware tracing through the primary and sideband fabric with software instrumentation.
- Intel Processor Trace support through the primary fabric.

---

<sup>1</sup> Processor data rates are estimated at 1-2 bits per clock per Core.  $2b/clock/core * 4 cores * 2 GHz = 16 Gb/s = 2 GB/s$

- Hardware event capturing through ODLA and the VIS network.
- Hardware event performance monitoring through SoCHAP and the VIS network.
- Support for non-VIS based hardware tracing including fabric trace hooks.

## 2.3 Trace Destinations

- Support on-chip trace collection to internal buffer or system memory.
- Support off-chip trace collection to capture hardware using the MIPI-PTIv1 and MIPI-HTI port.
- Support off-chip trace collection through USB3 via the Direct Connect Interface (DCI) technology with two supported modes (BSSB push mode and USB3 pull mode).

## 2.4 Primary Fabric Interface Features

- Provide PCI device presence
- Accessible from multiple root spaces.
- Support for single Message Signaled Interrupt (MSI).
- Implements multiple levels of error handling.

## 2.5 Sideband Fabric Interface Features

- Support for a single outstanding transaction at any given time.
- Provides alternate downstream access to memory mapped regions and configuration space registers.
- Accessible from multiple root spaces.
- Support for legacy INTx interrupt generation.

## 2.6 TAP Interface Features

- Provides alternate access method to all Intel Trace Hub components including memory mapped CSRs and configuration space registers, as well as to embedded memory arrays for debug.
- Support for running VIS Controller from TAP clock if other clocks are still down.

## 2.7 Consistent Programming Model

- Seen by software running on the main IA CPU as a PCI device.
- Support for three 64-bit PCI Base Address Registers (BARs).
- Support for an additional 64-bit programmable BAR for firmware sources.

## 2.8 Error Handling and Survivability

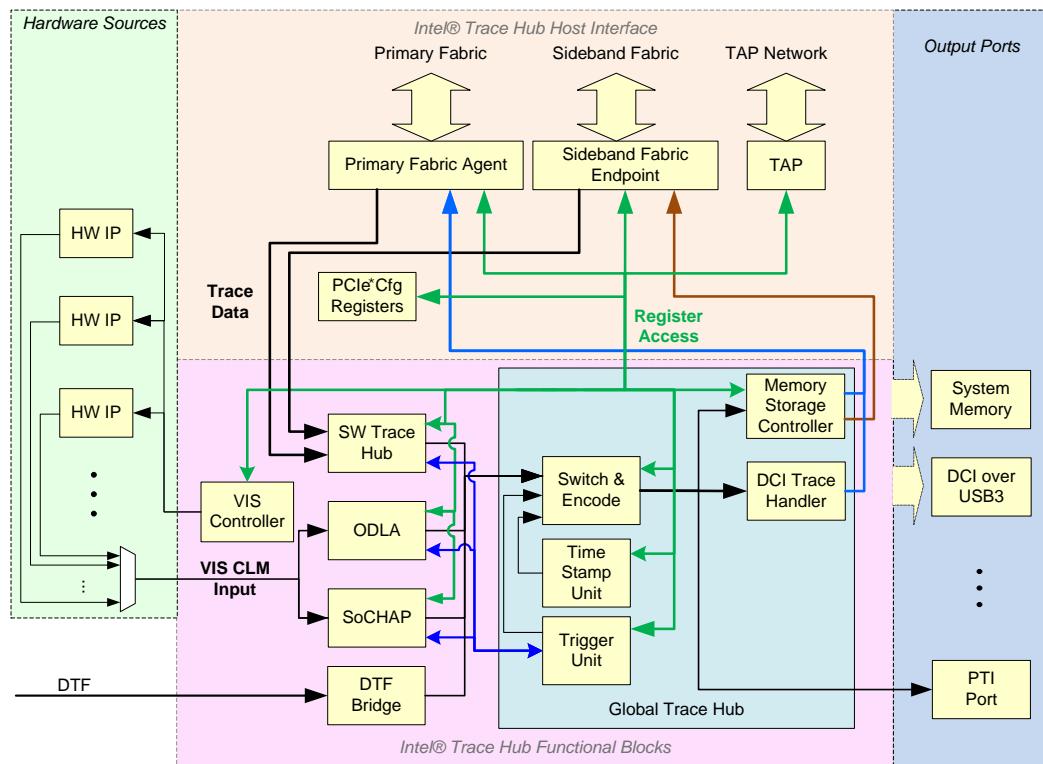
- Support for primary fabric and PCI standard error handling and logging.
- Support for software controlled functional reset.

## 3 Functional Description

### 3.1 Overview

The Intel Trace Hub comprises the blocks shown in Figure 1. Two other blocks not part of the Intel Trace Hub IP shown in Figure 1 include the hardware blocks/units feeding *trace data*<sup>2</sup> into the Intel Trace Hub through the Visualization of Internal Signals (VIS) network and other hardware trace data interfaces, hereafter simply referred to as *non-VIS trace sources*, and the group of output ports, which can include one or more destinations for the trace data. System memory is the only required destination while others are optional.

**Figure 1: Intel Trace Hub architecture high-level diagram**



As shown in Figure 1, trace data is sourced from one or more sources. These sources include hardware events routed through the VIS network or other non-VIS trace sources. Trace data can also be software driven and delivered through one of the fabric interfaces from any agent that can master one or both of the interfaces, and driven into the STH. Hardware-based trace data routed through the VIS network is driven into the ODLA, VER, and the SoCHAP blocks for processing. The output of these blocks is then driven into the GTH which encodes the data into MIPI STPv2.1 and routes to one of the output trace destinations. One trace destination required for all Intel Trace Hub implementations is

<sup>2</sup> In the context of the Intel Trace Hub architecture, trace data and debug data are used interchangeable to refer to the data collected from various hardware and software sources that enable an accurate reconstruction of system behavior for debug and performance optimization purposes.

system memory, which is handled by the Memory Storage Unit (MSU) through the primary fabric. Another trace destination is the USB3 Debug Controller (DbC), which can stream data off-chip (through the USB3 port) and is managed by the Intel Trace Hub DCI Handler (ITH DCIH) and transferred through the primary and sideband fabric interfaces with maximum bandwidth of 450MB/s. Finally, other trace destinations can be instantiated to provide a direct off-chip path for trace data capturing using internal/external tools. Those destinations include the PTI port with a bandwidth up to 800MB/s (16 single ended lanes double pumped at 200MHz), and HTI port with a bandwidth up to 100Gb/s (8 lanes at 12.5Gb/s).

The Intel Trace Hub host interface supports a primary fabric interface, and a sideband fabric interface. It also supports a Test Access Port (TAP) for JTAG-based access to the Intel Trace Hub components. In addition, the Intel Trace Hub host interface<sup>3</sup> contains some other units that are shared (or common) between all the functional blocks.

### 3.1.1 Functional Blocks

Referring to Figure 1, the Intel Trace Hub functional blocks include the Software Trace Hub (STH) block, the On-Die Logic Analyzer (ODLA)/VIS Event Recognition (VER) block, the System-on-Chip Performance Counters (SoCHAP) block, the Visualization of Internal Signals (VIS) Controller block, the DTF Bridge, and the Global Trace Hub (GTH) block.

The GTH, being at the heart of the Intel Trace Hub functional blocks, is comprised of multiple sub-blocks including the Time Stamp Counter Unit (TSCU), the Intel Trace Hub DCI Trace Handler block, the Memory Storage Unit (MSU) block, and the Common Trigger Sequencer (CTS)<sup>4</sup>. A brief overview of these functional blocks is presented in this section, with detailed descriptions being presented in each block's respective chapter.

The Visualization of Internal Signals (VIS) Controller block is responsible for programming/configuring the VIS network using a simple serial interface to route the various internal signals. Those signals can then be fed into the ODLA, VER, and the SoCHAP blocks through the Central Level Mux (CLM), for hardware debug and performance monitoring.

The On-Die Logic Analyzer (ODLA), when combined with VIS Event Recognition (VER) logic, is designed to capture hardware events using the Common Trigger Sequencer (CTS) for user-generated triggers. On the other hand, the SoCHAP architecture is designed to periodically sample certain hardware events and to capture useful information about system behavior and performance<sup>5</sup>. Those blocks are trace data sources to the GTH which collect data and encode it into the industry standard MIPI STPv2.1 (System Trace Protocol version 2)<sup>6</sup> before routing to one of the destination ports.

The STH is another source of trace data fed into the GTH. It is the means by which a device, often a CPU or micro-controller, can send debug data over the system fabric to the GTH, where it is collected, time stamped via the TSCU, encoded, and later sent to a trace destination (such as system memory, or a PTI port). Intel PT data can be sent to the STH

---

<sup>3</sup> Some design documentation will refer to the Intel Trace Hub Host Interface as the Common Blocks Cluster since it contains all the blocks shared between the different functional units.

<sup>4</sup> The Common Trigger Sequencer (CTS) is a component of the Trigger Unit (TU).

<sup>5</sup> The ODLA, the VIS Event Recognition, and SoCHAP architectures have been part of the debug architecture known as Lakemore. However, they have undergone multiple changes to fit into the Intel Trace Hub architecture framework.

<sup>6</sup> The MIPI STPv2 is the industry standard tracing protocol developed by the MIPI Alliance.



when Intel PT is configured to specifically direct its output to Intel Trace Hub MMIO space. (See Chapter 4 for more information.) STH can also include regular (non-Intel PT) software trace data, such as the message from a Linux kernel printk instruction or application printf instructions. All data destined for the STH can be sent over the primary fabric or the sideband fabric, according to bandwidth and the availability of the resources. The STH also provides software event triggering capabilities that are controlled by the CTS.

The DTF Bridge (NDB) provides the "trace source" functionality required by the Intel Trace Hub architecture, for trace sources that send their trace data via the DTF (Debug Trace Fabric) bus/protocol.

The Global Trace Hub (GTH) is responsible for collecting the data from the various sources mentioned above. Trigger sequences which enable and disable trace data collection and storage are generated in the GTH Trigger Unit. Once the GTH receives trace data, it is encoded into MIPI STPv2.1 protocol, and sent to a trace destination.

### 3.1.2 Output Ports

The Intel Trace Hub architecture is designed to support multiple trace destinations for the collected trace data. The only mandatory destination is system memory, which is reachable through the primary fabric interface. Other optional output ports include the USB3 port interface, the High Speed Trace Interface (HTI) and the Parallel Trace Interface (PTI) port.

The Intel Trace Hub DCI Trace Handler is the means by which trace data can be sent to the USB3 DbC and later off-chip through a USB3 port. The Intel Trace Hub DCI Handler is responsible for providing all the services needed, including signaling the availability of trace data in the MSC Trace Buffer (MTB) to the USB3 DbC, and forwarding the trace data when requested by the USB3 DbC.

The MIPI HTI is a standard high-speed trace interface, defined by the MIPI Alliance, which allows for the use, or re-use, of a standard high-speed external port such as Display Port or HDMI, to be used for sending out trace data. For detailed information about the MIPI HTI port, please refer to the MIPI HTI specification, and also your product's documentation.

The MIPI Parallel Trace Interface (PTI) port is another standard interface defined by the MIPI Alliance which is composed of a double pumped parallel output port along with a strobe that is used primarily in mobile market segment to send trace data off-chip. For detailed information about interfacing with the PTI port, see Chapter11 .

## 3.2 MTB/CSR Memory Map

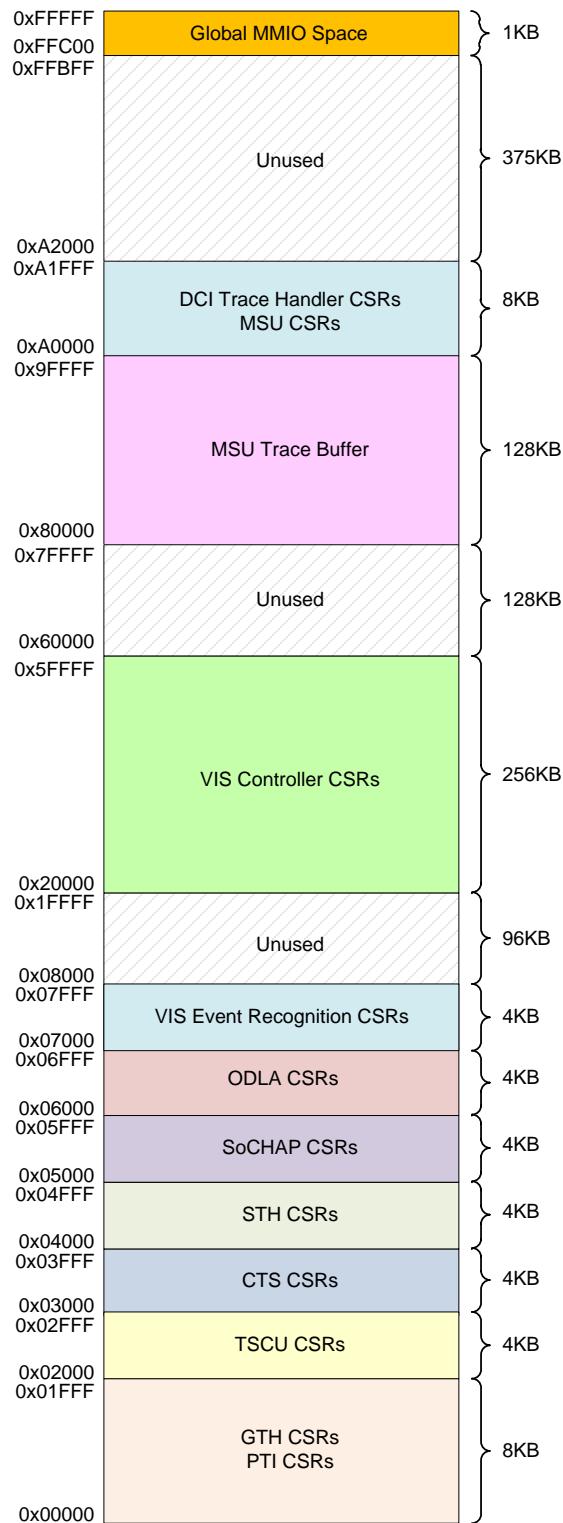
The CSR\_MTB\_BAR maps the configuration registers of the Intel Trace Hub and the MTB to 1MB of MMIO space. The *local* base address (which could be considered an offset into the CSR\_MTB\_BAR) values for each base address of the Intel Trace Hub's functional blocks is given in Table 3-1 and a visual presentation is shown in Figure 2.



Table 3-1: Intel Trace Hub MMIO Offset/Size

Functional Block	Symbol	BASE_ADDR	Size
<b>Global Trace Hub CSR PTI Port CSRs</b>	GTH_CSR_OFFSET	0x00000	8KB
<b>Time Stamp Counter CSRs</b>	TSCU_CSR_OFFSET	0x02000	4KB
<b>Comm. Trigger Sequencer CSRs</b>	CTS_CSR_OFFSET	0x03000	4KB
<b>Software Trace Hub CSRs</b>	STH_CSR_OFFSET	0x04000	4KB
<b>SoC CHAP Counters CSRs</b>	CHAP_CSR_OFFSET	0x05000	4KB
<b>On-Die Logic Analyzer</b>	ODLA_CSR_OFFSET	0x06000	4KB
<b>VIS Event Recognition CSR</b>	VER_CSR_OFFSET	0x07000	4KB
<b>VIS Register Controller CSRs</b>	VRC_CSR_OFFSET	0x20000	256KB
<b>MSC Trace Buffer MMIO MTB1 -&gt; MTB_LOCAL_OFFSET MTB2 -&gt; MTB1 + 0x8000</b>	MTB_LOCAL_OFFSET	0x80000	128KB
<b>Memory Storage Unit CSRs DCI Trace Handler CSRs</b>	MSU_CSR_OFFSET	0xA0000	8KB

**Figure 2: CSR\_MTB\_BAR MMIO Address Space Map**

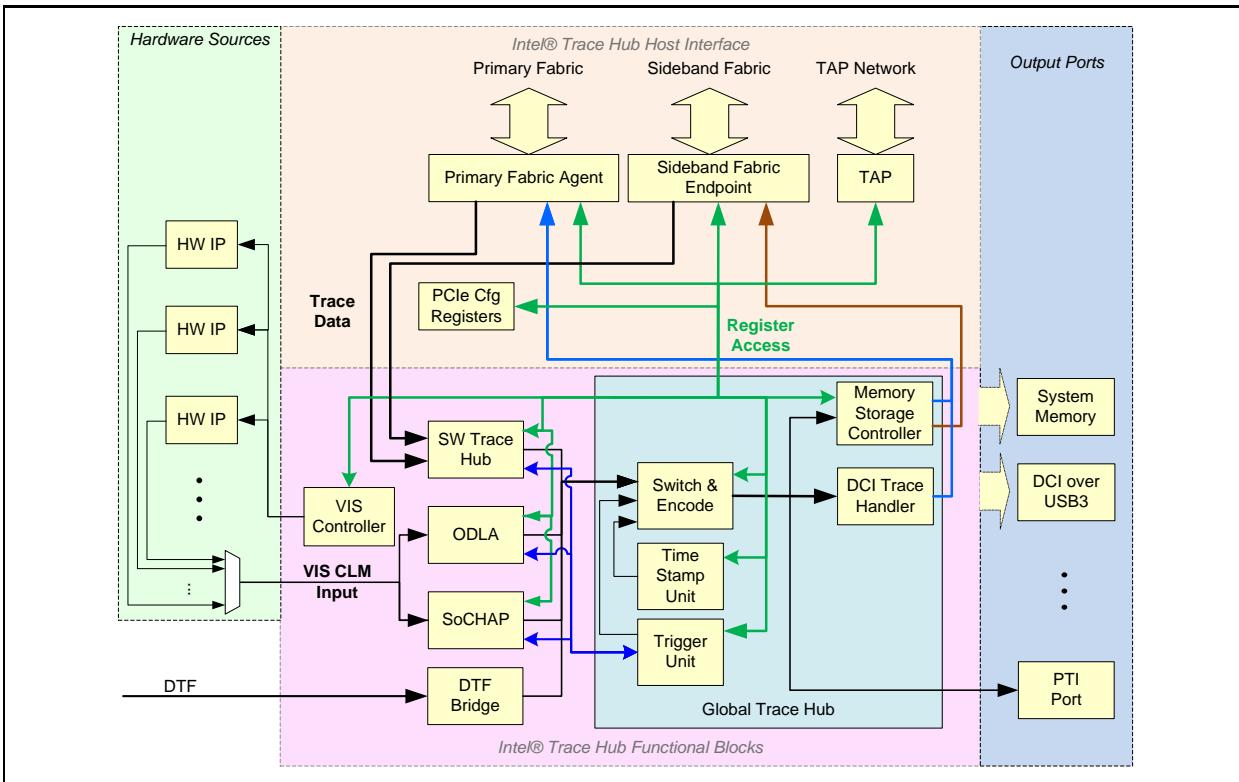


## 4 Software Trace Hub

The Software Trace Hub (STH) is one of the trace data sources feeding data into the Global Trace Hub (GTH) component of the Intel Trace Hub architecture, as shown in Figure 3. It is the means by which a device, often a CPU or micro-controller, can send debug messages over the system fabric to the GTH to be collected, encoded, time stamped, and later sent to a trace destination. Trace destinations can be either off-chip (for example, a high-speed trace port) or on-chip (for example, system memory).

Debug messages can include Intel Processor Trace (Intel PT) data, which can be sent to the STH when Intel PT hardware in the CPU is configured to direct its output to Intel Trace Hub MMIO space as explained elsewhere in this document. It can also include regular (non-PT) software and firmware trace data, such as the message from a Linux kernel `printf` instruction or application `printf` instructions. Hereafter, non-PT data will be referred to simply as *software trace data* or *regular software data*. Software messages are not limited to software running on the CPU core(s); they can also be sent from other microcontrollers running firmware, so long as they can master one of the interfaces to the STH (primary or sideband fabric).

**Figure 3: Intel Trace Hub Architecture**



The STH is accessible through both primary fabric and the sideband fabric interfaces, and is mapped to multiple spaces. However, despite being visible and accessible through different interface/spaces, the STH presents a single memory map, identical to all interfaces. This allows full access to all STH capabilities independent of the source or path the transaction took to reach the STH MMIO space. The STH is capable of high data transfer rates (up to



2GB/s<sup>7</sup>) to support up to four PT data streams (from four CPUs at up to 2GHz<sup>8</sup>), plus software trace data.

The STH is seen by software as three independent memory-mapped regions configured through the Intel Trace Hub PCI device Base Address Registers (BARs). The first region is used by the STH Control and Status Registers (CSRs) and includes some global packet generation commands as explained later in this document. The two other memory regions (or targets) are used for software trace data, and for Intel PT data, respectively. The Software Trace Memory Region (STMR) receives software trace data, which is then mapped to an internal protocol that can later be encoded by the GTH into STPv2.1 packets.

One important note is that, while the MIPI STPv2 specification supports data packets of widths 4, 8, 16, 32, and 64 bits, the Intel Trace Hub STH only supports widths of 8, 16, 32, and 64 bits (that is, it does not support 4-bit data packets). Further, to keep the logic simple, the STH will automatically round-up any data received that is not of the above widths to the next STPv2 supported size (for example, a 48-bit write will automatically be rounded up into a 64-bit dataset).

On the other hand, Intel PT data is claimed by the Intel PT Trace Memory Region (RTMR) controller (also referred to as the Intel PT Trace data target), and later forwarded to the GTH input buffer for encoding and transmission.

## 4.1 STH MMIO Regions

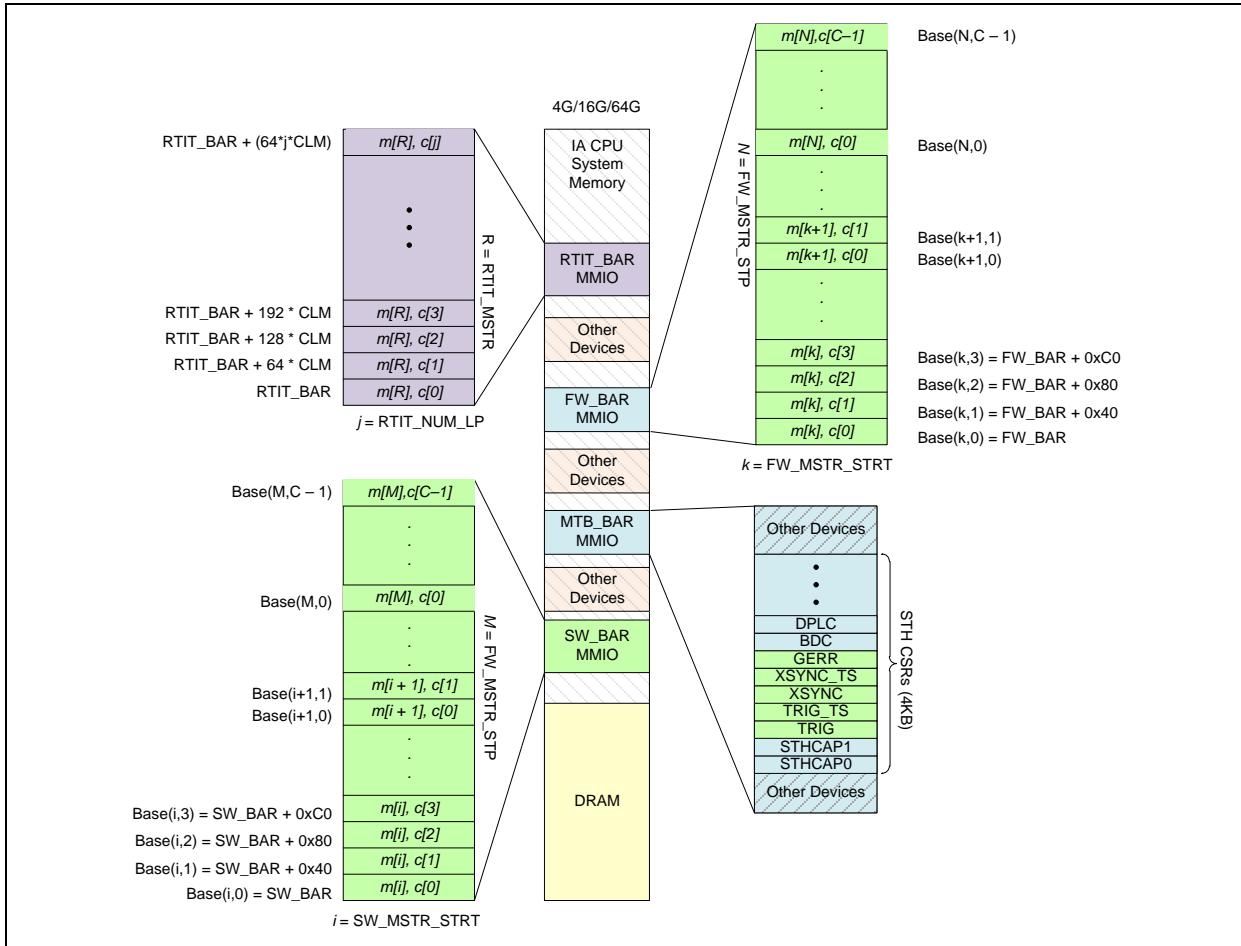
The STH appears to software in three MMIO regions in system memory. The STH has two dedicated MMIO regions while its configuration registers, including a few special registers, share a third MMIO region with the rest of the Intel Trace Hub components. The architecture of those regions is the topic of this section with an illustrative diagram of how the STH MMIO regions appear in system memory given in Figure 4.

---

<sup>7</sup> When operating at 250MHz or higher with a bus width of 64bits

<sup>8</sup> Intel Processor Trace data rates are estimated at 1-2 bits per clock per Core.  $2b/clock/core * 4 cores * 2GHz = 16Gb/s = 2GB/s$

Figure 4: Intel Trace Hub/STH MMIO Regions



#### 4.1.1 Software Trace Memory Region (STMR)

##### 4.1.1.1 General

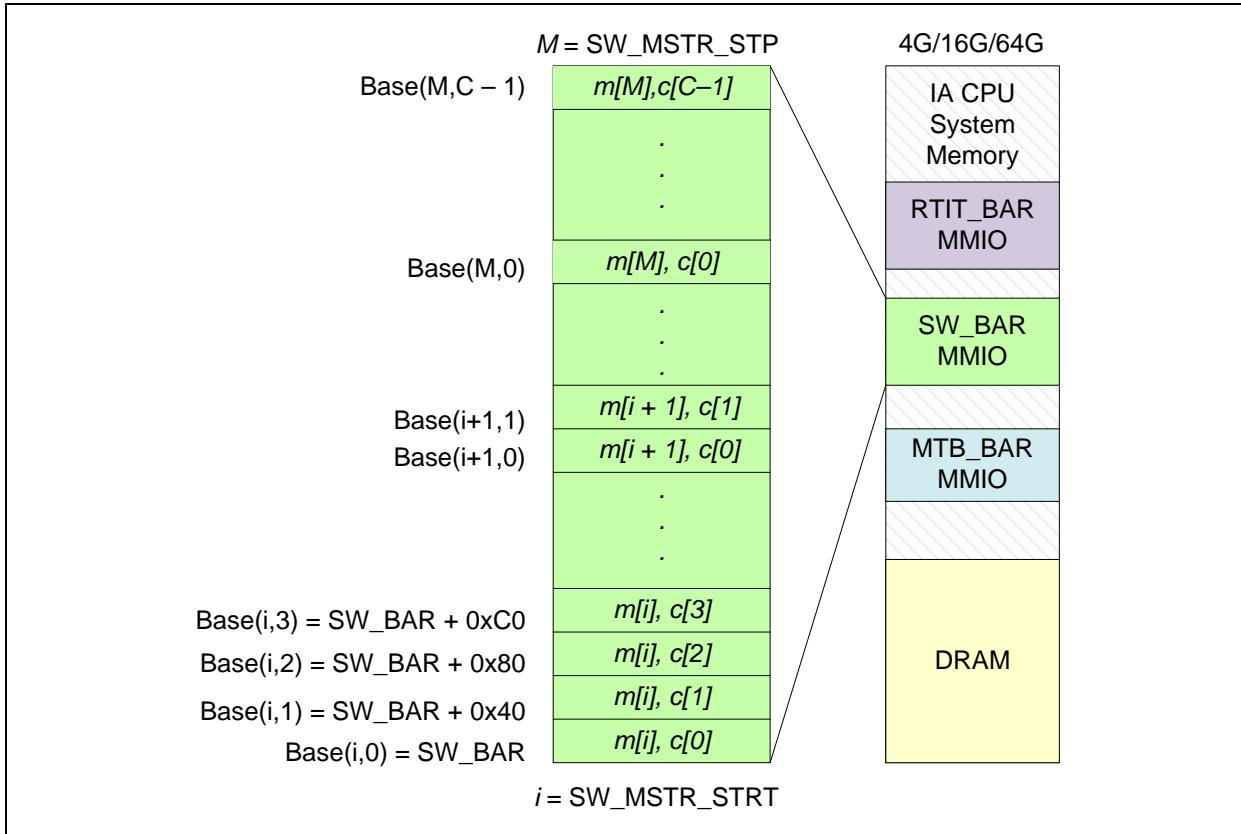
For software trace data, the Software Trace Memory Region (STMR) appears in system memory as determined by the SW\_BAR PCI BAR (which is a concatenation of SW\_UBAR and SW\_LBAR). One important requirement for the STMR region is the necessity of mapping it to Un-Cacheable (UC) memory space because it is designed only for a strongly-ordered memory model.

Software can determine the number of masters in the STMR region using the STHCAP0 register. Assuming STHMNUM – STHMSTR number of masters will be allocated for software with SW\_CHCNT channel per master, then, with 64B for each channel, the total size of STMR, in bytes, is given by the following formula:

$$Size(STMR) = 2^{\log_2 \lceil 64 \times (SW\_MSTR\_STP - SW\_MSTR\_STRT) \times SW\_CHCNT \rceil}$$

For example, an STMR implementation with 64 Masters and 128 Channels per Master would occupy  $64 \times 64 \times 128 = 512\text{KB}$  of MMIO space.

Figure 5: Software Trace Memory Region Memory Map



As shown in Figure 5, the address to which (debug) data is written determines the Master and Channel assigned to the data. For example, if software writes to address SW\_BAR, then it will be assigned to the first software Master number, denoted by the SW\_MSTR\_STRT parameter. The base address for master  $m$ , channel  $c$ , is given by the following formula:

$$Base(m - SW\_MSTR\_STRT, c) = 0x40 \times (SW\_CHCNT \times (m - SW\_MSTR\_STRT) + c) + SW\_BAR$$

Thus, given the following conditions:

- STH supports
  - 512 Masters
  - 16 Channels per Master
  - Base address of 0x3800\_0000
  - Starting master number of 16
- Master 34
- Channel 3

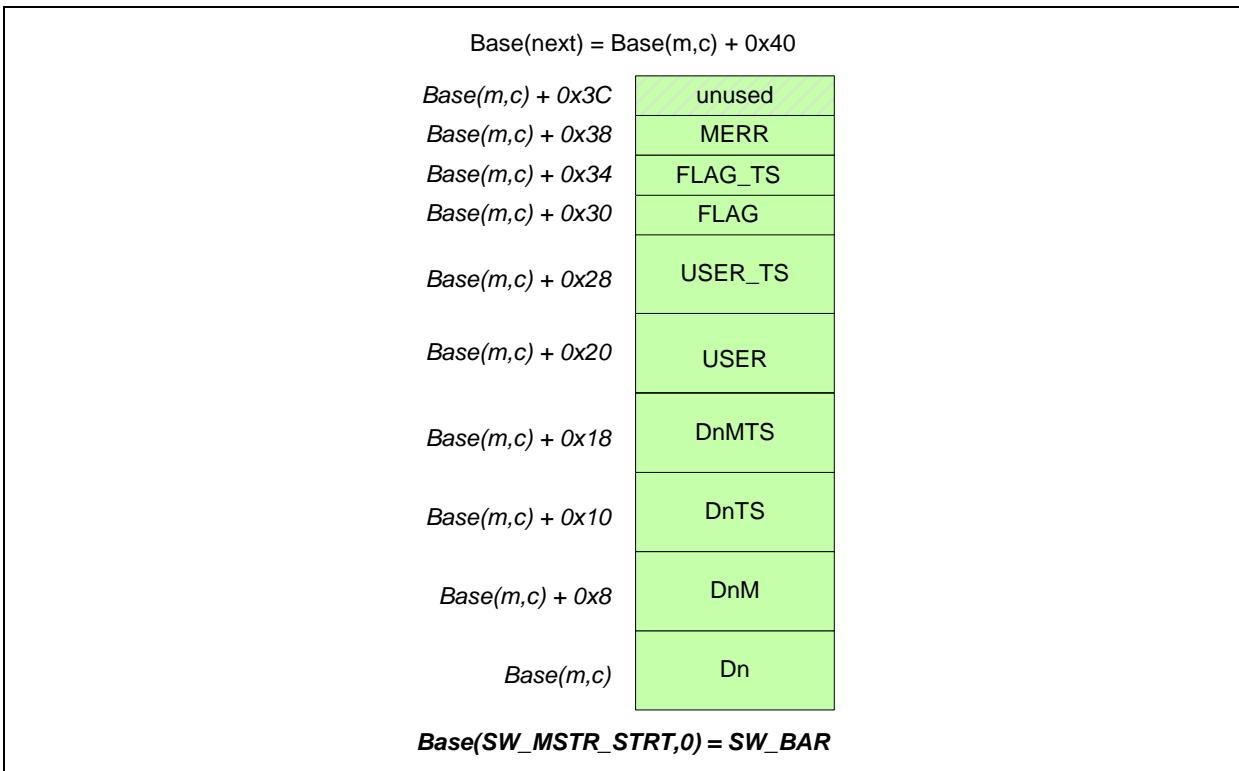
Then the base address for Master 34, Channel 3 is 0x380048C0.

As another example, if each software master is allocated 128 channels, and if software wishes to write some debug message to Master SW\_MSTR\_STRT + 4, Channel 1, then it needs to issue a write request to the base address SW\_BAR + 0x8040.

Within each Master/Channel MMIO region, the memory is presented as 64-bit wide because this is the maximum packet size that the MIPI STPv2.1 standard supports. Because there are a number of packet types that the STH can generate, and because software desires to control this specifically, a total of 64B is allotted for each Master/Channel. Each 64B region

is divided into several areas, according to the type of MIPI STPv2 packet that will be generated when software writes to that address. The various packets and their corresponding addresses within each Master/Channel region are shown in Figure 6.

**Figure 6: Software Master/Channel Memory Map Detail**

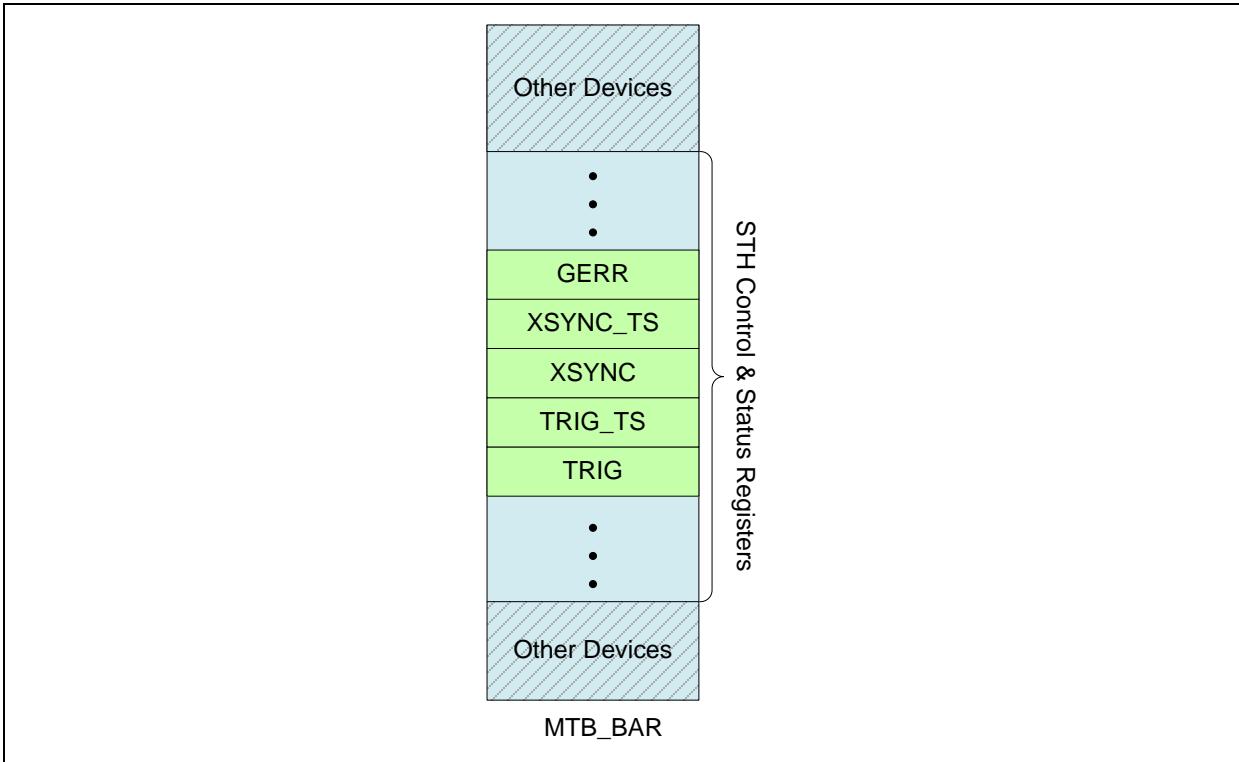


At the bottom of the region are the data packets: regular data, marked, time-stamped, and marked plus time-stamped. These are denoted in Figure 6 as Dn, DnM, DnTS, and DnMTS, respectively. The other packet types are above the data packet types: USER, USER\_TS, FLAG, FLAG\_TS, and MERR. Note that the MERR packet is a per-Master packet, not a per-Channel packet. That is, a MERR packet written to Master 0, Channel 0 will appear identical to a MERR packet written to any other Channel in that same Master.

#### 4.1.1.2 Special Packet Generation

The STH has the ability to generate five *global* packet types: Trigger (TRIG), Trigger with Time Stamp (TRIG\_TS), Cross-Synchronization (XSYNC), Cross-Synchronization with Time Stamp (XSYNC\_TS), and Global Error (GERR); see Figure 7. There is also one form of USER\_TS packet which is considered a special, or global, packet, described later in this section. These packets are 'global' because they are not associated with any Master or Channel. As such, they are generated by a write to the identically-named STH CSR (TRIG, TRIG\_TS, XSYNC, etc.). Since the MIPI STPv2 specification allows for 8-bits of payload with these packet types, the STH only sends the lowest-order 8-bits of data on a write to these 32-bit sized CSR locations.

**Figure 7: Special Packet Generation CSRs Map**



#### 4.1.1.2.1 Special Packets Master/Channel Assignment

It is very important to note that all of the special (or global) packet types described in this section are *not* related or tied to any specific Master or Channel, including the TRIG and TRIG\_TS packets. Because of this, all the packet types referred to as special or global packets are sent with the Master and Channel fields sets to zero. As such, they will be routed to the trace destination for Master 0 Channel 0 (specified by the GTH register SWDEST[0]).

#### 4.1.1.2.2 Trigger Packets

As mentioned above, a trigger packet can be generated by writing to the TRIG or TRIG\_TS CSR. The trigger with timestamp (TRIG\_TS) dataset can also be generated by the STH controller in response to the Common Trigger Sequencer (CTS) indicating a trigger event (i.e. asserting its trigger signal). This can be due to several possible scenarios as explained in Section 4.4.

#### 4.1.1.2.3 Cross Synchronization Packets

The cross synchronization packet can only be generated by writing to the XSYNC or XSYNC\_TS CSR. In this revision of the STH specification, there are no use cases for the XSYNC or XSYNC\_TS packets. While the STH will be able to generate these packets, the details of how and when these are generated are outside the scope of this specification.



#### 4.1.1.2.4 Global Error Packets

Global Error (GERR) packets are created in response to a data drop condition as explained in Section 4.3, and also when requested by a software write to the GERR register. The definition and use of GERR payload, except for data loss condition, is outside the scope of this specification.

#### 4.1.1.2.5 User Packets

There are two specific formats of USER\_TS packets that are considered “special packets”: the start-of-data-loss packet, and the end-of-data-loss packet. These are generated when the STH enters its lossy operation mode and starts dropping data. In this case, USER\_TS packets are used to indicate the start and end of the lossy operation period, so that reconstruction software can properly display the data. Like other special packets, these USER\_TS packets are sent only to Master 0 Channel 0.

User-defined packets (USER) and user-defined packets with time stamp (USER\_TS) are also created like other packets when software writes the appropriate address in the STMR MMIO region for the corresponding Master/Channel. These USER/USER\_TS packets are associated with a particular master and channel. The format, interpretation, and use of USER and USER\_TS packets other than for indicating the start and end of the data loss period is beyond the scope of this specification.

### 4.1.2 Firmware Trace Memory Region

#### 4.1.2.1 General

The Firmware Trace Memory Region (FTMR) appears in system memory as determined by the FW\_BAR PCI configuration space register (which is a concatenation of FW\_UBAR and FW\_LBAR). The term “firmware” (and FTMR) is used here to differentiate it from the STMR. “Firmware” trace sources are those devices, and their associated firmware (if any), that do not have PCI enumeration or discovery capabilities. As such, these devices need to be told, by some other device or process (e.g., system BIOS), where the FTMR is located in the system memory map. Once they are told the location of the FTMR, this location should not change. It is important to note that the FW\_BAR is not located in a standard PCI BAR location. This, again, emphasizes the unusual nature of this MMIO region, in that it will not be scanned or enumerated like a standard PCI BAR. Thus, an operating system will not be able to relocate this BAR as it does other PCI memory regions.

Similar to the STMR, the FTMR must be located in Un-Cacheable (UC) memory space because it is designed for only a strongly-ordered memory model.

Software can determine the number of masters in the FTMR region using the STHCAP1 and STHCAP2 registers. Assuming FW\_MSTR – FW\_MSTR\_STRT number of masters will be allocated for firmware, with SW\_CHCNT channel per master, then, with 64B for each channel, the total size of FTMR, in bytes, is given by the following formula:

$$\text{Size(FTMR)} = 2^{\log_2 \lceil 64 \times (\text{FW\_MSTR\_STP} - \text{FW\_MSTR\_STRT}) \times \text{SW\_CHCNT} \rceil}$$

For example, an FTMR implementation with 32 Masters and 128 Channels per Master would occupy  $64 \times 32 \times 128 = 256\text{KB}$  of MMIO space.

The FTMR is structured identically to the STMR in terms of addressing Masters/Channels, STP packet types within each channel, and so on. That is, Figure 5 and Figure 6 apply to the



FTMR (substituting FW for SW as appropriate). Please refer to section [4.1.1.1](#) for more details.

#### 4.1.3 Intel Processor Trace Memory Region

The STH optionally supports Intel® Processor Trace (Intel® PT). Intel PT support can be determined in the STH\_CAP1 register. Intel PT hardware uses a set of MSRs (specifically, the IA32\_RTIT\_\* MSRs) to set the base address and size of the destination port for each Core/Thread. Correspondingly, the STH memory window for Intel PT is configurable, using the RTIT\_BAR PCI Base Address Registers. The CPU MSRs and the RTIT\_BAR must be set to the same starting address in order for the STH to accurately capture the Intel PT trace data.

The STH's Intel PT Trace Memory Region (RTMR) is shown in Figure 8. The RTMR must be mapped into Uncacheable Speculative Write Combining (USWC) memory space, as the RTMR is designed specifically for this type of memory, matching the high-performance implementation of the IA Core's Intel PT hardware, which was designed to specifically utilize the CPUs write-combining buffers and USWC memory space.

Similar to the STMR, the RTMR is designed to accommodate a variable number of masters and channels. A channel is assigned to each logical processor (where a logical processor is a thread for a multi-threaded core, or a core for a single-threaded Intel® Atom™ processor), starting at channel 1 (the STP Channel 0 in the Intel PT Master is not used). Assuming *NUM\_LP* Intel PT logical processors will be implemented, and by allocating CLLP cachelines for each Intel PT logical processor, the total size of RTMR is given by the following formula in bytes:

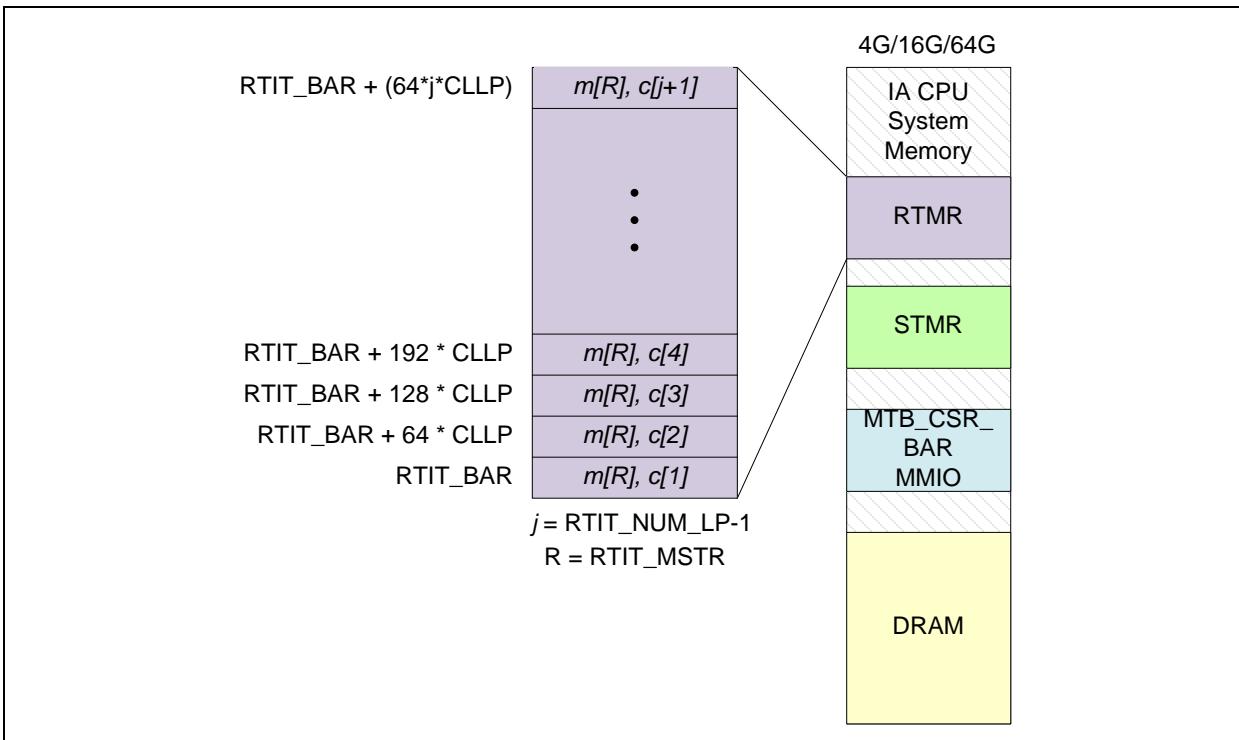
$$\text{Size}(RTMR) = 2^{\log_2[64 * \text{NUM\_LP} * \text{CLLP}]}$$

For example, an RTMR implementation with 4 Intel PT sources (threads) and 4 cache lines per Intel PT logical processor, would occupy  $64 * 4 * 4 = 1\text{KB}$  of MMIO space.

From Figure 8, it can be shown that the base address for each Intel PT logical processor is given by the following formula:

$$\text{Base}(RTIT\_MSTR\_STRT} + m) = 64 \times m + \text{CLLP} + \text{RTIT\_BAR}$$

Figure 8: Intel PT Trace Memory Region Memory Map



#### 4.1.3.1 Intel Processor Trace support

Unlike the STMR and FTMR, Intel Processor Trace support in the Intel Trace Hub implements only “plain” STPv2 data packets (data packets marked and time stamped are not needed). That is, any write to a location within the RTMR will result in an STPv2 data (neither marked nor timestamped) packet.

While technically possible, it is not advisable to send Intel PT data to DRAM through the Intel Trace Hub. From a system perspective, if PT data needs to be sent to DRAM, the CPU can directly send it to DRAM without involving the Intel Trace Hub.

#### 4.1.3.2 Re-ordering

Because the CPU can send Intel® PT data to the Intel Trace Hub out-of-order, a protocol has been created for the Intel Trace Hub that provides the ordering information to downstream decoding software, eliminating the need for hardware buffering in the Intel Trace Hub. Thus, the STH will send all PT writes straight through to the destination in the order they were received from the CPU—whether in-order or out-of-order.

To enable the decoding software’s ability to properly re-order the resultant PT data, a header is prepended to the PT data to aid software in doing the re-ordering. The header is only sent when it is needed: for the first PT write in an in-order stream or collection, and for each out-of-order write. Headers are not sent for in-order data. Data is expected to be in-order most of the time, so this makes for the most efficient protocol possible, exceeding 94% efficiency (including STPv2 encoding).

The Intel PT header is an STP D32M packet, with data payload as shown in Table 4-1.

**Table 4-1: Software Re-Ordered Intel Processor Trace Header Definition**

Bit(s)	Field	Description
31	Start	Start of PT Packet sequence. Indicates PT packet forwarding in the Intel Trace Hub was just turned on.
30	Tag	Tag bit. This bit is toggled for each set of cache lines received by STH. In effect, it becomes an address bit, or sequence number, useful to reconstruction software.
29	Rsvd	Reserved
28	cls	Cache Line Size. 0 = 64B 1= reserved
27:23	CLLP	Cache Lines per logical processor. 0 = 1 cache line 1 = 2 cache lines ... 1F = 32 cache lines
22:14	len	Number of valid PT bytes in this group of packets. Encoding is len – 1. Values 0 – 0x1FF correspond to 1 to 512 bytes.
13:0	address	Starting address (byte address) of valid data. 14 bits enables ability to distinguish 32 cache lines, 512B per cache line.

The re-ordering protocol can be distilled down to a set of requirements that govern how the Intel Trace Hub sends PT data out-of-order. These are enumerated in the list below. All requirements are per-logical-processor unless otherwise specified. See section 4.1.3.2.1 for definitions of terms used.

A PT packet set is a set or group of STPv2 packets that, together, comprise a single PT write received by the Intel Trace Hub on the primary fabric bus. Because the Intel Trace Hub operates on only 64 bits of data at any given time, but PT data can be as much as 64 bytes, the Intel Trace Hub will split the single PT write into multiple STPv2 packets.

- 1) The first PT packet set transmitted by the Intel Trace Hub will have a header.
- 2) The first PT packet set transmitted by the Intel Trace Hub will have a start bit set.
- 3) All header packets will contain encoded values indicating cache line size (CLS) and number of cache lines per processor/thread (CLLP).
- 4) All header packets will include the number of bytes, and the starting address within the cache line for the first byte.
  - a. For full cache lines, the number of bytes is equal to the cache line size (CLS)
- 5) All out-of-order and partial-cache-line packet sets transmitted by the Intel Trace Hub will have a header.
- 6) The Intel Trace Hub will include a “tag” bit in the header, identifying which open bucket a cache line belongs to. The tag bit will start at zero, toggle to one, then back to zero, and so on.
- 7) The determination of whether a packet is in-order or out-of-order is based solely on the starting address of the previous PT write set. That is, if, for CLLP = 4, cache lines are received in the following order: 1, 3, 2, 4, then headers will be generated for all

four cache lines, as they are all out-of-order with respect to the previous write set. To further illustrate this point, if the cache lines are written in the following order: 1, 2, 4, 3, then headers are generated for cache lines 1 (the starting cache line), 4, and 3; cache line 2 is the only cache line that is in-order with respect to the previous write set.

- 8) The header packet will not be delivered for any in-order full cache line. That is, any 64 byte Intel PT write to the Intel Trace Hub that follows, in-order, either a full or partial cache line, will not have a header.
  - a. The partial cache line write preceding such an in-order full cache line write must complete that partial cache line. That is, if bytes 0...n of cache line M were received by the Intel Trace Hub already, the only way cache line M+1 can be transmitted without a header packet is if the write immediately preceding cache line M+1 contains bytes n+1 ... 63 of cache line M.
- 9) The Intel Trace Hub STH will optimize the size of STPv2 data packet whenever possible.

#### 4.1.3.2.1 Intel Processor Trace Re-Ordering Definitions

Terms:

- 1) A cache line is the fundamental size of data in Intel PT.
- 2) "Bucket" refers to the number of cache lines for a logical processor. It is the same as CLLP parameter.
- 3) Only *one* out-of-order window for a logical processor can be open at any given time. An out-of-order window is two buckets. That is, a maximum of two buckets can be "open" at any given time when re-ordering data.
- 4) A PT packet set is a set or group of STPv2 packets that, together, comprise a single PT write received by the Intel Trace Hub on the primary fabric. Because the Intel Trace Hub operates on only 64 bits of data at any given time, but PT data can be as much as 64 bytes, the Intel Trace Hub will split the single PT write into multiple STPv2 packets.
- 5) A trace window is a consecutive series of data, made up of more than zero buckets. A trace window could be as small as one byte, or as large as several gigabytes. A trace window is bounded by the STH storenEn=0→1 and 1→0.
- 6) A trace buffer is a collection of data that can be as "small" as a partial trace window (e.g., attempting to store a large trace in a small ring buffer) to as "large" as several trace windows.
- 7) In-order refers to data bytes written to consecutive byte addresses (modulo CLLP). In-order scope only applies to transaction n and n+1. That is, transaction n+1 in-order-ness is determined solely by the address of transaction n. The following cache lines numbers are all considered in-order (CLLP = 4), even though they span multiple buckets: 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 0
- 8) Out-of-order refers to data bytes written to the Intel Trace Hub that occupy non-consecutive byte addresses (modulo CLLP). The following cache line write examples (all of which are valid examples) have at least one out-of-order cache line. Each number represents a cache line (numbered 1 to 4), and CLLP = 4.
  - 1, 3, 2, 4
  - 1, 2, 4, 3

- 1, 2, 3, 4, 4
- 1, 1, 2, 3, 4
- 1, 2, 3, 2, 4

#### 4.1.3.3 Intel Processor Trace and Lossy Operation

The Intel Processor Trace protocol is designed to accommodate the unreliable transmission of data to its destination. In order for the trace data to be reconstructed, the loss must be sensed, and accounted for, at the CPU itself. If data is dropped after being sent from the CPU, but before arriving at the final destination, then the PT trace will not be reconstructible; in other words, it will be useless.

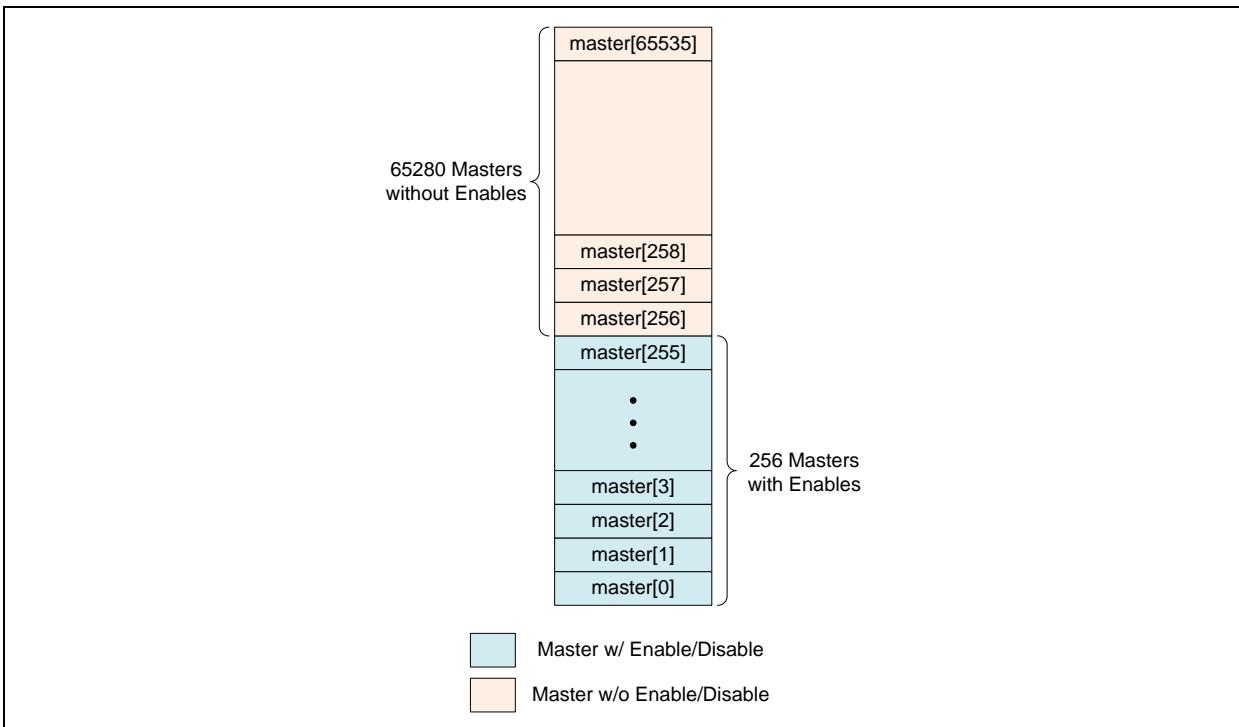
The Intel TH STH is, by design, “lossy”, so as to introduce the least possible perturbation in system operation while tracing is active. The STH Backpressure Duration Counter (BDC) can be set to a non-zero value, or it can be completely disabled (see the GTHMISC register), to decrease the lossiness of the STH. However, it is not possible to make the STH lossless. Increasing the BDC value above zero, or disabling it entirely, has the potential to negatively impact system performance because the Intel TH will assert backpressure to the primary fabric if trace data cannot be accepted.

## 4.2 Master Number Assignment and Allocation

The Intel Trace Hub has two kinds of Masters (Master, in this context, refers to a MIPI STPv2 Master): those with a dedicated enable bit (a per-master enable/disable), and those without. The purpose of a Master enable bit is to allow a master to send data to the GTH, while giving some other software (e.g., debug tool software – the user) the ability to control whether that data is sent to the destination or dropped. The primary use case for this capability is for “firmware” masters: these devices typically have very limited code space, and strongly desire that the execution path not change, whether the Intel Trace Hub debug messages are being sent or not. These firmware Masters will always send their debug data to the Intel Trace Hub. If the debugger desires to not see that debug information, simply disable the relevant Masters.

The number of Masters with a per-master enable is limited to the lower 256 (as shown in Figure 9) in order to minimize the STH physical size, while still providing sufficient per-master-enable capability.

**Figure 9: STH Master Allocation**

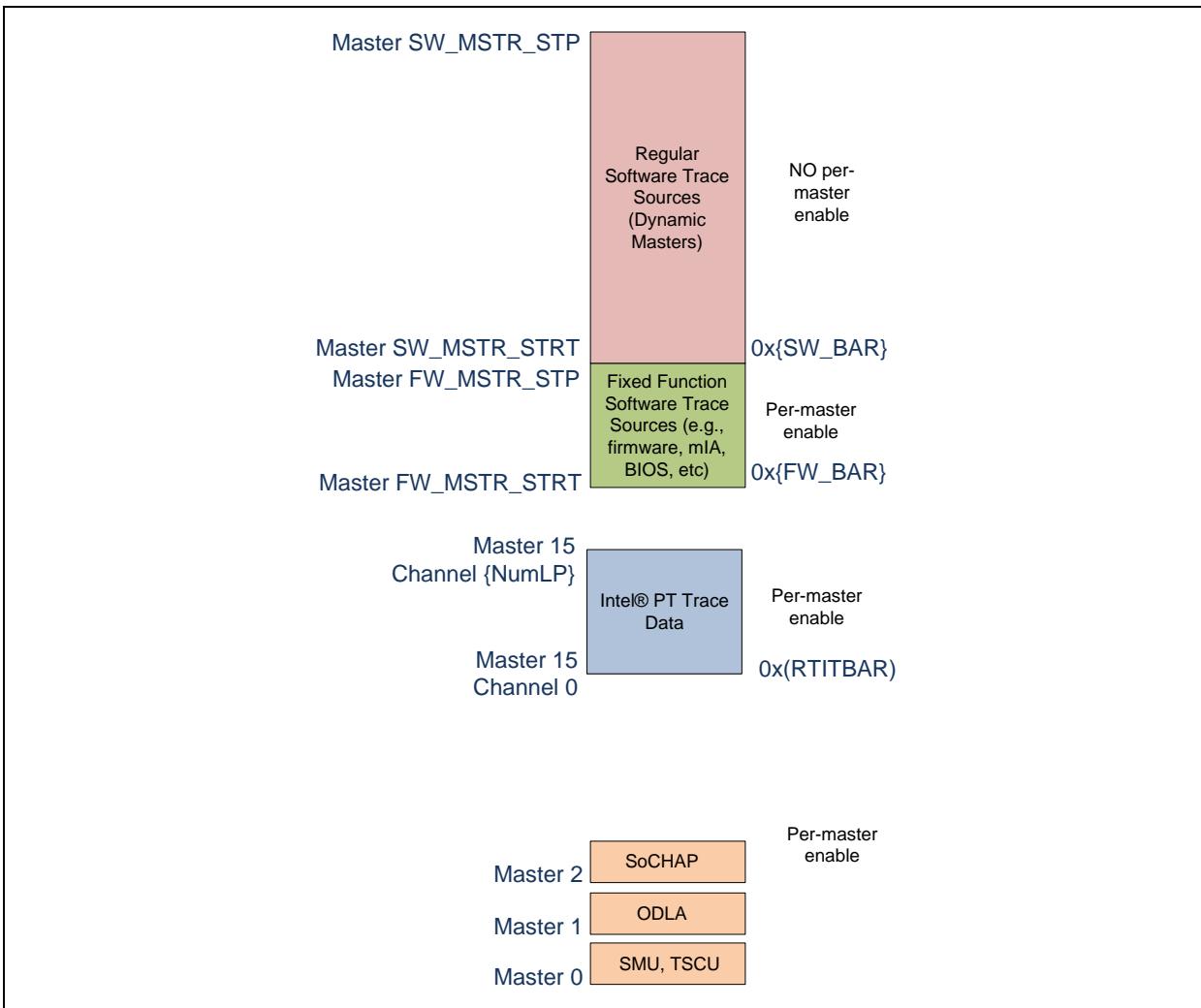


However, to enable different projects' ability to specify which sources use the Masters that implement an enable/disable control, and those that do not, the STH contains three registers that can be used to map the software and firmware Masters to a subset of the available Masters. These registers are contained within STHCAPO/1/2.

**Table 4-2: STH Master Allocation Registers**

Register	Description
FW_MSTR_STRT	<b>Firmware Master Start:</b> This parameter specifies the Master number of the first Firmware source. Firmware masters are those that are accessible via the FW_BAR memory window.
FW_MSTR_STP	<b>Firmware Master Stop:</b> This parameter specifies the highest Master number allocated to "firmware" devices. This number should be less than or equal to 255, as firmware masters need the per-master enable capability of the Intel Trace Hub.  The difference between SW_FW_MSTR_STRT and FW_MSTR_STP gives the number of Masters allocated to firmware.
SW_MSTR_STRT	<b>Software Master Start.</b> This parameter specifies the starting Master number for Software Masters. Software masters are those which are accessible via the PCI BAR #1, called the STMR.
SW_MSTR_STP	<b>Software Master Stop:</b> This parameter specifies the highest Master number allocated to "software". This number should be less than or equal to 65535 ( $2^{16}-1$ ).

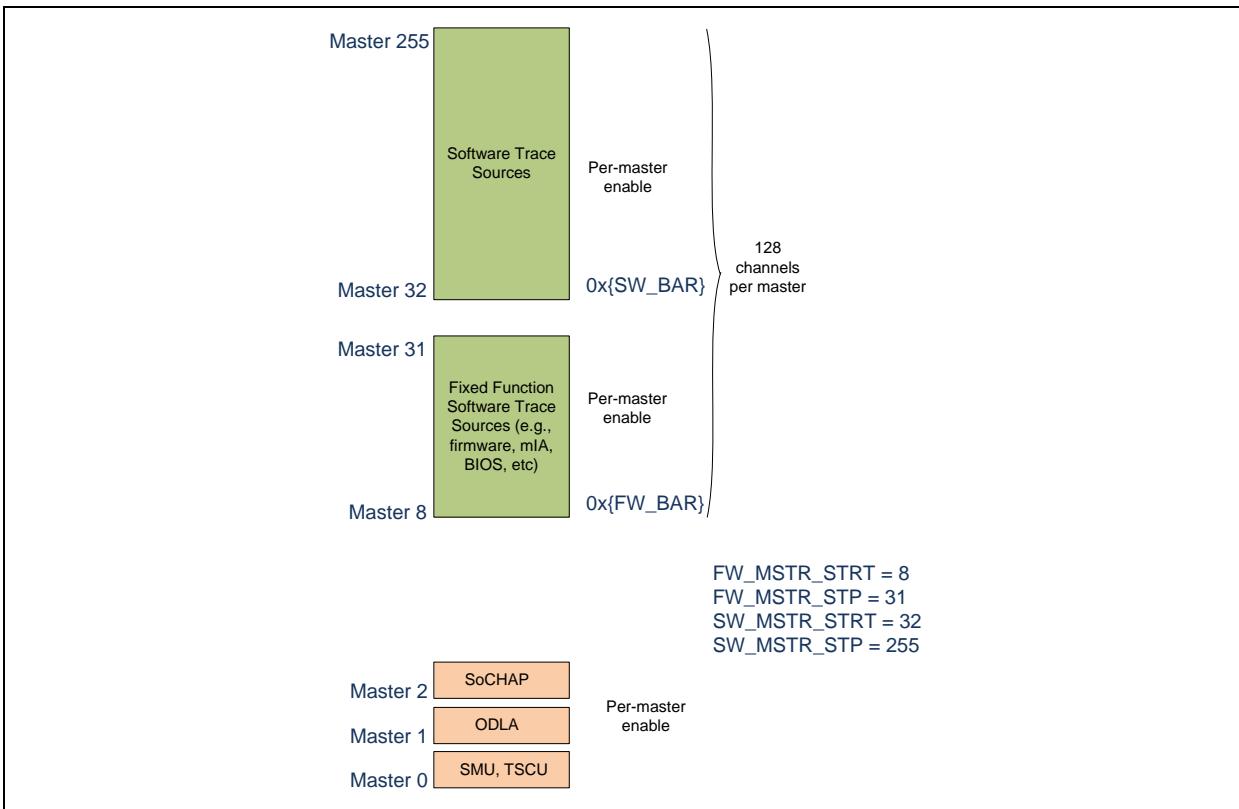
**Figure 10: Intel Trace Hub Master Allocation**



Each product can allocate its firmware and fixed-function masters, as well as software masters, to match their requirements. Two examples are provided as follows.

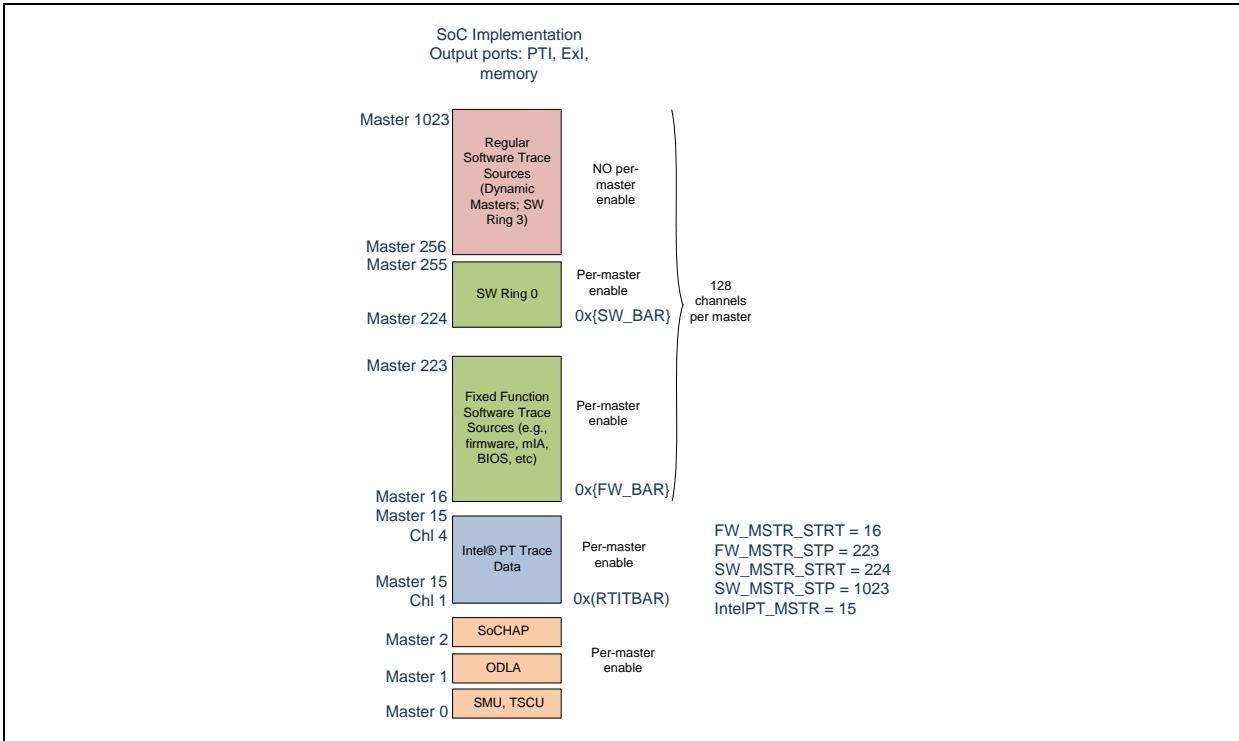
In the first example, shown in Figure 11, Intel PT support is not implemented, and only hardware and software tracing are supported. In Intel Trace Hub architecture, hardware sources are allocated to the lowest-numbered Masters (0, 1, and 2 in this generation). Firmware trace sources are assigned 24 Masters, starting at Master 8; and general software trace sources are assigned 224 Masters, starting at Master 32. In this example, all firmware and software trace sources have a per-master enable.

Figure 11: Master Allocation Example 1



In the second example, shown in Figure 12, Intel PT support is present, and many more software Masters are supported (greater than 256; the actual number is unimportant). In this example, firmware and fixed-function software trace sources were allocated to the “firmware” region, and have per-master enables. Software Ring 0 is assigned 28 Masters starting at Master 224. Because of the Master range assignment, the Ring 0 sources have a per-master enable. Starting at Master 256, regular software trace sources can be assigned to one or more Masters, according to the algorithm of the software or the desire of the user.

**Figure 12: Master Allocation Example 2**



**NOTE:** It is possible that a range of software Masters will appear in both the FW\_BAR region (FTMR) and the SW\_BAR region (STM). This is because the FW\_BAR must ask for, and be allocated, a memory size that is an integer power of two (i.e., it cannot be allocated 1.875MB; the nearest allowed size is 2MB). Since firmware Masters start above zero, and the SW\_MSTR\_STRT value is merely added to the Master number for offset zero of the SW\_BAR), then generally, masters FW\_MSTR\_STP to FW\_MSTR\_STP + FW\_MSTR\_STRT will be present in both the FW\_BAR and the SW\_BAR. It is the responsibility of firmware and software engineers to ensure their device(s) write to their assigned Master number in their assigned memory region (BAR).

## 4.3 Lossy Operation

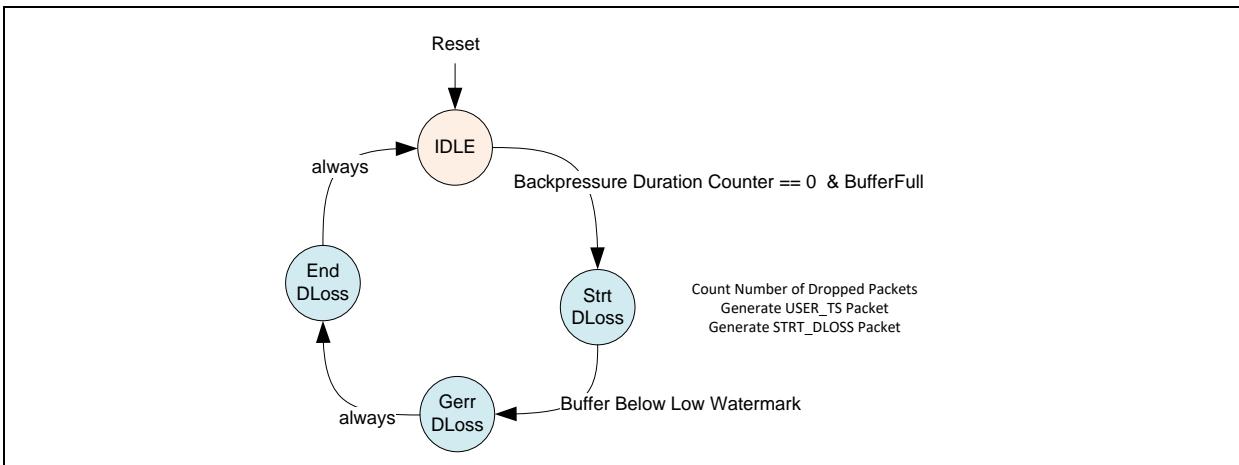
The STH is designed intentionally to be lossy. For it to be lossless would mean to introduce significant perturbation to system operation, which conflicts with the goals of the Intel® Trace Hub. The lossy operation supports minimal system perturbation while storing to primary fabric-based destinations.

When new incoming data is presented to the STH, but it is unable to accept incoming data (most likely because the GTH Input Buffer to which the STH is connected is full) it will start decrementing the Backpressure Duration Counter (BDC). If the GTH starts accepting data again before the counter expires, the counter is reset to its programmed value.

On the other hand, if the counter expires before the GTH starts accepting data again, the STH will enter its Drop Mode, and start its Recovery FSM (finite state machine). Once Drop mode is entered, all data written to the STM or RTMR will be accepted but immediately dropped. Similarly, any write requests targeting the special CSRs will be accepted

immediately and the resulting dataset dropped. Finally, no hardware triggers will be accepted and all resulting trigger datasets will also be dropped.

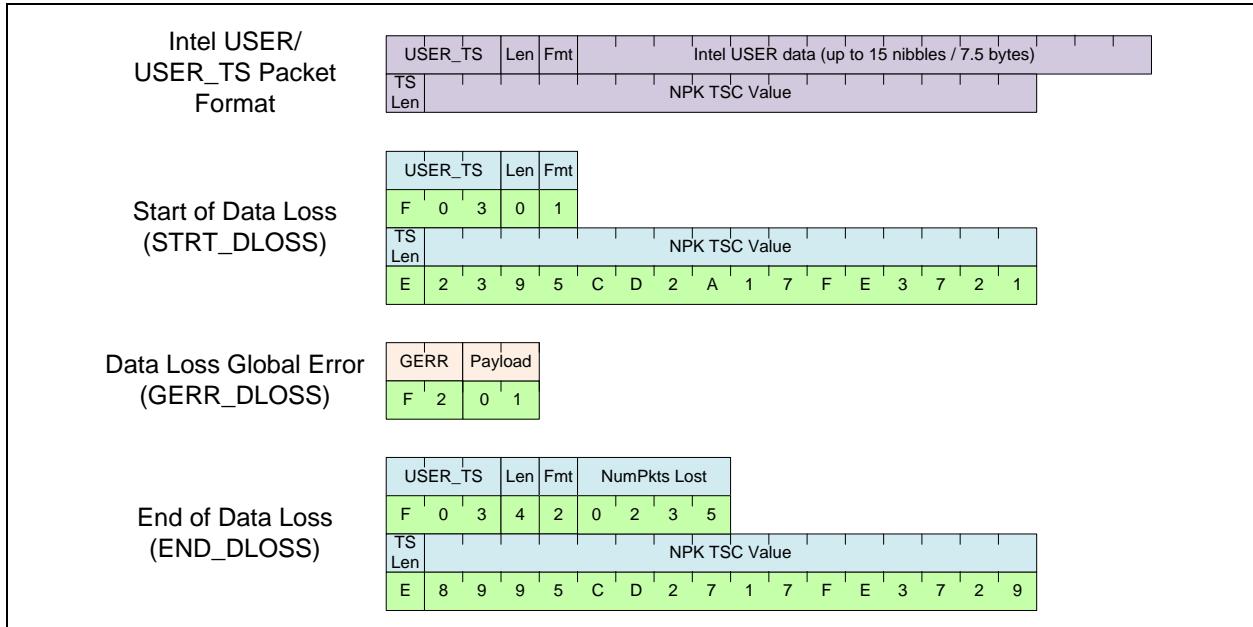
**Figure 13: STH Lossy Mode Recovery FSM**



In addition, the first of three special data packets defined to convey the data loss condition to the capture device is queued up. This special packet is referred to as the *Start of Data Loss Packet*, shown in detail in Figure 14. While the FSM is in the *StrtDLoss* state, it will also use a counter known as the Data Packet Lost Counter (DPLC) to count the number of data packets that have been dropped while in this mode of operation. The number of lost packets includes packets from the inbound bus (i.e., coming from primary or sideband fabric), but does not include trigger packet(s) that would have been generated as a result of a TrigIn assertion (from the Trigger Unit). This counter is also visible as a CSR and includes a sticky overflow bit that indicates when the number of data packets lost is greater than the 15-bit counter can accommodate.

The FSM will remain in the *StrtDLoss* state until the corresponding GTH's input buffer asserts its *lowWaterMark* signal, indicating that the input buffer is full only to its low water mark. This approach will ensure that, once data is accepted into STH again, a reasonable amount of data will be passed through before there is a chance of another backpressure and more data loss. Once the *lowWaterMark* signal is asserted, the FSM will move to the *GerrDLoss* state. The FSM is in this state for one clock cycle during which it will generate the second "data loss" special packet, the GERR\_DLOSS packet. Moreover, the DPLC will stop being incremented at this state. Next, the FSM will move to the *EndDLoss* state for one clock cycle, during which the third special packet is generated, referred to as End of Data Loss (END\_DLOSS) packet, which conveys the number of packets dropped during the lossy period, plus the time stamp right before the STH goes back into normal operation. The FSM will then move back to the *IDLE* state, allowing the STH to resume its normal operation. The three special packets generated by the Recovery FSM are depicted in Figure 14.

**Figure 14: Data Loss Special Packets Format**



## 4.4 Triggering and Events

### 4.4.1 Event Match and Mask

The STH monitors incoming data (from the Intel Trace Hub Host Interface) for up to four different data packets, simultaneously. If the incoming data matches the specification in one of the Event Mask/Match registers, the STH will assert an *event* signal to the Intel Trace Hub Trigger Unit. While the match/mask control register contains a BAR (Base Address Register) match and mask value, only transactions targeting the STH will actually be monitored. More specifically, only writes to the STMR, RTMR, and STH CSR range will be monitored; writes to GTH CSRs, or other units' CSRs, will not be monitored, and cannot result in an event assertion to the Trigger Unit. Further, event monitoring is independent of the STH's *storeEn* signal. That is, the event match/mask logic is monitoring and reporting events, whether *storeEn* is asserted or deasserted.

Based on its configuration, the CTS can assert back the *TrigIn* signal one or more times. The behavior of the STH in this situation is described in the following section.

---

**NOTE:** The CTS has the capability to assert *storeEn* to the STH based on an event signal from STH to CTS. Unfortunately, due to the latency from STH event detection to TrigIn assertion, the event that precipitates a *storeEn* assertion will disappear (will not be stored; not sent from STH to GTH) before the *storeEn* is actually asserted. Thus, the "triggering event" will not be the first event to appear in the resultant trace data.

---

### 4.4.2 Trigger with Timestamp Packet Generation

If TRIGTSPP.TRIGEN is set, then each time the *TrigIn* is asserted the STH will insert a TRIG\_TS packet into its data stream.



Take note that the STH can keep track of only one input trigger at a time while the GTH is full. If two or more TrigIn assertions are detected while the GTH is full, only the most recent TrigIn assertion will result in a TRIG\_TS packet (specifically a single TRIG\_TS packet, not multiple TRIG\_TS packets), with the timestamp of the most recent TrigIn assertion.

On the other hand, if BDC ever expires during the above process, everything is simply dropped/ignored until the STH recovers from the Drop Mode operation then re-start collecting and sending the received TRIG\_TS packets.

## 4.5 Time Stamping

The STH has the ability to time-stamp incoming data. Since it is a member of the Intel Trace Hub Architecture, it uses the Time Stamp Counter from the TSCU for time-stamping data upon demand. Because the STH will be physically very near the TSCU, the full 64-bit Intel Trace Hub TSC is connected as a direct input to the STH for use in time stamping.



## 5 DTF Bridge

---

The DTF Bridge (NDB) provides the "Trace Source" functionality required by the Intel Trace Hub architecture, for trace sources that send their trace data via the DTF (Debug Trace Fabric) bus/protocol. A trace source must have an "on/off" switch under the control of the Trigger Unit so that it can enable and disable the source(s) according to the user's/debugger's need (i.e., according to the trigger unit's settings).

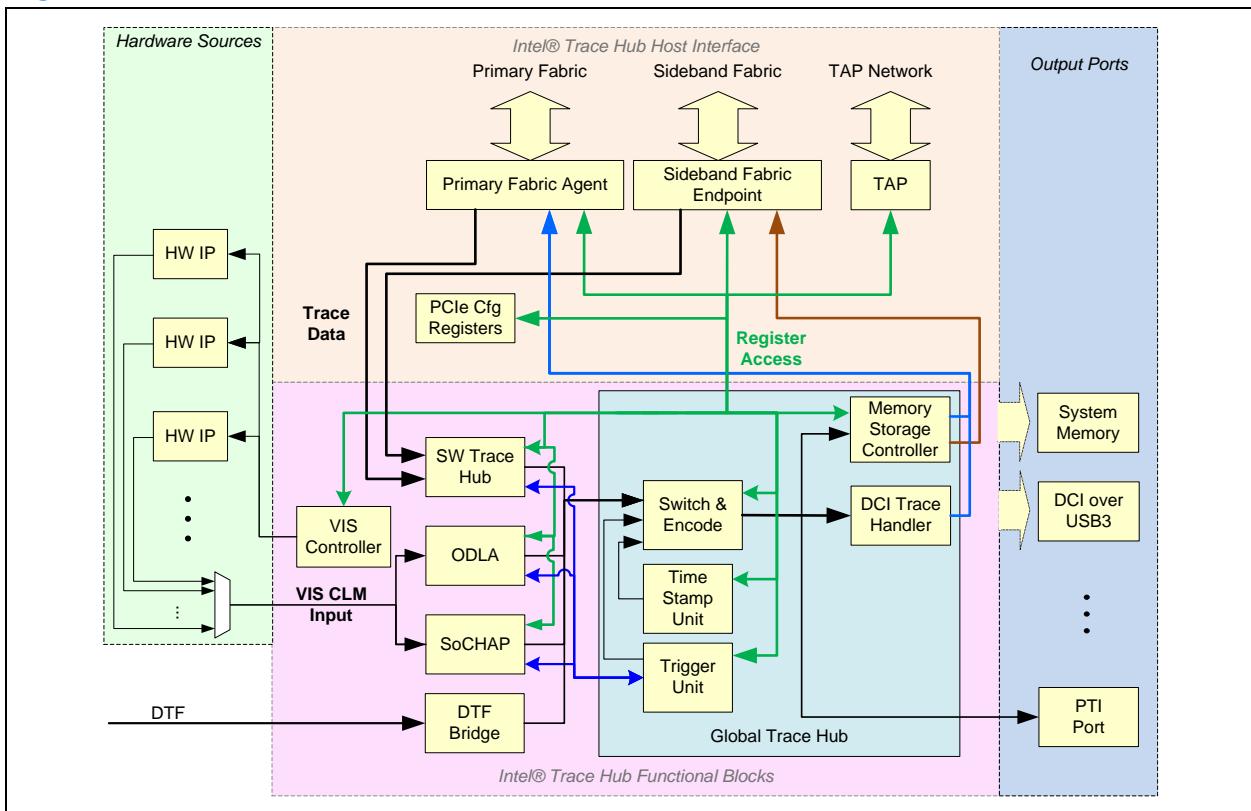
The DTF is a lossless transport for trace data from a source into the Intel Trace Hub. The transport inside the Intel Trace Hub is also lossless, providing for an end-to-end lossless trace solution. Since the NDB is merely a lossless bridge, there is little user-controllable functionality in it. First, the user must set each NDBMWn bit according to the trace destination for DTF sources. Second, the user must choose how long to "wait" after the DTF StoreEn deassertion before presuming the DTF is empty. This is because the DTF does not indicate to the Intel TH whether it is "empty" or not.

See the NDB\_CTRL register for more information.

## 6 Global Trace Hub

The Global Trace Hub (GTH) is an ingredient of the Intel Trace Hub Architecture: it is the central block that gathers all debug trace data<sup>9</sup>, encodes it in MIPI STPv2 format, and sends it to various destinations to suit the needs of the debugger. The debug trace data can be stored in system memory for a truly probe-less debug solution. It can also be sent to any supported trace destination or debug port, including the DCI and MIPI-PTI. The following figure® illustrates the GTH architecture in the context of the Intel Trace Hub architecture, and shows some of the internal GTH components.

**Figure 15: GTH in Intel Trace Hub**



### 6.1 Overview

- Supports the following trace data sources:
  - ODLA
  - SoCHAP

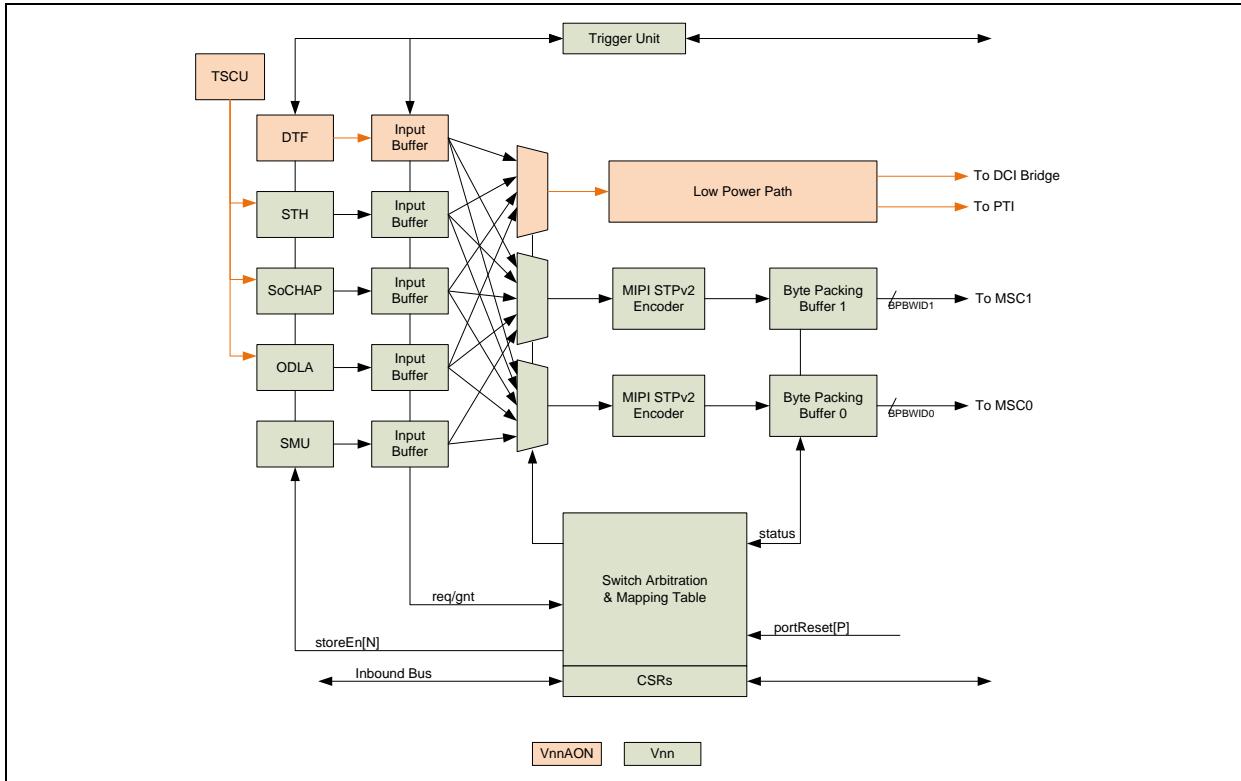
<sup>9</sup> Debug trace sources are those IP blocks (usually DFx blocks) that gather or generate data used for debug, and include ODLA, STH, and SoCHAP

- Software Trace Hub
  - Intel Processor Trace Sources
  - Regular Software sources
  - Firmware sources
- DTF Bridge
- STP Maintenance Unit (an integral component of GTH)
- **Compliant to MIPI STPv2.1 protocol standard**
  - Supports up to 65536 Masters, and 256 Channels per Master
- **Supports multiple output ports**
  - System DRAM (via primary fabric)
  - DCI
  - MIPI-PTI
  - MIPI-HTI
- **One destination for each STPv2.1 Master**
  - No multicast or broadcast
- **Backpressure:** Supports debug data sources with and without backpressure support
  - Trace Sources with backpressure support will cleanly drop packets
  - Trace sources without backpressure support will have forcibly dropped packets.
- **Intel Trace Hub Internal Bus interfaces** for sending and receiving debug trace data, and accessing CSRs (control/status registers)

## 6.2 Functional description

The GTH is a central element of the Intel Trace Hub architecture. The GTH is essentially a data switch with a protocol encapsulation function. It takes, as inputs, debug data from various trace sources, encapsulates that data into the MIPI STPv2.1 protocol, and sends it to a destination port. Because the GTH is a switch, supporting multiple simultaneous destination ports, different types of data can be sent to different ports. For example, Intel Processor Trace data can be sent to the PTI port, while software (application, OS, and driver) debug messages can be sent to system memory (via primary fabric). The number of output streams is implementation dependent, though there is a minimum of one port. Output ports include, but are not limited to: system memory (which can also be thought-of as the primary fabric interface), external bridges to other interfaces, MIPI HTI, and MIPI PTI.

**Figure 16: Global Trace Hub Architecture**



The Global Trace Hub is comprised of the following functional blocks:

- Input Buffers: a simple set of buffers that are provided to allow for absorbing input bursts.
- Data Switch: a full-crossbar switch, allowing any single input to be routed to any single output.
- MIPI STPv2.1 Encoders
- Byte Packing Buffers
- STP Maintenance Unit (SMU): sends periodic MIPI STPv2.1 synchronization packets to the trace destination.
- Switch Arbitration and Mapping Table
- Configuration and Status Registers: provides access to CSRs

These blocks are presented in more detail in the following subsections.

Throughout this document, the designation '[N]' and '[P]' are used to refer to trace sources and output ports, respectively. That is, whenever a signal or aspect of the architecture applies to the number of trace sources, or a particular trace source, the letter 'N' is used to refer to it. Whenever a signal or aspect of the architecture applies to, or refers to, the number of trace destinations, or a particular trace destination, the letter 'P' is used to refer to it. For example, in Figure 16, the signal *full[P]* uses the letter 'P', indicating there is a *full* signal for each output port. Similarly, the signal *grant[N]* uses the letter 'N', indicating there is a *grant* signal for each trace source (input buffer).

## 6.2.1 Input Buffers

The Input Buffer serves as a peak bandwidth buffer, as well as a latency buffer between the trace source and the Data Switch.

Data is accepted into the Input Buffer from a trace source when there is at least one free entry.

The Input Buffer performs the first two tasks in the overall data switching operation. First, it filters out trace data from disabled Masters. That is, when the MAST[N]EN bit in SWDEST[N] register is cleared (zero) for a particular Master, trace data is received at the input buffer from that Master and it will be immediately dropped.

Second, for Masters that are not disabled, the Input Buffer looks up the output port for that Master, and stores the destination port number (dest[2:0]) in the Input Buffer, along with the trace data.

The input buffer has one responsibility in supporting the GTH switch performance optimizations: it requests to send data to the Data Switch when there are sufficient data sets in the Input Buffer to ensure the programmable grant duration (*grantDur[3:0]*) will be fully utilized. The Data Switch arbiter is designed to accommodate the case where fewer than *grantDur* data sets are ready to be forwarded to a byte packing buffer. However, this results in lower efficiency for use of the STPv2.1 protocol, and should be avoided. The only exception to this rule is when the trace source is turned off (by deasserting its Store Enable). The input buffer then asserts its *req* signal as long as there is data in the input buffer, regardless of how many entries are in the buffer.

#### 6.2.1.1 Standard Trace Source Input Buffer Assignment

Since there are a number of standard trace sources that come with the Intel Trace Hub IP, these sources are assigned to fixed input buffers, so that software has a baseline from which to build a standard software release. Toward this end, the following assignments are part of the fundamental architecture of the Intel Trace Hub:

**Table 6-1: Standard Trace Source Assignments**

Trace Source	Trace Source Number	CTS Store Qual Signal	CTS Gasket StoreEn Signal
SMU	0	n/a	n/a
TSCU	1	n/a	n/a
SoCHAP	2	1	2
ODLA	3	2	3
STH	4	3	4
DTF	5	4	5

Additional details on *StoreQual* and *StoreEn* signal assignments are discussed later in this section.

#### 6.2.1.2 Input Buffer Depth

The depth of the input buffer is a synthesis parameter, and will depend on the needs of the particular data source, as well as the characteristics of the available output ports. A depth of zero is permitted, but only for single-destination trace sources. This will be used for the STP Maintenance Unit, as its data is always fixed-format, and does not need buffering. Zero Depth Input Buffers must have their grant duration set to 1.

### 6.2.1.3 Backpressure / Flow Control

The Input Buffer can be connected to trace sources that have backpressure capabilities, as well as those that do not. For sources that have backpressure capabilities, the Input Buffer will indicate to the source when the input buffer is full, and cannot accept more data. When the Input Buffer is full, and the trace source itself is unable to further-buffer data, it should engage its backpressure or flow-control mechanism. The method for dropping debug data and handling that condition in software or post-processing is determined solely by the trace source.

For debug trace sources that do not have backpressure capabilities, the Input Buffer's full condition signal will be left unconnected. In this case, the debug trace source will send its debug data, assuming the Input Buffer has captured it. When the Input Buffer is full, it will not capture the data sent from the trace source; the data will be lost.

The Input Buffer also has a threshold function, whereby it can indicate when the Input Buffer is filled below, or beyond, a certain value. The intended primary functionality is that of a low water mark: the signal (*LowWaterMark*) will be asserted when there are  $LWM_n$  (where  $n$  is the trace source number; see LWM registers) entries or less in the Input Buffer. The primary use of this signal is by the STH, which will use *lowWaterMark* to exit its 'drop mode'. In order to ensure the STH sees the *lowWaterMark* signal asserted and deasserted at the right time, the values programmed into the LWM0 and LWM1 registers need to be such that the low water mark is 1 less than the grant duration for the given trace source.

### 6.2.1.4 Input Buffer Empty

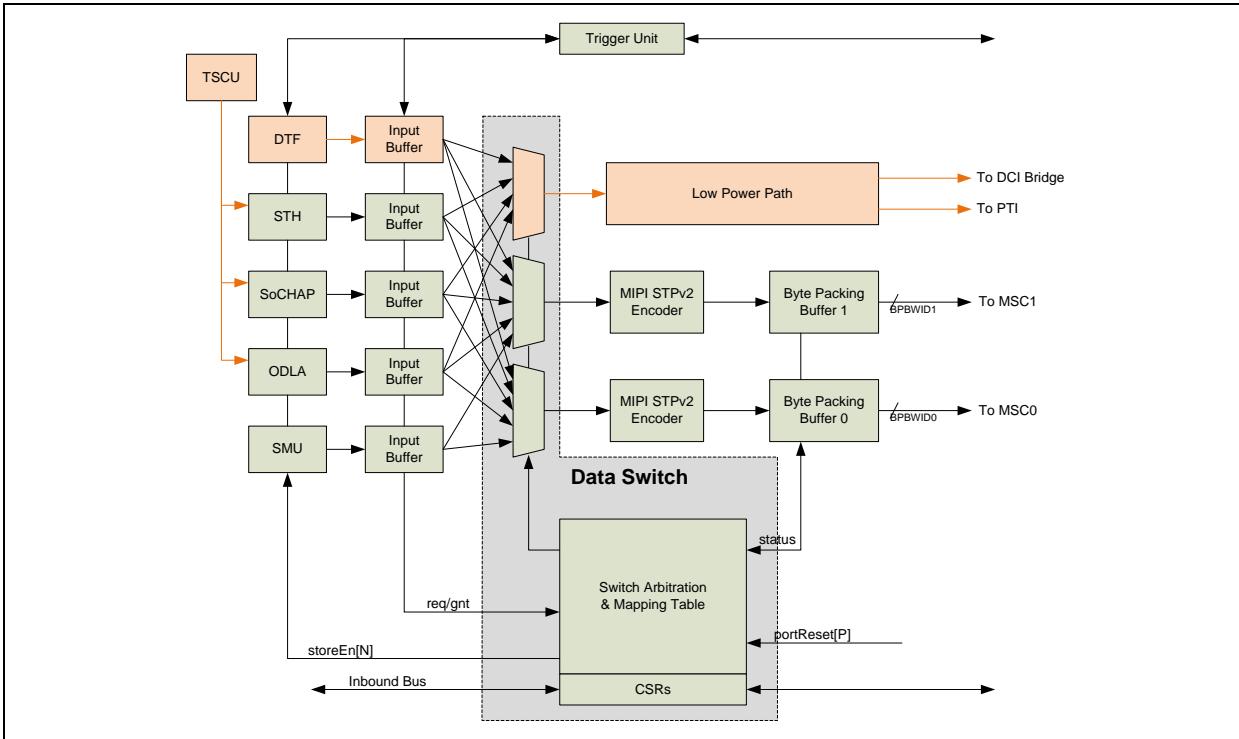
The Input Buffer generates a signal called *iBufEmpty*. This signal is a vector, *iBufEmpty<sub>t</sub>*, where ' $t$ ' is the number of Memory Storage Controllers (MSCs). The *iBufEmpty* signal indicates that, for purposes of the trace destination, the input buffer is empty. It does not indicate the input buffer is actually empty, but rather that, as far as the trace destination  $t$  is concerned, the input buffer is empty. This signal is driven by each input buffer to the MSC to determine if the pipeline feeding it is empty.

## 6.2.2 Data Switch

The Data Switch performs the switching function of the GTH, forwarding data from an input buffer to a destination according to the settings in GTH DEST[N] registers. All data switching decisions are based solely on the STPv2.1 Master, and not on the Channel. That is, all data from all Channels of a given Master will be sent to the same trace destination.

The architecture of the Data Switch allows any input to be sent to any output.

**Figure 17: Data Switch**



Arbitration among the input buffers in the GTH is a simple round-robin algorithm, per Data Switch mux, starting with trace source #0, which must always be the STP Maintenance Unit (SMU). This ensures that, when tracing is started, the ASYNC, VERSION, and FREQ packets are sent first. Trace source #1 does not exist starting in 2<sup>nd</sup> generation of the Intel Trace Hub, as the TSCU now uses the same time base as the CPU, and no time correlation packets are needed (see TSCU Chapter for more details). Arbitration among the trace sources (2 and higher) is also round-robin, with each arbiter receiving single requests from trace sources that have data available for that particular trace destination.

The arbiter employs a mechanism to increase data and packet throughput: programmable grant duration. The grant duration is programmable, allowing designs to maximize protocol efficiency on a per-debug-scenario basis. A greater number of packets sent back-to-back from one trace source results in increased efficiency in the STP protocol, in terms of number of data bits transferred vs. total bits to perform the transfer. This is important because the efficiency of the first packet from a given master is rather low: it ranges from 22% to 70%, depending on the data size. The efficiency of the STP protocol can be increased to as much as 94% if buffering is increased, and several data sets from a single master are sent in a burst. Grant duration can be set from 1 to MAXGNT[n] consecutive clocks. Since there is very little efficiency increase beyond 16 consecutive packets (2%, when increasing from 16 to 32), even for the smallest packet size (8 data bits), the value of the MAXGNT[n] parameter is limited to 16 or less.

When a trace destination is reset, or otherwise needs re-initialization in the middle of sending trace data, it will assert its *portReset* signal. Depending on the setting of GTHOPT.DRP bit, the byte packing buffer may fill up, asserting its full output, and stalling the entire pipeline. The Input buffers will eventually fill up, and stall the trace sources. Because the *portReset* signal is asserted, the SMU will prepare packets for the port in reset. Also, the arbiter for the output port must be "reset", in the sense that it will wait for

*portReset* to deassert, and then immediately start granting at Input Buffer 0 (that is, it does not pick up where it left off). When *portReset* is deasserted, the pipeline will be uninstalled. In-flight packets will be drained, and the arbiter will begin again at Input Buffer 0 (the SMU) for its arbitration.

### 6.2.3 MIPI \* STP Encoder

The MIPI STP Encoder's main function is to translate a data set received from the Data Switch to one or more MIPI STPv2.x encoded packets.

The encoder keeps track of the currently active Master and Channel. For each new data set received from the Data Switch, if the Master and Channel are unchanged, the encoder will not create Master (M16, M8) and Channel (C8) packets, resulting in a more efficient use of the available bandwidth. Conversely, when the new data set Master differs from the current Master, a Master (M16 or M8) packet will be created. Similarly, when the new data set Channel is different than the current Channel, a Channel (C8) packet will be created. Thus, the encoder will create from one, two or three MIPI STP packets for each data set received.

The currently active Master and Channel are reset to zero when the STP Encoder detects the VERSION packet in its input data set. This will result in the next data set having a Master packet, and perhaps a Channel packet, unless the next data set is from Master 0 and Channel 0. The stored timestamp is also reset to zero, which will result in the next time stamp encoded to be a maximum-length time-stamp. This is done to meet the MIPI STP requirement that the next timestamp after a VERSION packet is the longest time stamp that the system is able to produce.

The final stage of the MIPI STP encoder is the Byte Packing Buffer block, which is responsible for ensuring that the MIPI STP-encoded packet(s) are placed onto the low order byte lanes of the output bus. The data is mapped onto the output bus in network order, by nibble, from low (least significant) nibble to high (most significant) nibble. That is, the least-significant nibble on the output bus is the first nibble, in terms of network order.

Since the Intel Trace Hub operates inside Intel SoCs, the STP-encoded *payload* data it produces is little-endian (STP opcodes are merely a stream of nibbles, and have no endianness associated with them. The payload portion of the packets can have an endianness, hence the translation from nibble order to network order is important). That is, the least-significant nibble of a byte, word, DWord, or Qword is sent first, followed by the next least significant nibble, and so on. When sent to an output port, the first nibble is, obviously, the first nibble to be sent. When stored to memory, the data will be stored with first nibble in the least significant nibble of byte 0, second nibble in the most significant nibble of byte 0, the third nibble in the least significant nibble of byte 1, and so on. It is also the first nibble received, and the first to be interpreted. See the next five figures below for detailed information on nibble order for each STP packet encoded by the Intel Trace Hub. See Figure 23 for a diagram showing MIPI STP encoded data as it is stored in memory.

Figure 18: Nibble order (1 of 5)

Note: all data represented below is written in "reading order", but transmitted in increasing nibble # order.

nibble #	bits	Color Coding	opcode	opcode related value	master	channel	user data	time stamp
	M8 0x21	M16 0x7654					D64	
	C8 0xAB	C8 0x32					0x012345	
	D8 0xCD	D32 0x01234567					67	
0	3:0	MB	0x1				D64	
1	7:4	C8	0x1	M16	0xF	C8	0x1	0x7
2	11:8	master	0x2		0x1	channel	0x1	
3	15:12	C8	0x3		0x4		0x0	0x6
4	19:16		0xB		0x5			0x5
5	23:20	channel	0xA		0x6			0x4
6	27:24	D8	0x4	C8	0x3			0x3
7	31:28		0xD		0x2			0x2
8	35:32	data	0xC		0x3			0x1
9	39:36		NULL (pad)	0x0	D32	0x6		0x0
10	43:40				0x7			0x7
11	47:44				0x6			0x6
12	51:48				0x5			0x5
13	55:52				0x4			0x4
14	59:56				0x3			0x3
15	63:60				0x2			0x2
16	67:64				0x1			0x1
17	71:68				0x0			0x0
18	75:72							
19	79:76							
20	83:80							
21	87:84							
22	91:88							
23	95:92							
24	99:96							
25	103:100							
26	107:104							
27	111:108							
28	115:112							
29	119:116							
30	123:120							
31	127:124							
32	131:128							
33	135:132							
34	139:136							
35	143:140							
36	147:144							
37	151:148							
38	155:152							
39	159:156							
40	163:160							
41	167:164							
42	171:168							
43	175:172							
44	179:176							
45	183:180							

Figure 19: Nibble order (2 of 5)

Note: all data represented below is written in "reading order", but transmitted in increasing nibble # order.

nibble #	bits	MB 0x35   C8 0x21   D64 0x1234567 0x01234567	Maintenance Packet, Freq = 416.67MHz	Color Coding	opcodes	opcode related value	master	channel	user data	time stamp
0	3:0	MB	0x1		0xF	D8	0x4	D16	0x5	D32 0x1234567
1	7:4	master	0x5		0xF	data	0x5		0xD	0x7
2	11:8		0x3		0xF		0xA		0xC	0x6
3	15:12	C8	0x3		0xF	NULL (pad)	0x0		0xB	0x5
4	19:16	channel	0x1		0xF				0xA	0x4
5	23:20		0x2		0xF				NULL (pad)	0x3
6	27:24	D64	0x7		0xF					0x2
7	31:28		0x7		0xF					0x1
8	35:32		0x6		0xF					0x0
9	39:36		0x5		0xF					NULL (pad)
10	43:40		0x4	ASYNC	0xF					
11	47:44		0x3		0xF					
12	51:48		0x2		0xF					
13	55:52		0x1		0xF					
14	59:56		0x0		0xF					
15	63:60		0x7		0xF					
16	67:64		0x6		0xF					
17	71:68		0x5		0xF					
18	75:72		0x4		0xF					
19	79:76		0x3		0xF					
20	83:80		0x2		0xF					
21	87:84		0x1		0xF					
22	91:88		0x0		0xF					
23	95:92	NULL (pad)	0x0	VERSION	0x0					
24	99:96				0x0					
25	103:100				0x3					
26	107:104				0xF					
27	111:108				0x0					
28	115:112				0x8					
29	119:116				0xA					
30	123:120				0x2					
31	127:124			FREQ	0x4					
32	131:128				0xD					
33	135:132				0x5					
34	139:136				0xD					
35	143:140				0x8					
36	147:144				0x1					
37	151:148				NULL (pad)	0x0				
38	155:152									
39	159:156									
40	163:160									
41	167:164									
42	171:168									
43	175:172									
44	179:176									
45	183:180									

Figure 20: Nibble order (3 of 5)

*Note: all data represented below is written in "reading order", but transmitted in increasing nibble # order.*

nibble #	bits	Color Coding	opcodes	opcode related value	master	channel	user data	time stamp
	FLAG_TS, ts=0xFEDCBA98765432				USER_TS		USER_TS	
10		FLAG_TS, ts=0x3579		USER, data = 0x0123456789ABCDEF	Master=0x4321	Channel=0x2	data = 0x123456789ABCDEF	
0	3:0	FLAG_TS	0xE		M16	0xF		0xF
1	7:4	ts size	0xE			0x1	USER_TS	0x0
2	11:8		0x0			0x1		0x3
3	15:12		0x1			0x2	USER size	0xF
4	19:16		0x2		master	0x3		0xF
5	23:20		0x3			0x4		0xE
6	27:24		0x4		C8	0x3		0xD
7	31:28		0x5			0x0		0xC
8	35:32		0x6		channel	0x2		0xB
9	39:36		0x7			0xA		0xA
10	43:40		0x8		USER_TS	0x0		0x9
11	47:44		0x9			0x3		0x8
12	51:48		0xA			0xF		0x7
13	55:52		0xB			0x6		0x6
14	59:56		0xC			0x5		0x5
15	63:60		0xD			0x4		0x4
16	67:64		0xE			0x3		0x3
17	71:68		0xF			0x2		0x2
18	75:72					0x1		0x1
19	79:76					0x0		0x0
20	83:80				data	0x8	TS size	0xE
21	87:84					0x7		0x0
22	91:88					0x6		0x2
23	95:92					0x5		0x4
24	99:96					0x4		0x6
25	103:100					0x3		0x8
26	107:104					0x2		0xA
27	111:108					0x1		0xC
28	115:112					0x0		0xE
29	119:116				TS size	0xE		0x1
30	123:120					0x0		0x3
31	127:124					0x2		0x5
32	131:128					0x4		0x7
33	135:132					0x6		0x9
34	139:136					0x8		0xB
35	143:140					0xA		0xD
36	147:144					0xC		0xF
37	151:148				time stamp	0xE	NULL (pad)	0x0
38	155:152					0x1		
39	159:156					0x3		
40	163:160					0x5		
41	167:164					0x7		
42	171:168					0x9		
43	175:172					0xB		
44	179:176					0xD		
45	183:180					0xF		

Figure 21: Nibble order (4 of 5)

Note: all data represented below is written in "reading order", but transmitted in increasing nibble # order.

nibble #	bits	Color Coding	opcodes	opcode related value	master	channel	user data	time stamp
	USER_TS data = 0x12 ts = 0x34			TRIG_TS data=0x23 ts = 0x9876	XSYNC data=0x12			XSYNC_TS data = 0x45 ts=0x3579a
0	3:0	0xF		0xF		0xF		0xF
1	7:4	USER_TS 0x0	TRIG	0x0	TRIG_TS 0x0	XSYNC 0x0	XSYNC_TS 0x0	0xB
2	11:8	0x3		0x6		0xA		
3	15:12	USER size 0x2		data 0x3	data 0x3	data 0x2		0x5
4	19:16	data 0x2		0x2		0x1	data 0x4	
5	23:20	0x1	NULL (pad)	0x0	TS size 0x4	NULL (pad) 0x0	TS size 0x5	
6	27:24	TS size 0x2			0x6			0xa
7	31:28	time stamp 0x4			0x7			0x9
8	35:32	0x3			0x8			0x7
9	39:36	NULL (pad) 0x0			0x9			0x5
10	43:40							0x3
11	47:44							NULL (pad) 0x0
12	51:48							
13	55:52							
14	59:56							
15	63:60							
16	67:64							
17	71:68							
18	75:72							
19	79:76							
20	83:80							
21	87:84							
22	91:88							
23	95:92							
24	99:96							
25	103:100							
26	107:104							
27	111:108							
28	115:112							
29	119:116							
30	123:120							
31	127:124							
32	131:128							
33	135:132							
34	139:136							
35	143:140							
36	147:144							
37	151:148							
38	155:152							
39	159:156							
40	163:160							
41	167:164							
42	171:168							
43	175:172							
44	179:176							
45	183:180							

Figure 22: Nibble order (5 of 5)

Note: all data represented below is written in "reading order", but transmitted in increasing nibble # order.

nibble #	bits	GERR data = 0x01		D64TS data = 0x01234567 89ABCDEF	Color Coding	opcode opcodes	opcode related value	master	channel	user data	time stamp	TIME_TS	TIME_TS
0	3:0	GERR 0xF		D64TS 0xF		FLAG	0xF	M16	0xF	TIME_TS 0xF	ts_other[63:0] = 0x8900112233445566	ts_other[23:0] = 0xAB5678	
1	7:4	GERR 0x2		D64TS 0x7		FLAG	0xE		0x1	TIME_TS 0x0		TIME_TS 0x0	
2	11:8	data 0x1		data	data	0xF		master	0x4	0x5		0x5	
3	15:12					0xE			0x3	clock_doma 0x1	clock_doma 0x1		
4	19:16					0xD			0x2	other_ts_for 0x4	other_ts_for 0x4		
5	23:20					0xC			0x1	TS_Other_size 0xE	TS_Other_size 0x6		
6	27:24					0xB			C8 0x3	0x6		0x8	
7	31:28					0xA			channel 0x6	0x6		0x7	
8	35:32					0x9			0x5	0x5		0x6	
9	39:36					0x8			D64TS 0xF	0x5	TS_Other 0x5	0x5	
10	43:40					0x7			0x7	0x4		0xB	
11	47:44					0x6			0xF	0x4		0xA	
12	51:48					0x5			0xE	0x3	TS size 0x5		
13	55:52					0x4			0xD	0x3		0xE	
14	59:56					0x3			0xC	0x2		0xC	
15	63:60					0x2			0xB	0x2	time stamp 0xA		
16	67:64					0x1			0xA	0x1		0x9	
17	71:68					0x0			0x9	0x1		0x7	
18	75:72			time stamp	time stamp	TS size 0x6		data	0x8	0x0			
19	79:76					0xA			0x7	0x0			
20	83:80					0x9			0x6	0x9			
21	87:84					0x7			0x5	0x8			
22	91:88					0x5			0x4	TS size 0xE			
23	95:92					0x3			0x3	0xF			
24	99:96					0x1			0x2	0xE			
25	103:100					NULL (pad) 0x0			0x1	0x1	0xD		
26	107:104								0x0	time stamp 0xC			
27	111:108								TS size 0xE	0x0	0xC		
28	115:112			time stamp	time stamp			time stamp	0x3	0xB			
29	119:116								0x6	0xA			
30	123:120								X04	0x9			
31	127:124								X09	0x8			
32	131:128								X08	0x7			
33	135:132								X01	0x6			
34	139:136								X06	0x5			
35	143:140								X02	0x4			
36	147:144								X03	0x3			
37	151:148								X04	0x2			
38	155:152								X07	0x1			
39	159:156								0XF	0x0			
40	163:160								0X2				
41	167:164								0XC				
42	171:168								0X9				
43	175:172								0XA				
44	179:176												
45	183:180												

**Figure 23: Encoder Nibble Ordering in Memory**

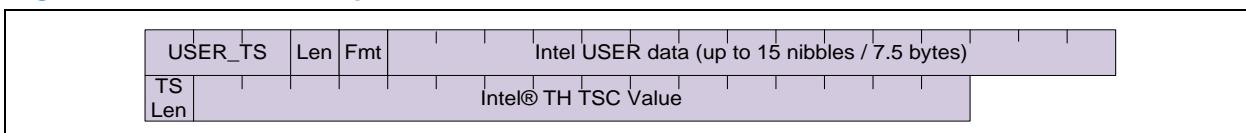
Bit #	7	6	5	4	3	2	1	0
Byte 5	0	0	0	0	1	0	1	0
Byte 4	0	1	0	1	0	1	0	0
Byte 3	0	1	1	1	0	1	0	0
Byte 2	0	0	1	1	1	0	1	0
Byte 1	0	1	0	1	0	0	1	0
Byte 0	0	0	0	1	1	1	1	1

Each encoder in the GTH has the option to create or suppress NULL packets. For output ports that require constant data, NULL packet generation should be enabled; for output ports that do not require constant data, or where “filler” packets are not useful (such as system memory), NULL packet generation should be disabled.

#### 6.2.3.1 Intel Trace Hub hardware-specific USER packet format

The MIPI STPv2.1 protocol allows for a user-defined packet format, called USER or USER\_TS, shown in Figure 24. However, the STPv2.1 standard does not specify the contents of the data packet. Since the Intel Trace Hub uses this packet format for a variety of reasons and purposes, and since it is important that both hardware and software easily recognize the various USER packets that the Intel Trace Hub can create, we herein define the standard for the Intel Trace Hub hardware-specific use of the USER packet.

The USER packet format does not apply to software trace sources, which can emit USER packets by writing to the appropriate MMIO location within the STH. These hardware-generated USER packets are easily distinguished from USER packets created by software because the hardware-generated packets are sent to Master 0 Channel 0. No software-generated packets can be sent to this master/channel combination.

**Figure 24: STPv2.1 USER packet format**


The STPv2.1 USER packet is used to convey two different types of information, or Intel-specific packet types: a Start of Data Loss packet, and an End of Data Loss packet. Exactly

when these packets are inserted is the responsibility of each individual trace source. Consult the respective trace source chapter (e.g., STH Chapter) for more information.

#### 6.2.3.1.1 Start of Data Loss Packet

The Start of Data Loss packet is a USER\_TS packet sent by the STH when it starts dropping incoming data (i.e., when drop mode is on/active). This packet is a USER\_TS packet with Intel Trace Hub defined USER format type 1, as shown in Figure 25.

**Figure 25: Start of Data Loss Packet**

Start of Data Loss (STRT_DLOSS)	USER_TS			Len	Fmt	Intel® TH TSC Value											
	F	0	3	0	1												
	TS	Len															
	E	2	3	9	5	C	D	2	A	1	7	F	E	3	7	2	1

#### 6.2.3.1.2 End of Data Loss Packet

The End of Data Loss packet is a USER\_TS packet sent by the STH when it resumes normal operation after dropping incoming data (i.e., after drop mode is ended and normal (lossless) operation resumes). This packet is a USER\_TS packet with Intel Trace Hub defined USER format type 2, as shown in Figure 26. The End of Data Loss packet indicates the number of packets that were dropped by the STH (not including any TRIG packets that would have been created as a result of a trig\_in received from the CTS). It also includes a time stamp so that the downstream software knows when the data loss period ended.

**Figure 26: End of Data Loss Packet**

End of Data Loss (END_DLOSS)	USER_TS			Len	Fmt	NumPkts Lost	Intel® TH TSC Value											
	F	0	3	4	2	0	2	3	5									
	TS	Len																
	E	8	9	9	5	C	D	2	7	1	7	F	E	3	7	2	9	

#### 6.2.4 Byte Packing Buffer

The Byte Packing Buffer's function is to map, or pack, the variable number of bytes from the STPv2.1 encoder to a fixed-width output with as few NULL packets as possible.

The BPB also has the ability to drop STP-encoded packets, if specified by the GTHOPT.PnDRP bit. If the GTHOPT.PnDRP bit is asserted, the trace destination is treated as if it is always ready to receive data. If the downstream block is not ready to receive data, the data will be permanently lost. If GTHOPT.PnDRP bit is deasserted, the trace destination will not be presumed to always be ready for data, and the data transfer protocol to the trace destination will be followed. If the BPB fills up, it will assert its *full* signal, thereby backpressuring the STP encoder and GTH Data Switch.



If NULL packet generation is enabled (by GTHOPT.PnNULL bit), then, for every clock cycle in which the BPB does not have a row of data to send to the trace destination, one byte of NULL packets (two STP NULL packets) will be sent to the trace destination.

Finally, the BPB can be manually flushed using the GTHOPT.PnFLUSH register bit. Manual flushing in this manner is not needed during normal operation, as the hardware initiates BPB Flush operations when they are needed. If the collection of trace data is under “manual control” (such as when using the StoreEnOvrd bits), then the BPB must be flushed at the end of tracing using the PnFLUSH bit to ensure all trace data is flushed to the destination.

### 6.2.5 STP Maintenance Unit

The STP Maintenance Unit (SMU) is responsible for all aspects of maintaining the STPv2.1 link at the protocol level, per the MIPI STPv2.1 specification. Requirements include:

- periodically sending out ASYNC, VERSION, and FREQ packets
- Sending ASYNC, VERSION and FREQ packets at the beginning of a trace, including whenever a destination port is reset or retrained.

The second requirement is met by sending ASYNC, VERSION, and FREQ packets at the beginning of a trace, defined as the assertion of a *StoreEn* signal. To ease implementation, the SMU is assigned to trace source #0 on the GTH data switch, and the TSCU is trace source #1. This way, the arbiter will start (out of reset) with trace source #0, automatically transition to source #1 after the maintenance packet is sent, and then move on to the data sources. With this approach, the requirements of the MIPI STPv2.1 specification are guaranteed by the architecture.

The Intel Trace Hub architecture implements one SMU per output queue so that it may send maintenance packets to each output queue at an optimal rate for that output queue. For example, system memory may be capable of 5GB/s of user data (STPv2.1 data), and will likely have maintenance packets inserted every 5us or so. This would be far too often for a USB3 interface, which would need maintenance packets about once every 40us. Each SMU is assigned to the same master number, Master 0, as all “administrative” packets are assigned to this master number in the Intel Trace Hub architecture. The Master number is used only by the downstream MIPI STPv2.1 encoder to set the current Master (and channel) to zero, as required by the MIPI specification. That is, the master number is not used by the arbiter, as there is one SMU per arbiter and output queue.

The SMU also has an *Enable* input signal to control whether it sends trace data (maintenance packets) or not. This is done so that the SMU does not send maintenance packets when all trace sources for the respective output port are turned off (see also section 6.2.10 Automatic Trace Source Enable/Disable). Each SMU’s *Enable* signal is chosen according to its output port. In this way, each SMU is automatically enabled and disabled based on whether there are active trace sources for its respective destination.

The SMU timer is started upon first *get* received from the arbiter, after *portReset* is deasserted. Thereafter, the SMU will decrement as long as its *enable* is asserted and *portReset* is deasserted. If either *enable* or *portReset* is asserted, then the SMU timer is reset to its starting value, and the SMU sends no packets until *enable* is again asserted and *portReset* is deasserted.

When the Maintenance Timer expires, it asserts its “maintenance packet request” output, indicating a request to send a maintenance packet. When the Data Switch arbiter asserts the *get* signal to the SMU Input Buffer, the maintenance packet data set will be driven to

the Input Buffer (and Data Switch, since the Input Buffer for the SMU is a zero-depth buffer), and the latched maintenance packet request is cleared.

The other conditions that can cause the SMU to request to send maintenance packets are when a trace source for the SMU's destination is enabled (*StoreEn* assertion), and when a trace destination port becomes operational. When the *portReset* signal is deasserted, the SMU's maintenance packet insertion process is triggered.

The content of the SMU maintenance packet data is fixed for a given implementation, with the exception of the GTH\_FREQ field. The GTH\_FREQ field comes from the GTH\_FREQ register. The maintenance packet data and consists of three STPv2.1 packets, as shown in Table 6-2 below.

**Table 6-2: SMU Packet Data**

Packet Number	Packet type	Packet data	Comments
1	ASYNC	0xFF_FFFF FFFF_FFFF FFFF_FFF0	21 nibbles of 0xF, one nibble of 0x0
2	VERSION	0xF003	0x3 signifies STPv2NAT timestamp format
3	FREQ	0xF08{GTH_FREQ}	

**NOTES:**

1. Data is shown in network order—the same order as it is sent to the destination. That is, the first nibble, reading left to right, is the first nibble to be sent.

The GTH\_FREQ portion of the FREQ packet will depend on the GTH\_FREQ register value (which has a default value equal to the GTH\_FREQ synthesis parameter). In any case, the GTH\_FREQ data is the operating frequency, in Hertz, of the GTH, specified as a 32-bit binary value. For example, 400MHz is 0x17D78400.

## 6.2.6 Low Power Path

The 2<sup>nd</sup> generation of Intel Trace Hub architecture adds a dedicated tracing path that can be powered by a separate power supply in the SOC. This Low Power Path (LPP) logic has a relatively small footprint and can be used for a low-power-mode tracing path from trace sources on DTF to PTI or BSSB (depending on configuration). This same LPP also provides a trace path in full power mode from any trace source to either PTI or BSSB through the DCI bridge. A high-level diagram of the Global Trace Hub, showing the LPP logic blocks, is shown in Figure 16.

LPP supports seamless tracing into and out of system low-power states by providing a single-bit-control to switch from full-power mode (e.g. tracing to system memory), to low-power mode (e.g. tracing to PTI or BSSB). Inherent to the architecture is support for a continuous trace data sink for sources that are always-on on DTF. That is, the hardware switching capability provided by the LPP makes any low power state entry/exit invisible to DTF trace sources.

**Note:** When LPP is switched to its Low Power Mode (LPM), all trace data is routed



automatically to either PTI or BSSB (according to the LPP\_CTL.DEST bit) by the LPP hardware. That is, trace data is sent automatically by hardware to LPP independent of the routing tables (i.e. SWDEST and Master Enable controls are not applicable in this mode).

The LPP Encoder & Output Unit also includes the LPP\_CTL register. This register contains the control and status bits for the LPP logic. If the SOC supports the LPP logic on a separate ("low power") power rail, then while the main power is off, the LPPCTRL register is only accessible through TAP using the REQMSG and RSPMSG TAP instructions. When main power is on, the LPP\_CTL register is accessible as a standard MMIO space register.

The only Store Enable signal applicable for low power mode is the one for the NDB. In "low power mode", this signal will be asserted when the LPMEN and LPMSTEN bits in the LPP\_CTL register are set. In full power mode, the NDB store enable signal will be asserted based on the Store Enable from the trigger unit (see chapter 10 ).

If configured to trace to PTI, LPP can support a bandwidth of up to 2bytes/clock, while when configured to trace to an external bridge, it can support a bandwidth of up to 1byte/clock.

For further details on the PTI port, please refer to chapter 11 .

### 6.2.7 DTF Input

Some trace sources an in SOC transmit their trace data to the Intel Trace Hub using a special-purpose hardware connection called DTF. This DTF uses MIPI STP semantics (e.g., Master and Channel) to transmit the data to the GTH. Please see Chapter 5 for details.

### 6.2.8 DCI Trace Handler

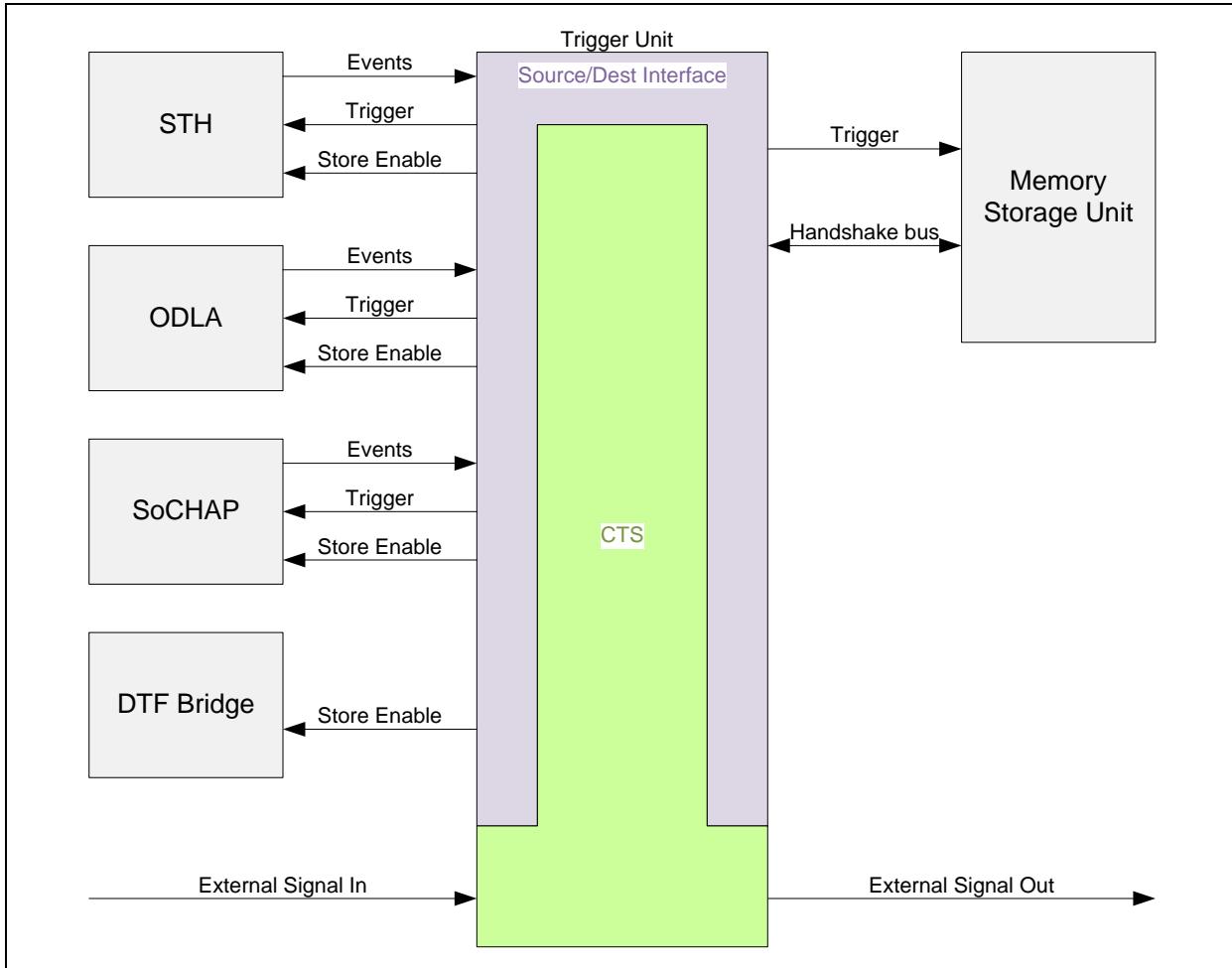
The Intel Trace Hub DCI Trace Handler (Intel TH DCI TH) is a trace destination, or output port. While it is a separate output port in concept, it shares output port #0 with the MSU, as it shares a number of functional requirements with the MSU. For example, both the MSU and Intel TH DCI TH send data over the primary bus, and must buffer a certain amount of data before sending it to its final destination. For efficiency and area savings, the MSU and DCI Trace Handler have a single set of memory buffers and output logic between them.

For more information, see the DCI Trace Handler chapter.

### 6.2.9 Trigger Unit

The triggering capability in the Intel Trace Hub takes "events" and "signals" as input, processes them, and takes actions based on sequences or combinations of these events. To state it slightly differently, events and signals are Trigger Unit state machine inputs, with each state machine state/clause taking separate, user-specified actions such as trigger, start store, stop store, start timer, and so on. The Trigger Unit takes input events from inside the Intel Trace Hub and signals from outside the Intel Trace Hub. Note that there is only one trigger output from the Trigger Unit, and it is routed simultaneously to all destinations (trace sources and MSU).

See the Common Trigger Sequencer in chapter 10 for more information on the trigger sequencer block (CTS).

**Figure 27: Trigger Unit**


#### 6.2.9.1 Source/Dest Interface Shim

As shown in Figure 27, the Source/Dest Interface shim is located between the CTS and the trace sources, and between the CTS and the MSU. The shim's purpose is to bridge the *storeQual* signals from the CTS to the *storeEn* signals to the sources, and also to implement the "block and drain" functionality required for the MSU's multi-window functionality.

The mapping of *storeQual* to *storeEn* signals is given in Table 6-1. Since *Stor\_Qual[0]* is reserved for the special function of *captureDone*, and since the SMU and TSCU do not need a *storeEn* driven from the CTS, there is a rather unique mapping of *Stor\_Qual* to *storeEn* signals, as shown in the table below.

**Table 6-3: Store\_Qual to Trace Source Mapping**

<b>Stor_Qual #</b>	<b>Destination</b>
0	CaptureDone
1	SoCHAP
2	ODLA

Stor_Qual #	Destination
3	STH
4	DTF Bridge

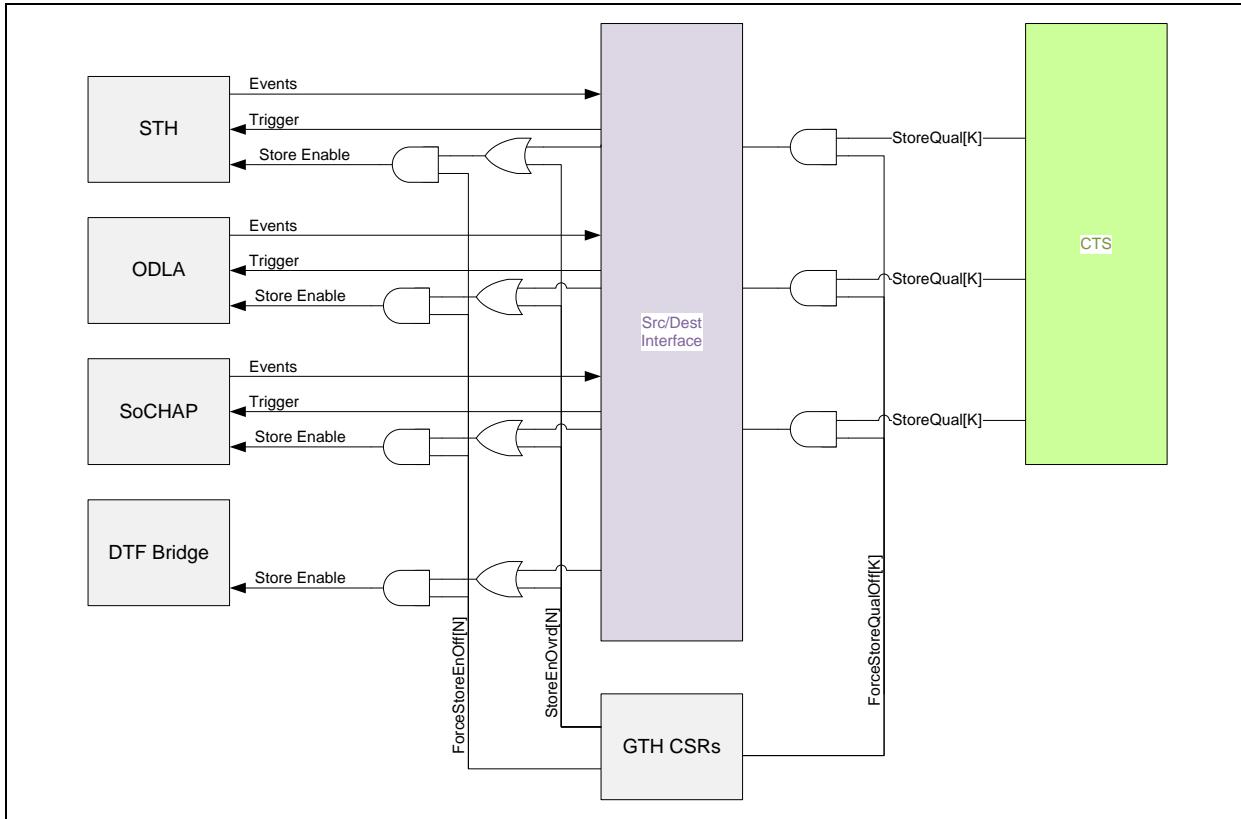
The multi-window support requires the shim to “interrupt” (de-assert temporarily) the store enable to the trace sources during a window switch, such that the entire pipeline can drain and switch to the next window. This “interrupting” is only enabled for those trace sources that have their “mode” bit set in the Source Control Register. Further, the “interrupting” of the *storeEn* signals will only happen if the trigger sequencer has reached its “trigger” state/condition. That is, if the sequencer has not yet asserted the *trigger* output, then the de-assertion of *storeQual* signal(s) will not result in a window switch operation; it will only serve to pause trace capture for the duration of *storeQual* deassertion.

When completely stopping trace storage in multi-buffer mode, the *captureDone* signal must be asserted simultaneously with the deassertion of the *storeQual[N]* signals. If the *captureDone* assertion is delayed (even for 1 clock cycle), and if the *Trigger* signal has been asserted, then (for store-to-memory) a window switch will be performed before tracing is stopped and all data is flushed.

#### 6.2.9.1.1 Manual Override

The Source/Dest Interface shim also implements a “manual override” feature to allow very simple trace capture. The “manual override” mode is the equivalent of setting the trigger specification to “If Anything, start storing immediately”, and “stop storing when I tell you.” This is accomplished by software setting the *storeEn* override bits (GTH register SCR) for each trace source to be enabled, and starts the flow of the trace data. Figure 28 shows the details of the manual override logic.

Figure 28: Trigger Unit Manual Override Logic



Stopping the flow of trace data is most easily accomplished by software setting the appropriate ForceStoreEnOff bits in the SCR2 register. Using this method also allows software to manually assert the "Capture Done" signal, which will cause all data in the Intel Trace Hub pipeline and buffers to be flushed to the destination.

#### 6.2.9.1.2 Event Inputs

The CTS has 16 inputs for events; 14 are used by the Intel Trace Hub for its sources, as shown in Table 6-4. The two remaining event inputs are made available to the SOC for its usage (they are "external" to the Intel TH). Please refer to your product-specific documentation for information regarding the usage of these "External" event inputs.

Table 6-4: Trigger Unit Event Connections

Event #	Source	Event #	Source
0	VER* 0	8	SOCHAP 0
1	VER 1	9	SOCHAP 1
2	VER 2	10	STH 0
3	VER 3	11	STH 1
4	VER 4	12	STH 2

5	VER 5	13	STH 3
6	VER 6 or dataXfer	14	External
7	VER 7 or dataXfer	15	External

\* NOTE: VER = VIS Event Recognizer

The Trigger Unit makes four each Signal\_In and Signal\_Out signals available to the SoC for use according to the SoC usage needs. Please refer to your product-specific documentation for information on how these signals are connected.

#### 6.2.9.1.3 Events 6, 7

Events 6 and 7 are unique starting with the 2<sup>nd</sup> generation of the Intel Trace Hub, in that they offer the option to provide an indication of the number of bytes transferred from the BPP to the trace destination (e.g., MSC0, HTI, etc). This is useful for counting total bytes post-trigger, such that the trigger unit can be configured to allow only so much storage after the trigger, as is common in other trace capture solutions. When Event 6 or 7 is configured to indicate the number of bytes transferred, the signal will be asserted once for every "chunk" of data transferred. The size of each "chunk" is shown in the table below

Destination	Chunk size
System DRAM	16 bytes
DbC.Trace	16 bytes
HTI	8 bytes
PTI	PTI bus width, or 1 byte if External Bridge mode

#### 6.2.10 Automatic Trace Source Enable/Disable

The SMU, like all other trace sources, has an *Enable* input signal to control whether it sends its trace data (maintenance packets) or not. There is, however, no *StoreQual* signal from the trigger sequencer for the SMUs. Instead, the Intel Trace Hub automatically controls whether the SMU *Enable* signals are asserted or deasserted.

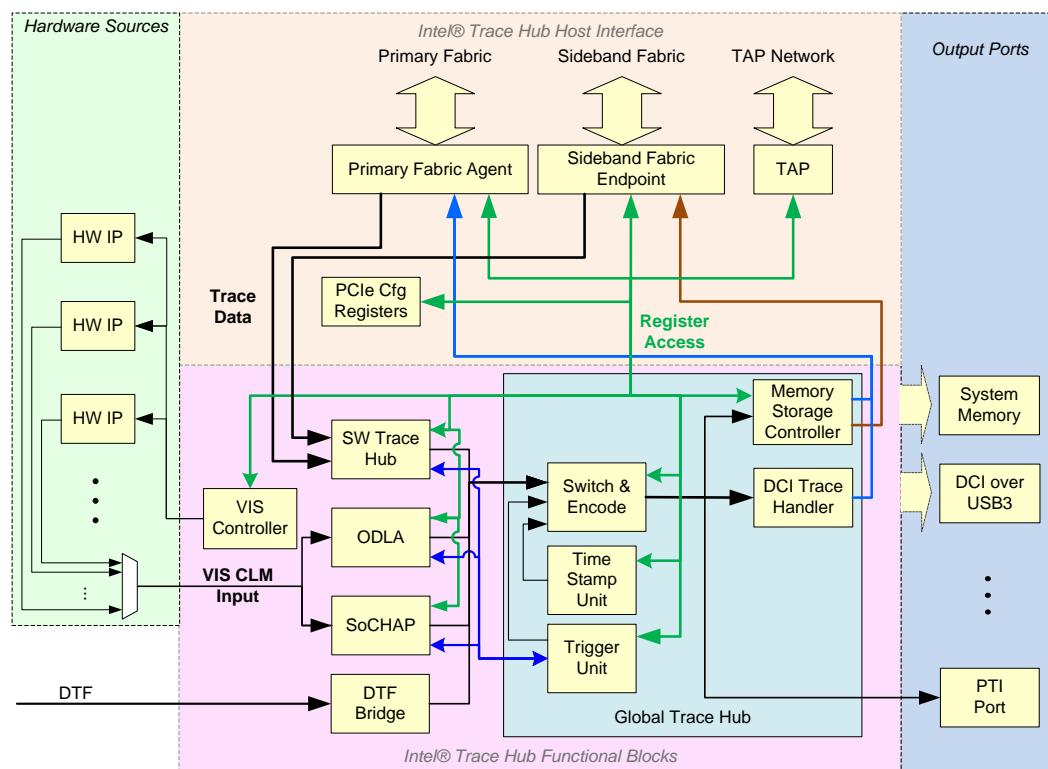
## 7 Timestamp Counter Unit

The TSCU provides the common timebase function for the Intel Trace Hub (Intel TH) so that captured Intel TH data can be reconstructed and correlated with other trace data generated on the same platform. As shown in Figure 29, the TSCU is part of the Global Trace Hub.

The TSCU provides a 64-bit timestamp counter that is shared throughout the debug and trace system architecture. This timestamp is based on the Always-Running Timer (ART) and is scaled-up to the Intel TH clock rate to achieve a high-resolution timestamp. This timestamp is used for marking native MIPI STPv2 trace source packet's timestamp fields. Additionally, ODLA, a non-native MIPI STPv2 trace source, is directly using this timestamp and storing values, as needed, in the TSC packets it creates within internal ODLA logic.

Intel Processor Trace is a non-native STPv2 trace source, but embeds timestamp information into its trace data using the same coarse time base (the ART time domain) as the Intel TH and the rest of the debug & trace system, so its data is easily and directly referenced to Intel TH timestamp values. The TSCU employs a method for synchronizing and maintaining a local copy of the ART called a Common Timer Copy (CTC).

**Figure 29: SoC timestamp correlation architecture – system view**



The TSCU provides the following features and capabilities:

- Provides a free running 64-bit timestamp value to all Intel TH logic based on the CTC and scaled-up to the Intel TH clock rate.
- All packets timestamped by the Intel TH are based on the ART value, allowing for straight-forward time correlation with all other trace sources using ART.



## 7.1 Intel Trace Hub Timestamp

The TSCU maintains its own local copy of the ART timestamp counter. It does this by downloading a copy of the ART after power-up or reset, and thereafter maintaining its own copy of the ART/CTC. All trace sources are correlated to the timestamp counter by being explicitly timestamped at the trace source with the current Intel TH timestamp value, or via post-processing software that correlates an embedded common SoC derived timestamp value (ART) to an Intel TH timestamp value (Intel Processor Trace, STH). The Intel TH timestamp counter is reset by a powergood reset only.

## 7.2 CTC Tracking Unit

The TSCU maintains its own local copy of the ART timestamp counter using the CTC Tracking unit inside the TSCU. It does this by downloading a copy of the ART after power-up or reset, and thereafter maintaining its own copy of the ART/CTC using the same clock as all other ART-based timestamp counters in the system. The SoC's CPUs use the ART as the reference for generating their local TSC values after clock crossing and tracking/scaling logic. Each core maintains its own local, fast-domain TSC that is synchronized to each other, even after returning from a powered down state, so that a coherent timestamp is available for reference. The core's local TSC value is what is returned when a RDTSC instruction is executed, for example. The ART is also the base for the timestamp values embedded in Intel Processor Trace traces.

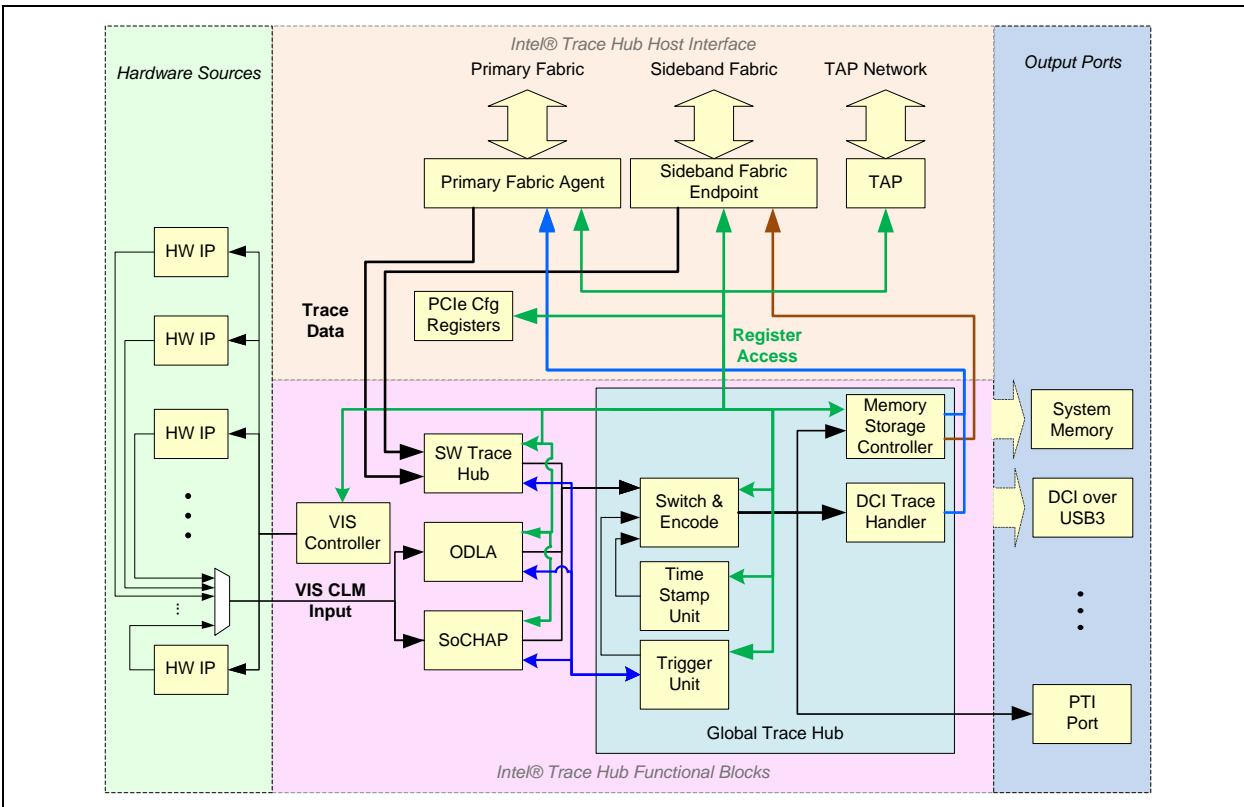
Intel TH tracks the ART value so that, for example, Intel Processor Trace traces can be time correlated with other trace sources.

Because the ART value is downloaded and synchronized after every power-up and/or reset exit sequence, and this synchronization takes time, it is possible that trace data is collected before the Intel TH timestamp is sync'd with the ART value. Data that is collected before the sync is complete will be timestamped starting from time=0. Once the synchronization is complete, trace data will be timestamped with the correct ART-based value. When this happens, time may appear to suddenly "jump forward" by a large amount. This is normal and expected behavior of the Intel TH.

## 8 Memory Storage Unit

This section defines the high level architecture of the Intel Trace Hub Memory Storage Unit and its components. Figure 30 shows the MSU in the context of the Intel TH block.

**Figure 30: Memory Storage Unit in Intel Trace Hub**



### 8.1 Functional Description

#### 8.1.1 MSU Overview

The MSU is responsible for taking trace data from the Byte Packing Buffers, temporarily storing it in its own trace buffers, and routing the data to memory, or a memory-mapped IO destination.

The MSU is comprised of two Memory Storage Controllers and an MSU arbiter to arbitrate and control their access to the primary fabric. Two MSCs are required in order to meet the requirements of the “windowed” trace capture use cases supported by the Intel Trace Hub.

To ensure good performance, the MSCs wait until a full cache line of data has accumulated in the MTBs before writing that data to system memory. Only when in a “flush” condition (e.g. a window switch, or an end-of-tracing condition), or when specifically configured via the MSCnCTL.MSCnLEN register field does an MSC write less than a cache line of data.

### 8.1.2 MSU Operational Modes

The Memory Storage Unit operates in four modes. Single block mode (or CSR-driven mode), multi-block mode (or linked-list mode), DCI trace mode (or Intel TH DCI trace handler mode), and Internal Buffer mode. All operating modes have an initialization stage, a storage stage, and a retrieval stage. In the initialization stage, software initializes the CSRs and memory blocks (if applicable) with information required by the hardware for the storage stage. When software is done initializing the MSU (i.e., one or both MSCs), it moves to the storage stage by configuring the trigger unit according to the needs of the user, and then setting the CTS control register Sequencer\_Enable bit. Hardware performs the tracing and storage operations, and then sends an interrupt to software (see `msu_interrupt` description) to indicate it is finished (see `CaptureDone` description), and data is available.

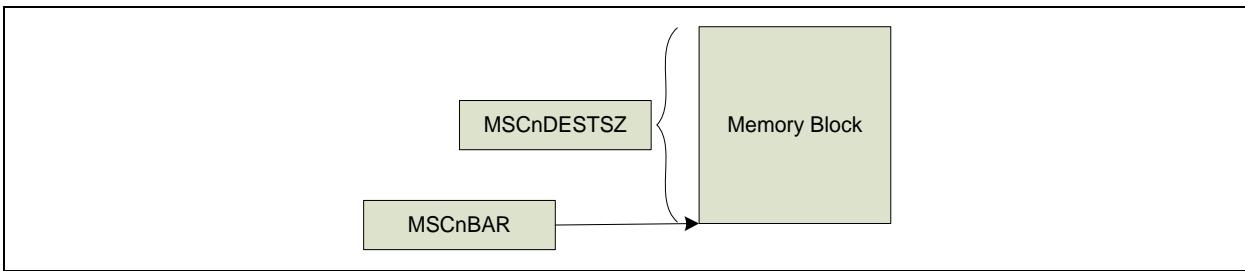
Under normal conditions, the MSCs are reset at power on. The MSCs support multiple captures in different operating modes, however there is no soft reset mechanism implemented in the MSCs to aid in this transition. For this case, the MSC requires the current capture to complete properly with no residual data, then the MSCnEN must be de-asserted, then the MSC must be re-initialized and finally re-started in the new mode by setting MSCnEN.

The four operational modes and the memory block header formats are described in the followed sections.

#### 8.1.2.1 Single Block or CSR-Driven Mode

The single block mode, also known as the CSR-driven mode, shown in Figure 31 is the simple and straight-forward mode. The user programs the MSC CSRs with the starting memory address, size, and wrap settings. This allows the user to store trace data to memory with a minimum of setup information. In single block (CSR-driven) mode, the MSU operation is simplified and is not responsible for performing a memory read and a memory write of header information since this information exists in CSRs. The single-block mode requires the user to dedicate a portion of memory for the trace buffer that is not visible to the OS.

Figure 31: MSC Single Block or CSR-Driven Mode



**NOTE:** If wrapping is not enabled for Single Block mode, then when the memory block/buffer is filled, the MSC will stop writing data to memory, and will apply upstream backpressure. Because the MSC will backpressure the upstream pipeline, the pipeLineEmpty bits will never be asserted, because the data pipeline cannot be emptied. Software must account for this case when stopping tracing operations.



### 8.1.2.2 Multi-Block or Linked-List Mode

The multi-block or linked-list mode allows the user to store trace data to system memory while the system is under the control of an OS. For this mode, one or more blocks of memory is allocated, either by BIOS, or by a process running under the operating system. A memory “block” in this context is a single, physically contiguous portion of memory. The same software (BIOS or OS) then initializes the memory block with header information describing the block and its relationship to other blocks; notably each block contains two addresses linking the block to neighboring blocks. This is similar to descriptor tables used in/by USB and ethernet drivers. Once the linked list is initialized, the software writes the starting address of the first memory block to the MSCnBAR registers. Upon setting MSCnEN, the MSC fetches the first descriptor in preparation for storing trace data to system memory.

### 8.1.2.3 DCI Trace Handler Mode

In the DCI trace handler mode, the two trace buffers in the MSU are utilized as a single combined buffer. In this mode, only trace destination #0 is valid (trace destination #1, or MSC #1, is an invalid destination, even though MSC1 must be set to DCI Trace Handler Mode).

The Intel TH DCI trace handler is responsible for the data once it is in the trace buffers. The DCI trace handler has two sub-modes of operation: DbC.Trace mode, and BSSB mode.

In DbC.Trace mode the two MSC's act as a memory resource for the DCI Trace Handler, which handles all the interaction with the DbC.Trace endpoint. The DCI trace handler waits for the trace buffer depth to reach a certain threshold and then programs the DBC (using memory write cycles) to perform a DMA transfer. Once programmed, the DBC starts the DMA transfer of the trace data and these transactions appear to the MSU as inbound memory read cycles, where the trace data is then transferred to the DbC.Trace endpoint.

In BSSB mode, the DCI trace handler initiates the transfer of data from the unified trace buffer as memory-mapped write commands to the DCI Bridge.

### 8.1.2.4 Internal Buffer mode

The Internal Buffer mode of operation is a mode where the trace buffers will capture trace data without transporting it out to main memory. This mode is useful for debug of a system where main memory is not available. Read transactions that target the trace buffers will always return a single a QWORD (64 bits) data quantity, and bursting is not supported. These reads are only supported in Internal Buffer mode and are ignored in other modes. Writing of the trace buffers from the primary or sideband fabric is not supported in any operational mode.

The wrap configuration bit will be used to determine if the MSU should wrap or halt trace capture at the end of the trace buffer.

**NOTE:** If wrapping is not enabled for Internal Buffer mode, then when the internal buffer is filled, the MSC will stop storing data to the internal buffer. Because the MSC will backpressure the upstream pipeline, the pipeLineEmpty bits will never be asserted, because the pipeline cannot be emptied. Software must account for this case when stopping tracing operations.

The reading of the trace buffers in internal buffer mode can be from the primary interface, sideband interface, or the TAP. Address bit 15 used to select between MTB0 and MTB1.

**NOTE:** Reading of the MTBs in internal buffer mode while trace data is being stored in the MTBs is not supported. There is currently no way to prevent the MTB from under-run (reading more data than is actually present). Furthermore, the WRAPSTAT will not function properly, due to the fact that reading from the MTB changes the read pointer value, which is used in the full and empty calculations.

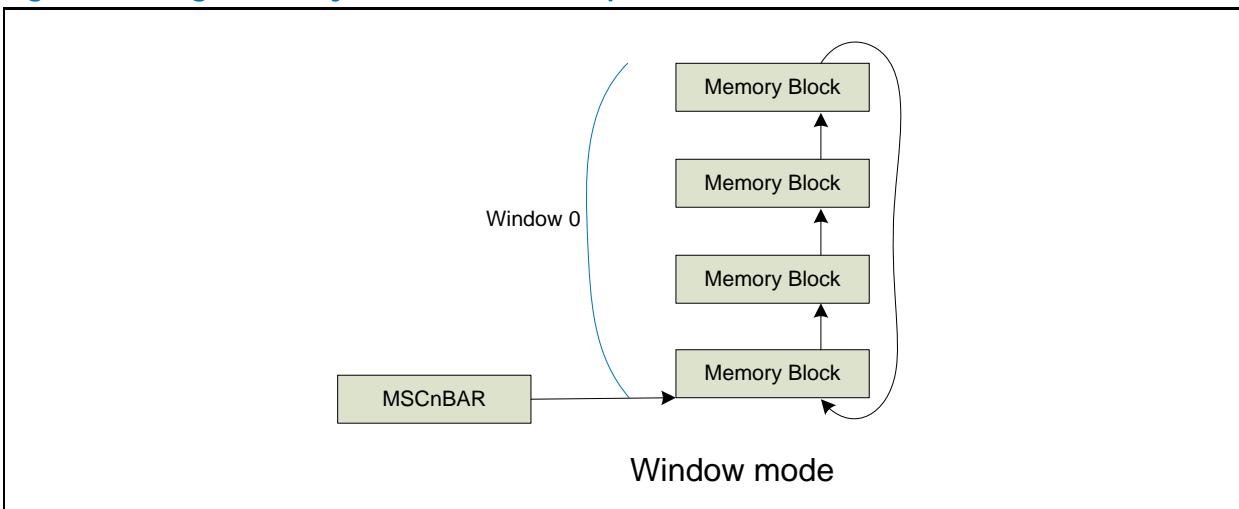
**NOTE:** When switching to internal buffer mode from another operational mode, some trace data will be lost in the "preload" buffer. This is because, in a "normal" operational mode (anything other than internal buffer mode), data from the MTB is loaded into a "preload" buffer so that it can be immediately sent to the primary bus when the it is granted to the Intel Trace Hub. The preload for each MSC buffer is one stage deep, and cannot be read. Thus, the first 8 bytes of trace data will not be available for extraction when switching from a normal operational mode to Internal Buffer mode.

### 8.1.3 Multi-Block Memory Windows

Memory blocks can be chained together through programming of the software header. A collection or chain of memory blocks, all with the same next window starting address is called a memory window.

Figure 32 gives a visual representation of the multi-block memory window concept.

**Figure 32: Single Memory Window with Multiple Blocks**

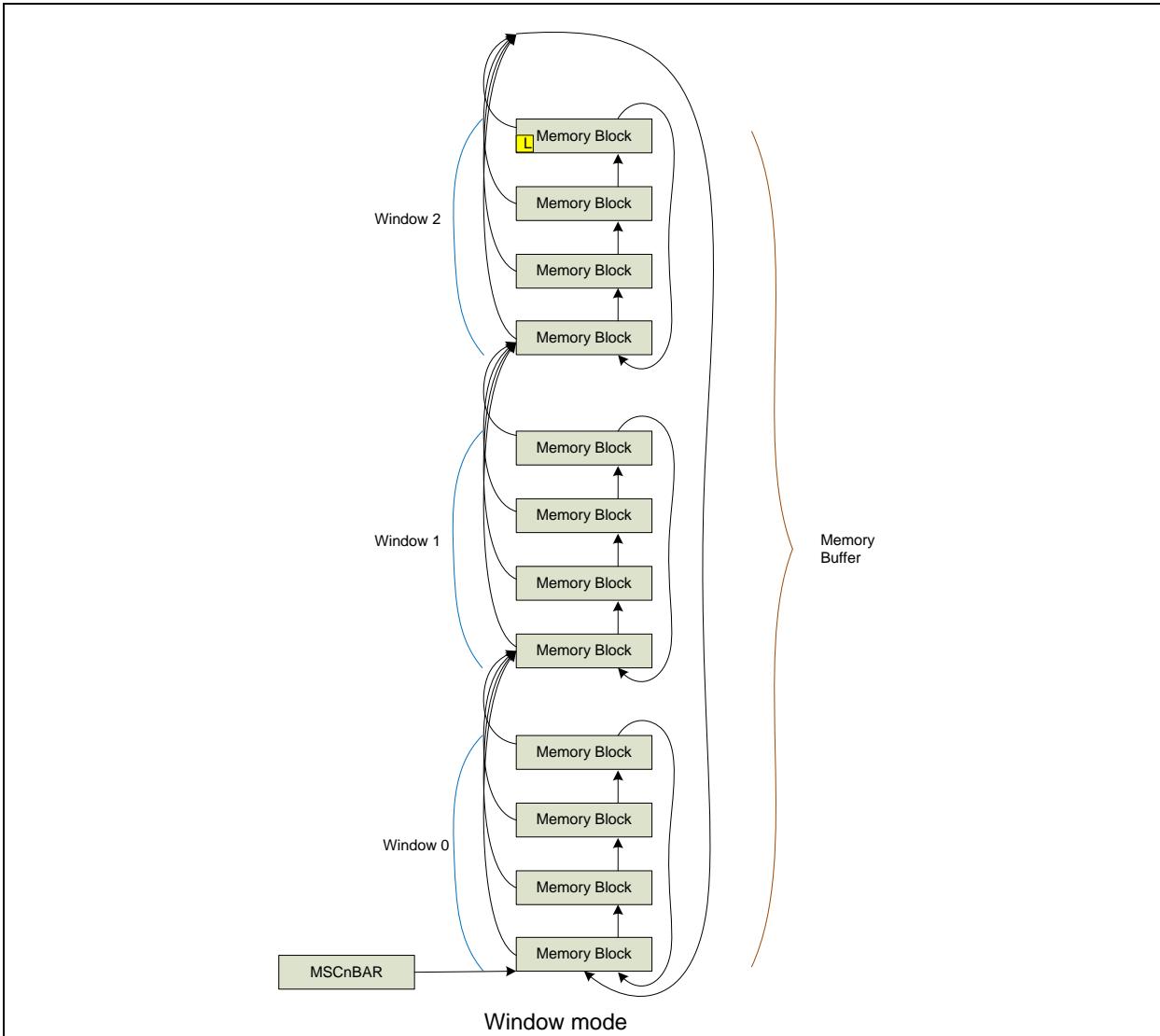


### 8.1.4 Multi-Window Memory Buffer

Multiple windows can be chained together through programming of the software header. The collection of memory windows together makes up the memory buffer for a given MSC. Multi-block windows are supported to enable the Intel TH to use OS-managed memory. Contemporary operating systems allocate memory blocks in increments of 4 kB, and don't guarantee delivery to any single block that is more than 4 kB, or to any multiple blocks that are contiguous; multi-block windows are, therefore, a necessity so that the MSC can store trace data in buffers larger than 4 kB.

Figure 33 gives a visual representation of the multiple chained windows concept.

Figure 33: Multiple Memory Windows



Repeated storage is the intended use case for multiple windows within a single memory buffer. That is, the user desires to store data before and/or after a given event (trigger), and to do this several times in a single capture session. To accomplish this, the Intel TH trigger unit is configured to assert the *storeEn* signal for one or more trace sources, and begin watching for the trigger event. During this time, the MSC is storing data into a single window (wrapping is always on in this mode). When the trigger condition is reached, the trigger unit begins a post-trigger fill period (the length of time of which is determined by a counter/timer). When the post-trigger fill period is reached, the trigger unit (as it was previously configured) de-asserts the *storeEn* signals to all trace sources sending data to the MSC(s) operating in multi-window mode. This de-assertion of the *storeEn* signals is interpreted by the MSU as a “switch window” command. Once all in-flight data is written to system DRAM, the MSC(s) switch to the next window (where they might, for example, begin writing to memory at MSCnNWSA, specified in the then-current memory block header).

### 8.1.5 Memory Block Header Description

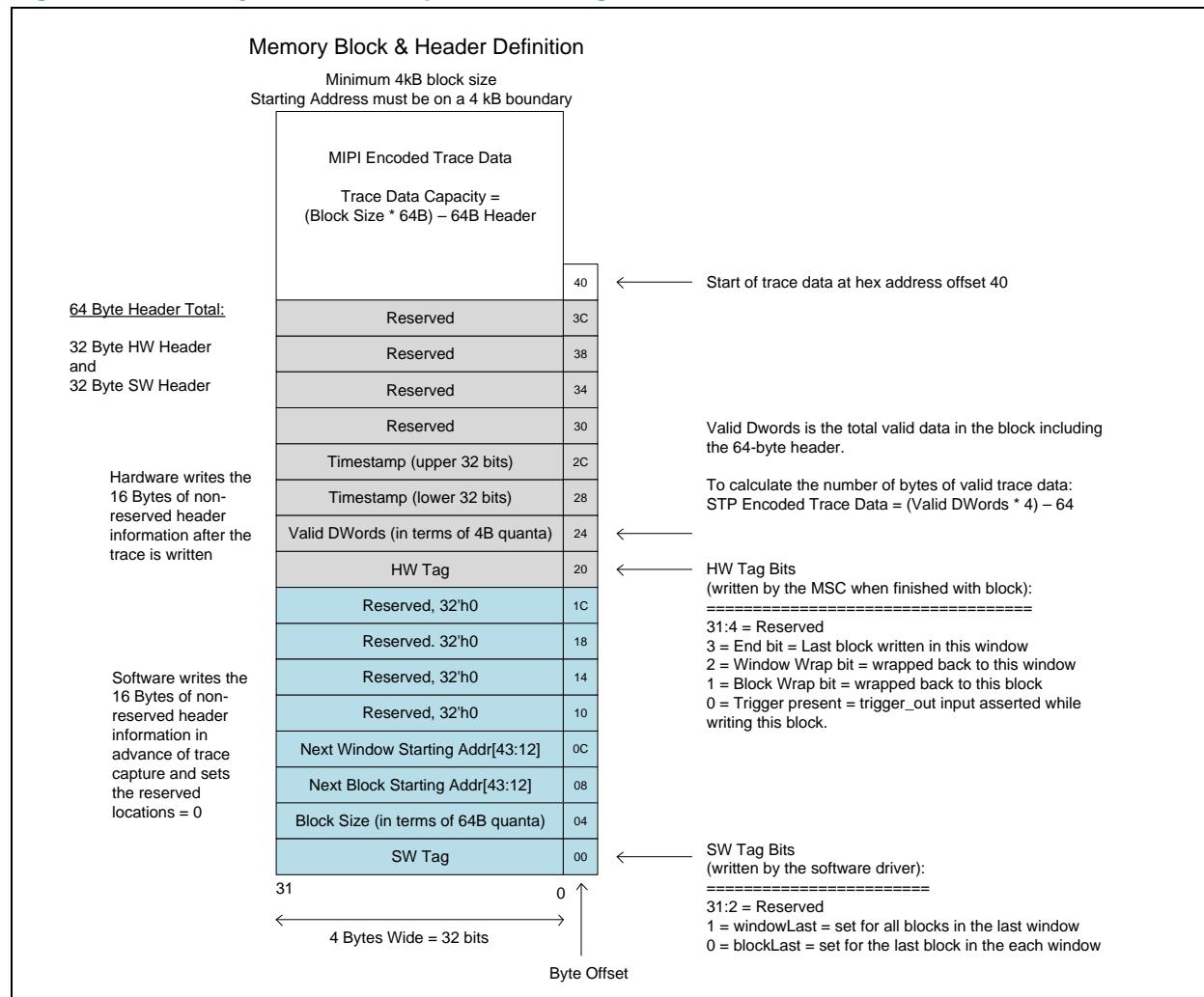
The memory block in single block (CSR-driven) mode has no header information, and can thus utilize 100 percent of its storage for storage of trace data.

For multi-block mode, the first 64 bytes of each memory block are reserved for header information. The header is divided into two areas: the first 32 bytes are reserved for software configuration, and the second 32 bytes are written by the MSC hardware. The software headers contain the information required to chain together the blocks into windows and buffers, as well as some descriptive ‘tags’ that describe the block. The driver software is responsible for gaining block allocation from the OS, and writing the software header information. The pointer to the first memory block must be put into the MSCnBAR registers.

The MSC will write the hardware header information when it is finished writing to the block.

Figure 34 shows the memory header topology within the memory block.

**Figure 34: Memory Block Description Showing Header**





### 8.1.5.1 Software Header Description

The software header contains the following 32-bit values:

**Ox0:** The 32-bit **SW Tag** implements the following bits. All unused bit locations must be written with zeros, and are to be ignored when reading.

- **Window Last** in bit position 1 indicates that the given block is in the last memory window. This bit should be set for all blocks in the last window, and written with zeros for all blocks not in the last window. This is used by the MSC hardware to know when it has wrapped past the last window to know when to set the window wrap bit.
- **Block Last** in bit position 0 indicates that the given block is the last block in this window. The MSC hardware uses this to determine when to set the block wrap bit in single buffer mode.

**Ox4: Block Size** defines the size of the current memory block. The value programmed in the block size is multiplied by 64 to represent the size of this block in bytes. The amount of trace data that can be captured is equal to this Block Size \* 64, less the 64 bytes of header. The minimum block size supported by the MSU is 4 kBytes. The capacity of Valid DWords is =  $2^{32}$  DWords = 16 GB, so block sizes should be kept to a maximum of 16 GB.

**Ox8: Next Memory Block Starting Address** is the physical address that points to the next block within this window. The value programmed in the next memory block starting address is left-shifted by 12 bits to determine the physical address. Because of this 12 bit shift, all blocks must start on 4 KB boundaries.

**OxC: Next Window Starting Address** is the physical address that points to the starting address of the first memory block in the next window when working in multi-window mode. Each memory block of the current window should have programmed into it the same value for Next Window Starting Address. This is because the MSU can jump to the next window from any block within the current window. The MSU will read the current block header and use this information, and does not save the header information from previous block headers. The value programmed in Next Window Starting Address is left shifted by 12 bits to determine the physical address. Because of this 12 bit shift, all blocks must start on 4 KB boundaries.

### 8.1.5.2 Hardware Header Description

The hardware header contains the following 32-bit values:

**Ox20:** The 32-bit **HW Tag** implements the following bits. All unused bit locations must be written with zeros, and ignored when reading.

- **End bit** in bit position 3 indicates that this memory block is the last block written with trace data in this window. That is, the trace capture was halted by either SwitchBuffer or CaptureDone and the last data for the window/buffer was stored in this memory block. Software uses the end bit to know where the end of the trace is. The beginning of the trace is also in this block if the block wrap bit is set. If the block wrap bit is not set, then the beginning of the trace is at the start of the window.
- **Window Wrap bit** in bit position 2 indicates the trace that is stored wrapped back to the first memory window. Once the MSU wraps back to the first window, it sets all window wrap bits in all HW headers.
- **Block Wrap bit** in bit position 1 indicates the trace that is stored in the window wrapped back to the first block in this memory window. That is, more data was



stored to the window than fits the size of the window, so the trace wrapped past the end back to the beginning of the window. Once the MSU wraps back to the start of the window, it will set all block wrap bits in all HW headers in this current window. If this bit is set, the meaning of the number of valid DWords field is altered. When block wrap = 1, valid Dwords indicates the position of the newest data in the memory window/buffer and valid Dwords + 1 indicates the position of the oldest data in the memory window/buffer. There is a corner case where the user could configure only one window. A window wrap would also be a block wrap. In this case, the MSU sets the window wrap bit.

- **Trigger Present** in bit position 0 indicates that the CTS asserted its Trigger\_out output to the system while the MSU was storing data to this memory block. The Trigger\_out is a very fast sideband signal to the MSU. Any action taken by ODLA to mark the trace data with this trigger\_out indication will likely be delayed from this point in time due to the buffering of data along the Intel TH pipeline, including the trace buffer in the MSC. Due to this buffering, it is very possible that the data being captured by ODLA at the time of the Trigger\_out assertion could be stored in subsequent memory blocks.

**0x24: Valid DWords (in terms of DWords)** indicates the total number of valid DWords in the memory block. To get a byte count, multiply valid DWords by 4. This result includes the 64 bytes of software and hardware header information. To calculate the number of bytes of valid trace data (the MIPI Encoded Trace Data) = (valid DWords \* 4) – 64.

In general, if the block fills with valid trace data, and the MSU exits to the next block at the end of the current block, then valid DWords will match the block size in DWords.

The exception case is if the MSU switches windows (due to switchBuffer) thus causing it to exit before reaching the end of the block; in this case, valid DWords \* 4 will be the offset to the next data that it would have written. In this case, SW can read valid DWords, and multiply it by 4 which can be added to the block's starting address to calculate the physical address of the oldest trace data if block wrap = 1, or to one past the end of the valid trace data if block wrap = 0. This latter calculation would also be used if the trace capture halted before the end of the current block due to CaptureDone.

**0x28: Timestamp lower** is the lower 32-bits of the TSCU timestamp. The TSCU timestamp counter is captured continuously by the MSU. This timestamp value is written by the MSU into each hardware header when it has completed writing the trace to the memory block. The timestamp information in the chain of memory blocks can be quickly scanned by the user post trace capture to navigate through the memory blocks and windows and to find a point in time that is interesting to the user.

**0x2C: Timestamp upper** is the upper 32-bits of the TSCU timestamp counter.

## 8.1.6 Operational Details

### 8.1.6.1 MSCnMODE, Wrap Enable and Wrap Status Behavior

Table 8-1 lists the relationships between the WRAPENn, WRAPSTATn, block wrap, window wrap and msu\_odla\_memwrap output for each operational mode of the MSC. Note that the msu\_odla\_memwrap signal is an output to the ODLA logic that indicates the memory buffer has wrapped, thus potentially over-writing trace data containing the last timestamp. For this reason, ODLA will wake if it's in a deep compression mode, and insert a timestamp packet into the trace.

**Table 8-1: Wrap Enable, Status, and Signal Relationships for Each Operational Mode**

<b>MSCnM ODE CSR</b>	<b>WRAPENn bit</b>	<b>WRAPSTATn status register behavior</b>	<b>BlockWrap and WindowWrap HW header bits</b>	<b>msu_odla_ memwrap output signal</b>
00 Single Block or CSR Driven Mode	<p>WRAPENn = 1 enables the MMI to MSCnSIZEstore more trace data than can be stored in one pass as indicated by MSCnSIZE. The MSU will "wrap" the memory write address back to MSCnBAR thus creating a circular trace buffer that will over-write older data at the start of the buffer with newer data. The trace buffer will then contain data at the end of the trace capture.</p> <p>When WRAPENn = 0, the MMI will stop writing trace data at MSCnBAR + MSCnSIZE. When it stops writing, it will set the MSU_INT bit in the MSU_STS register, and (naturally) backpressure the entire upstream data path. Thus, the pipeline cannot drain (cannot reach empty) in this case.</p> <p>WRAPENn controls the memory address wrap in Single Block mode, and thus wrapping the Trace Buffer write pointer past MTBDEP by the MSC Source Interface is always enabled in this mode.</p>	<p>WRAPSTATn will be set to 1 when WRAPENn = 0 and the memory block is filled (that is, when MSCnSIZE*4k bytes or more are written). Else WRAPSTATn = 0.</p> <p>WRAPSTATn = 1 when WRAPENn = 1 and MSCnSIZE * 4k bytes, or more, are written to the memory buffer. WRAPSTATn will remain set to 1 until MSCnEN = 0 or until reset is asserted, which will clear WRAPSTATn.</p>	Undefined for this mode	The MMI asserts a pulse on msu_odla_memwrap each time it wraps past MSCnBAR + MSCnSIZE
01 Multi Window Mode	Block and Window wrapping is always enabled in multi window mode.	Undefined for this mode.	<p>If the MSU wraps past the last block of a window as indicated by the blockLast bit, then all subsequent BlockWrap HW header bits will be set while in that window.</p> <p>If the MSU wraps past the last window back to the first window as indicated by the windowLast bit, then all subsequent WindowWrap HW header bits will be set.</p>	The MMI asserts a pulse on msu_odla_memwrap each time it wraps past the end of the last block in the window as indicated by the block last bit.  The MMI also asserts a pulse on msu_odla_memwrap after it exits a window when the WindowLast bit is set.

MSCnM ODE CSR	WRAPENn bit	WRAPSTATn status register behavior	BlockWrap and WindowWrap HW header bits	msu_odla_ memwrap output signal
10 DCI Trace Handler Mode	WRAPENn enables wrapping the Trace Buffer write pointer past MTBDEP by the MSC Source Interface. This should be set by the SW.	WRAPSTATn will be set to 1 when the Trace Buffer write pointer wraps past MTBDEP. This should be the normal operating condition.  WRAPSTATn will remain set to 1 until MSCnEN = 0 or until reset is asserted, which will clear WRAPSTATn.	Undefined for this mode	Not used. msu_odla_m emwrap should always be driven to 0 when in DbC.Trace mode.
11 Internal buffer mode	WRAPENn enables wrapping of the Trace Buffer write pointer past MTBDEP by the MSI in debug mode.	WRAPSTATn = 1 if the Trace Buffer write pointer wraps. That is, if MTBDEP * MD_WIDTH bytes (or more) are written, WRAPSTATn will be set.  WRAPSTATn will remain set to 1 until MSCnEN = 0 or until reset is asserted, which will clear WRAPSTATn.  These conditions are valid only if the MTB(s) are not read while trace data is being sent to the MSC(s).	Undefined for this mode	The MSC Source Interface asserts a pulse on msu_odla_m emwrap each time it wraps past the end of the Trace Buffer.

### 8.1.7 DCI Trace Handler

The Intel Trace Hub DCI Trace Handler (Intel TH DCI TH, or simply DCI TH) is a trace destination, or output port within the MSU. While it is a unique output port from a capability viewpoint, it shares a number of functional requirements with the MSU, and so is tightly coupled to the MSU architecture. For example, both the MSU and the DCI trace handler must buffer a certain amount of data before sending it to its final destination. For efficiency and area savings, the MSU and DCI Trace Handler have a single set of memory buffers and output logic between them.

#### 8.1.7.1 Multiple Independent Captures

The MSC is capable of performing multiple captures independent of the functional mode without the need for a power cycle or hardware reset. Each trace capture operation is atomic, and must complete before the next capture can be started. These independent capture operations must be delimited by setting MSCnEN = 0.

In single or multi-block modes, trace capture is normally completed when the trigger unit asserts the *captureDone* signal to the MSC, and the pipeline is flushed. These operations can also be terminated forcefully (though not without consequences) by the software by clearing MSCnEN to 0.

In DCI Trace Handler mode, both MSCs must be enabled and set to DCI Trace Handler mode to operate. Clearing either of the MSCnEN bits will halt the MSU's DCI Trace Handler mode operation. The *captureDone* input from the CTS gasket is not used by the MSCs in this mode (though it is used by the DCI Trace Handler). More details on this mode of operation are available in the [DCI Trace Handler](#) chapter.



The captureDone input is also not used in internal buffer mode.

When the software clears MSCnEN to 0, the MSCnTBWP, MSCnTBRP (trace buffer write and read pointers) are reset to 0. The MSCnMWP (memory write pointer) is loaded with the contents of the MSCnBAR. If any of this status information is required to be used or saved, the software must read these CSRs before clearing MSCnEN. The process of clearing MSCnEN to 0 allows for the all-important state from the previous capture operation to be reset, thus allowing a new configuration (including a new starting base address and mode of operation) to be programmed. Setting MSCnEN to 1 will re-start the MSC capture operation in the new mode of operation.

When MSCnEN is cleared to 0, the trace data in the trace buffers (MTBs) is not cleared. It is possible to subsequently switch the MSC to internal buffer mode and read the trace buffer contents if this aids in debugging the system.



## 9 DCI Trace Handler

The Direct Connect Interface (DCI) is an emerging transport technology for closed-chassis debug access. The primary purpose of DCI is to allow existing functional I/O (e.g., a USB port) to be used for debug of the system and/or silicon, and to gain access to trace/debug features in the silicon.

The Intel Trace Hub DCI Trace Handler (DCI TH) is an ingredient of the Intel Trace Hub Architecture. It orchestrates transfer of debug trace data collected from the Intel Trace Hub to a USB port. The main purpose of the DCI TH is to provide high speed continuous streaming by smoothing out transport latencies and the bursty nature of debug/trace traffic through the use of a large buffer. The DCI TH supports two types of trace streaming destinations:

- USB Streaming: High-speed streaming over a USB port supporting a xHCI.DBC.Trace interface. The USB Host controller is responsible for packetizing and streaming over USB.
- BSSB Streaming: Low-speed streaming to USB port supporting a Boundary Scan Side Band (BSSB) transport protocol. The DCI Bridge is responsible for packetizing streaming over DCI.BSSB

The streaming mode is configurable in the DCI TH. Current usage modes suggest configuration of the mode by the debug host software stack. While the BSSB Streaming mode exists in Intel TH hardware, the LPP direct path is the preferred and recommended method to send trace data to DCI.

The DCI TH is responsible for delivering data from the Global Trace Hub (GTH) to a USB Host controller or DCI Bridge.

As with the other Intel Trace Hub components, the DCI Trace Handler can function independently of the operating system, as it is completely configurable and controllable via JTAG.

### 9.1 Functional Requirements and Limitations

#### 9.1.1 Requirements

- Supports the following Streaming modes:
  - "High-speed" streaming over USB through xHCI.DBC.Trace end point
  - BSSB mode streaming through the DCI Bridge.
- Supports switching between USB and BSSB modes (Host assisted)
  - One USB Port model: Requires Unplugging and plugging USB cable/ BSSB adaptor
  - Two USB Port model: One USB port streaming from XHCI.DBC.Trace and the other streaming in BSSB model.
    - Only One destination active at any time
    - Host assisted Switching between destinations.



- Can back-pressure GTH for lossless trace streaming.
- Support data transfers through the primary and sideband fabrics
  - All trace streaming related communication to/from DCI TH in DBC mode is on the primary fabric.
  - All trace streaming related communication to/from DCI TH in BSSB mode is on the sideband fabric.

### 9.1.2 Limitations

- No simultaneous streaming to more than one destination:
  - Only one device (xHCl.DbC or BSSB) is supported for streaming at any time.
  - Store to memory is not supported while DCI Trace Handler is active.
- No error detection/correction at storage buffer.
- Not a reliable data transmittal protocol
  - No support for reliable protocol or retries of data transfers.

## 9.2 Trace Streaming Modes

DCI TH can support two streaming modes: "High speed" USB mode streaming, and "low speed" BSSB mode streaming.

### 9.2.1 "High Speed" USB Mode

In "high speed" USB mode, trace data is streamed over the primary fabric to a USB port. A USB host controller (xHCl) provides a debug trace interface (xHCl.DBC.Trace) to facilitate the data transfer to it. The xHCl.DBC.Trace interface provides a DBC.Trace.IN register (in MMIO space) to setup a DMA transfer. A write to this register with data address and size will trigger a DMA in the form of one or more memory reads to the data address. When the data address points to the DCI TH memory buffer, it results in debug/trace data streaming.

### 9.2.2 "Low speed" BSSB Mode

In "low speed" Boundary Scan Side Band (BSSB) mode, the DCI TH writes the trace data to a buffer in the DCI Bridge. The DCI bridge packetizes this data into DCI packets and streams them over a USB port configured in BSSB mode. DCI packets are of constant 64B in size and allows up to 60B data payload. Communication between the DCI Trace Handler and the DCI Bridge is accomplished over the sideband fabric interface. Since the DCI TH handles data only in integral multiples of 8B (owing to internal microarchitecture details), and the sideband fabric is limited to 8B of payload per transaction, the DCI Trace Handler transfers streaming data to the DCI Bridge in 56B blocks over 7 sideband transactions.

While the hardware for BSSB mode exists, the recommended method for sending trace data to a BSSB port is via the LPP path.



## 9.3 Switching Trace Streaming Modes

The DCI Trace Handler does not currently support dynamic switching of streaming modes. One mode must be completely finished before the DCI TH can be reconfigured for another streaming mode, and then started up.

## 9.4 End of Trace handling

In an end-of-the-trace scenario, the DCI Trace Handler must be flushed to ensure all data is transmitted to the destination. There are two methods to flush the DCI Trace Handler: automatic flush, or manual flush.

The automatic flush case utilizes the Intel TH Trigger Unit to assert the captureDone output at the end of trace capture. When the DCI TH sees this assertion, it will automatically enter its “flush mode”, sending all data to the destination. The FLUSHDONE bit can be polled to determine when all flushing is done (despite the fact that the MSCs are involved in this trace destination, the interrupt is not asserted in DCI mode).

The manual flush case must be used when the Host software has not been configured the Intel TH for automatic flushing at the end of trace capture. Host/debug software will need a way to detect the end of the trace. This can be determined from Intel TH trigger state, perhaps the rate or flow of data, a specific message, or some other condition (exact method is outside the scope of this specification). Once the software detects this condition, it can force a manual flush of the last partial packet from the trace buffer by using the STREAMCFG1.FLUSH bit.

In both cases, if there is no data in the MTBs to send to the DbC.Trace or DCI Bridge, the DCI TH will immediately cease operation, sending no further writes to either endpoint, and will set the FLUSHDONE bit.

## 9.5 Error Handling

### 9.5.1 DbC.Trace Device Unresponsive

When the DbC.Trace device becomes un-responsive (from the DCI Trace Handler’s viewpoint – not requesting data or returning status) it is detected at the DCI TH through a time-out counter. After the counter expires, the DbC is deemed un-responsive, and the DCI TH starts a fresh cycle of setting up DMA transfers to the DbC.Trace endpoint.

After a number of such time outs occur (MAXTIMEOUTS), the DCI Trace Handler asserts its *portReset* signal to the GTH. This signal informs GTH that the trace destination is not able to accept data, allows GTH to use its PnDROP policy to drop the trace data if so configured. It is recommended to set the PnDROP policy bit, or to ensure the STH BDC is set to a low value so that data is dropped when the DCI TH is unable to make forward progress.

### 9.5.2 DCI Bridge Unresponsive

When the DCI Bridge becomes unresponsive (from the DCI Trace Handler’s viewpoint – not requesting data or returning status), it is detected through a time-out counter. Similar to the DbC.Trace unresponsive scenario, a time out counter triggers a “reset” and allows the DCI TH to repeat its attempt to send the data. After multiple timeouts (MAXTIMEOUTS), a port reset is asserted to GTH.



### 9.5.3 Unexpected Reads from DbC

Unexpected reads from DbC (when the DCI Trace Handler doesn't have any outstanding DMA) are completed with dummy data.

### 9.5.4 Unexpected Credit Returns from DbC

Credit returns from DBC when DCI Trace Handler has full credits are ignored. Credit return when DBC returns a credit before completing all the outstanding reads for a given DMA is processed normally.

### 9.5.5 Unexpected Credit Returns from DCI Bridge

Credit returns from the DCI Bridge when DCI Trace Handler has full credits are ignored. When the DCI Bridge returns credits before a full 56B transfer, it is processed normally and DCI Trace Handler starts counting 8B transfers towards a new transaction.

### 9.5.6 Reads to a Non-Sequential or Unexpected Address from DbC

The DCI Trace Handler assumes the reads from the DbC.Trace to come for addresses in sequential order for the DMAs advertised. As such, the DCI TH does not inspect or use the address of the read request. Rather, it supplies the data from the trace buffer in a sequential manner for each successive read. Any non-linearity in addresses of the read request is ignored.

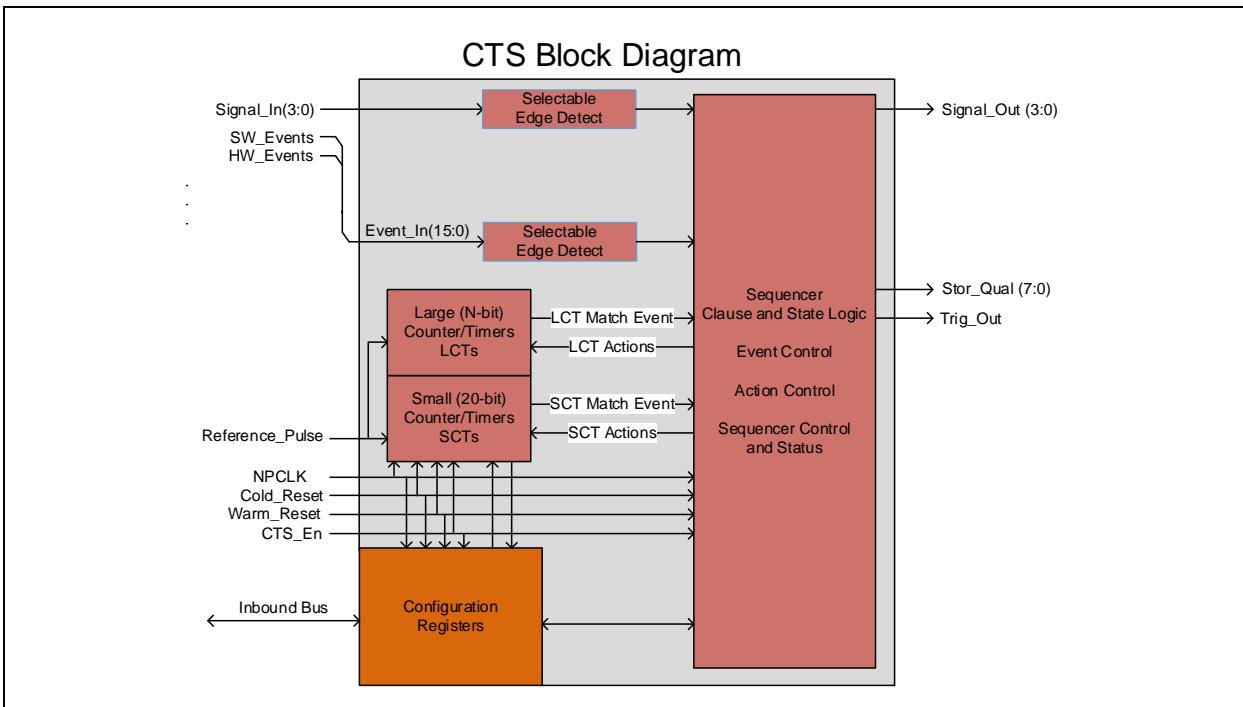
# 10 Common Trigger Sequencer

The Common Trigger Sequencer (CTS) provides the core triggering capability for the Intel Trace Hub. The CTS provides the ability to detect one or more sequences of events, and to take one or more actions for each detected series of events. For each sequence of events, one is chosen to be called the "Trigger" – typically the state or condition enabling the start of data storage.

## 10.1 Functional Description

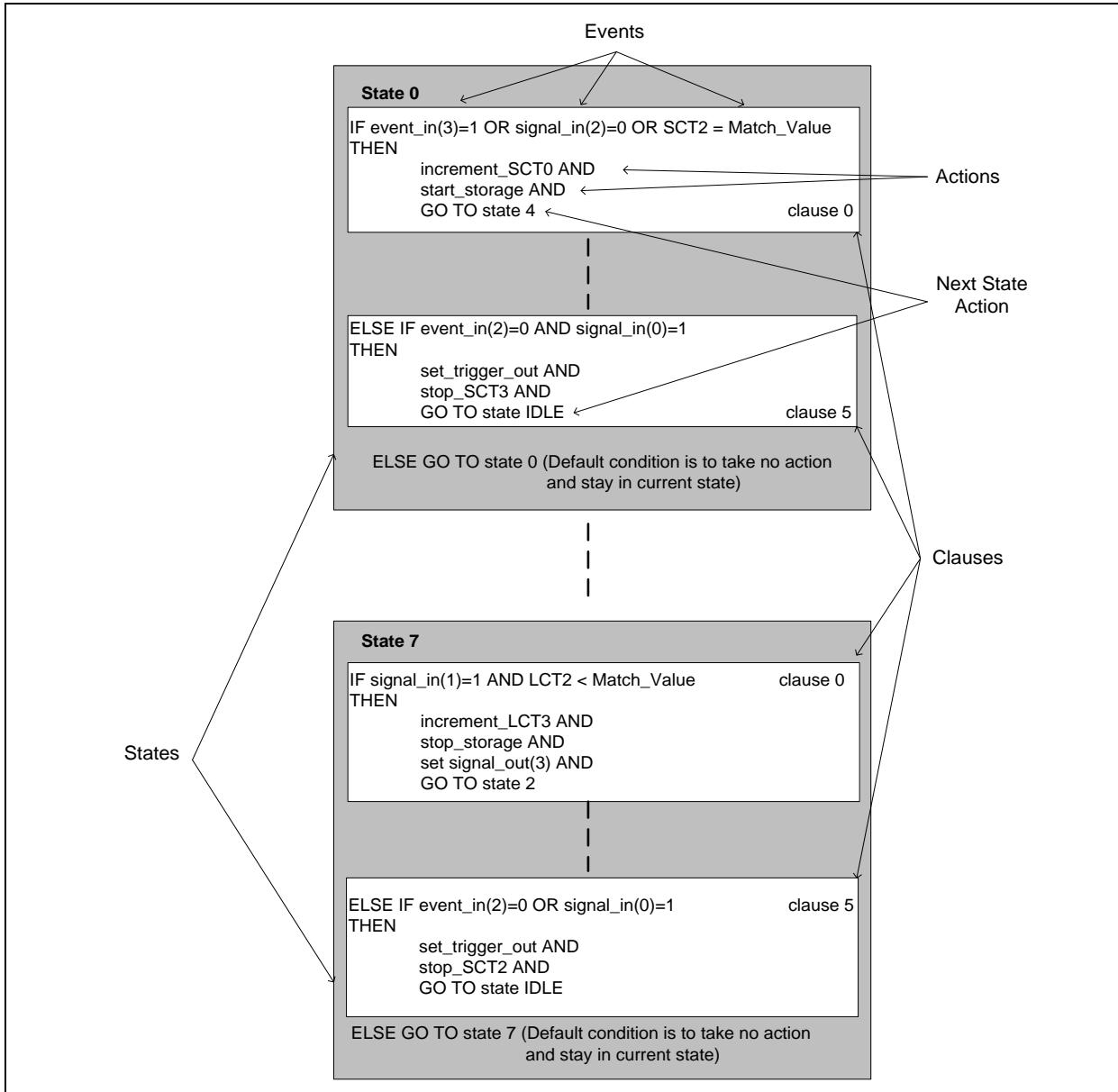
Figure 35 shows a CTS block diagram. The major blocks include the Trigger Sequencer State Machine, the Global Counters, and the Timers and the Configuration Registers. Each of these sub-blocks is described in detail in later sections.

**Figure 35 : CTS Block Diagram**



The CTS utilizes states and clauses to process events and take actions. Figure 36 describes the terminology of the relationship between clauses, states, event inputs, action outputs and next-state indication within the CTS. A **clause** is an IF or ELSE IF pairing of events and actions within a state. A **state** represents the present condition of the system that is being monitored (or traced), and can be the result of one or more conditions, events, or sequences of events. For example, a state can represent the starting point of a system – like initial condition(s), or it can represent that a particular sequence of events has occurred. An **event** is the signal (or combinations of signals in the case of external match/mask logic), including both the assertion and de-assertion of signals, which are used as inputs to the clause. **Actions** are stimulus that can be defined to be taken by a clause when event inputs match or "hit".

Figure 36: Textual Example of Sequencer Terms



In order to build a trigger specification (which is the set of states, events, clauses, and actions the sequence will follow), the user must determine:

- (1) the events (inputs) that will be needed,
- (2) the resources that will be used, and
- (3) the sequence (if any) that must be followed.

Then the trigger specification is written by groupings (states) of a series of IF ... THEN ... ELSE statements (clauses). Each of these concepts is described throughout the remainder of this document.



### 10.1.1 CTS Global Counter/Timer Resources

The CTS implements global counter/timers where all clauses of all states have access to and shared control of all counter/timers. The number and size of the counter resources are selectable via parameters. There are two types of counter/timers available. The “small” counter/timer (SCT) is 20-bits wide and the “large” counter/timer (LCT) is 45-bits wide. The maximum number of SCTs is 4 and the maximum number of LCTs is 4; the actual number is a per-project parameter (please consult your product-specific documentation for details). The SCTs and LCTs are implemented as count up (incremental) counter/timers. They are reset to 0, and compared against a programmable match value. This comparison from all counter/timers is sent as an event signal to all clauses. The value programmed for comparison is unique to each counter/timer, but shared by all clauses. Counting of events or measuring of time using the SCTs and LCTs can span across state transitions.

Operating as either a counter or a timer, the SCTs and LCTs will match against a respective *match\_value* register every cycle. The result of this match translates into an event input available to all clauses of all states in that current cycle. This allows for full rate operation and sampling of the counter/timers.

When operating in timer mode, the \*CTn\_start\_inc action will start the timer. It will operate as an incrementing timer and compare its *match\_value* register every cycle. If a match event occurs, the SCT or LCT will stop incrementing and set the match event output to the clause logic, and remain set. It will then wait for a \*CTn\_clear action or hard reset to reset back to zero. While counting in timer mode, the LCTs will stop and suspend operation if the \*CT\_stop action is signaled. As a result, the contents of the counter will remain unchanged, and the counter will then wait for either a \*CTn\_start\_inc action signal to continue from the current value, or a \*CTn\_clear action signal to clear the counter to zero. The SCTs don't have the ability to stop and a stop action is not defined for the SCTs. In any single cycle, the timer can be instructed to clear and start indicated by the \*CTn\_clear and \*CTn\_start\_inc action signals *true* simultaneously. In such a case, the timer will reset to the value 1h. This effectively allows the counter to start counting in the same cycle that it's cleared.

When operating in counter mode, the \*CTn\_start\_inc action will be used to tell the counter when to increment. It will operate as an incrementing counter and compare its *match\_value* register every cycle. If a match event occurs, the SCT or LCT will set the match event output to the clause logic. Since the user may desire to count events beyond the match event, the SCT or LCT will continue to increment if the \*CTn\_start\_inc bit is set after the match event, instructing it to increment, allowing the user to count past the match event. Keep in mind that the match event is sticky, so it will remain set if the CT increments beyond the match value. This match event signal is cleared when the CT is cleared or reset, yielding a simpler, more flexible design. The \*CTn\_clear action signal will cause the counter to reset to zero. The \*CTn\_stop action input is valid for timer mode and undefined in counter mode.

The following table shows the counter and timer sizes translated to ranges for different example frequencies.

**Table 10-2: Counter and Timer Ranges**

Counter or Timer Width	Maximum Count Range	Timer range at 100 MHz	Timer range at 2 GHz NPCLK (as an example frequency)
20 – bit SCT	1 M events	10.5 mSec	523.7 uSec
45 – bit LCT	32 T events	4.1 days	4.9 hours

#### 10.1.1.1 Peak Timer Mode for LCTs

The Peak Timer mode for LCTs gives the user the ability to measure the worst-case time from one event to another event and store the maximum (peak) count value in a readable status register for inspection by the user.

This feature is enabled with the LCT\_Mode(1) configuration register bit in the LCT Control Register. The comparison logic for the LCT\_0 uses the peak count status value for the LCT match comparison if the LCT\_Mode(1) = 1. The clause logic can be configured to match on start and stop events, and to take the LCT *clear* action upon the second event. The LCT operates in timer mode during this operation and counts on NPCLK. If the time between events exceeds the previously-stored peak count time (i.e. a match), then the peak count status is updated with the contents of the LCT functional register.

The following example highlights the value of the requested capability to repeatedly measure time between event\_0 and event\_1 (or any combinations of events currently available within a clause), and save as status the maximum duration.

*State\_0:*

```
// start LCT_0 upon first event, then free running
If event_0 = 1 then start LCT_0 and go to State_0
// clear LCT_0 upon second event
Else if event_1 = 1 then clear_LCT_0 and go to State_0
// updates peak timer if LCT_0_Match = 1
Else stay in State_0
// default clause
```

To summarize, the peak timer value for its cycle by cycle comparison, and the peak timer value is a status register that is updated with the LCT clear action that is readable via the sideband fabric. Start or Stop actions will not affect the peak timer logic, which improves the flexibility of this feature. While in the IDLE state after a trigger action the peak timer contents will be preserved to give the user software time to read its contents. The peak timer status register will be cleared when the CTS is enabled and transitions out of the IDLE state.

#### 10.1.1.2 Dedicated Event Counter Mode for SCTs

The Dedicated Event Counter mode for SCTs gives the user the ability to count event\_in events without the need for increment control by the clause logic. Additional configuration register bits to support this include the SCTn\_Mode(1) bit, and the SCTn\_EventSel(n:0) bit to select which event\_in(15:0) to count. SCTn\_EventSel(n:0) will only implement the number of bits required to select the full range of Event\_In(n:0) inputs selected via parameterization. For example, if 8 Event\_in(7:0) inputs are parameterized into the design, then three SCTn\_EventSel(2:0) registers are required.



If the dedicated event counter mode is selected by the user, then the SCTn\_EventSel(n:0) will be used as a mux select to choose what event\_in(15:0) input to count. The SCTn\_start\_inc action will not be used as an increment action. The SCTn\_clear action will still be used to clear the counter. In this mode, the SCT will only increment if the selected Event\_in input wire is (high true) in the current cycle. This increment decision is made each cycle that the CTS is active, which means the SCT dedicated event counter activity starts counting from zero when CTS is enabled and exits the IDLE state, and continues until the CTS enters the IDLE state or is disabled. Additionally, the SCTn\_match\_value will still be compared against every cycle to create the SCTn\_match event to the clause and state logic.

The user can program the SCTn\_match\_value to a desired value, enable the dedicated event counter mode, and select which SCT is to count which Event\_In input wire. The user can then take advantage of the SCTn\_match event knowing that hardware is supporting the generation of comparing for a certain event count on the configured SCT. At any time the user can clear the SCT to zero and know that the dedicated event counter will continue counting from zero. If the user clears the dedicated event counter in the same cycle that the selected Event\_In is asserted, then the dedicated event counter will count that event, and actually be cleared and incremented to 1 (the SCT is actually cleared to 1 in this case).

To clarify: to use this mode the user will have to perform the following actions:

- Program SCTn to be in the Local Counter mode
- Select which event\_in(15:0) to select to count
- Program the match value for SCTn
- Program the trigger clause to "If SCTn\_match then trigger"

The ELSE clause would not be needed to perform the event counting as the logic to do this is automatic, and supports multiple counters operating in parallel, potentially freeing up multiple clauses. This is known as a local counter operation (CTS is calling this the dedicated-event counter mode) and matches the capabilities of external logic analyzers and other on-die methods to count events. The CTS is not designed to be a performance counter *per se*; however, it does add capabilities to the CTS small counter/timers to aid in performance counting.

#### 10.1.1.3 Internal Counter/Timer Events

Counter/Timers are global resources that operate in a dual mode as either a counter or a timer. The mode of operation is controlled by that counter/timer's control register settings. In both modes of operation, the counter or timer is reset to zero and increments from zero. The current value of the counter/timer is matched against a programmed match value every cycle. Having the counter/timers operate in dual modes is done to share the increment and match logic for both modes. When configured to operate in a given mode, its resources are available to all clauses of all states, but each operates in one mode at any time. Each counter/timer has an output match event signal that is available to all clauses of all states. Once a match is signaled, it is "sticky" and thus remains true until the CT is cleared, or reset. The timer or counter will match against the counter's match\_value setting programmed in its control register. Each clause can individually select any combination of SCTs or LCTs match event output as *match* or *not matched* event inputs enabled for that clause. There is an option to match when the match value is not equal to the value in counter match value register (the event signal being low indicates not a match). This option is set via the SCTn\_mode or LCTn\_mode bits in the clause event mode registers. The counter will only count if the start\_inc action is true for that cycle, and the counter will



count the number of cycles that the start\_inc action is true. In timer mode, the start\_inc action means to start the timer free running, and will increment every cycle until it sees a stop or clear action or a reset event. In timer mode, the timer will always function at NPCLK frequency, but it may count using a prescaler to get coarse granularity. Timers have 2 modes of operation. The timer can either be free running at NPCLK frequency, or it can be configured to count Reference\_Pulse input signals being high true. For Intel Trace Hub 1.0 the Reference\_Pulse input is always asserted, so there is no difference between these two modes.

### 10.1.2 External Input Events

There are 20 external event inputs that can be combined to create a match event per clause / per state in the sequencer. These are the Event\_In(15:0) and Signal\_In(3:0) input pins. These external events are combined with internal counter/timer events and available in combination to all clauses of all states.

There are control bits in the CTS Control register and the clause event mode enable control registers that control the treatment of the primary inputs and their effect on each of the clauses within each state. Each clause of each state will have a unique event mode and event enable register to control the clause's behavior. The input configuration register is shared by all clauses of all states. Both Event\_In(15:0) and Signal\_In(3:0) inputs have rising edge detectors that can be selected with registers in the CTS Control configuration register.

To select which of the inputs should be considered for a match event per clause, there are a set of control registers for each clause for each state of the sequencer. In each clause control register, there is an OP Type bit which indicates whether all of the enabled input matches for that clause are "ORed" together or "ANDed" together. The reset or default value of all these register bits is 0. If we are over budget on area, consider that these registers can be made to be a non-reset type, and the software could then be sure to write valid values into all when it configures the CTS.

### 10.1.3 Sequencer Actions

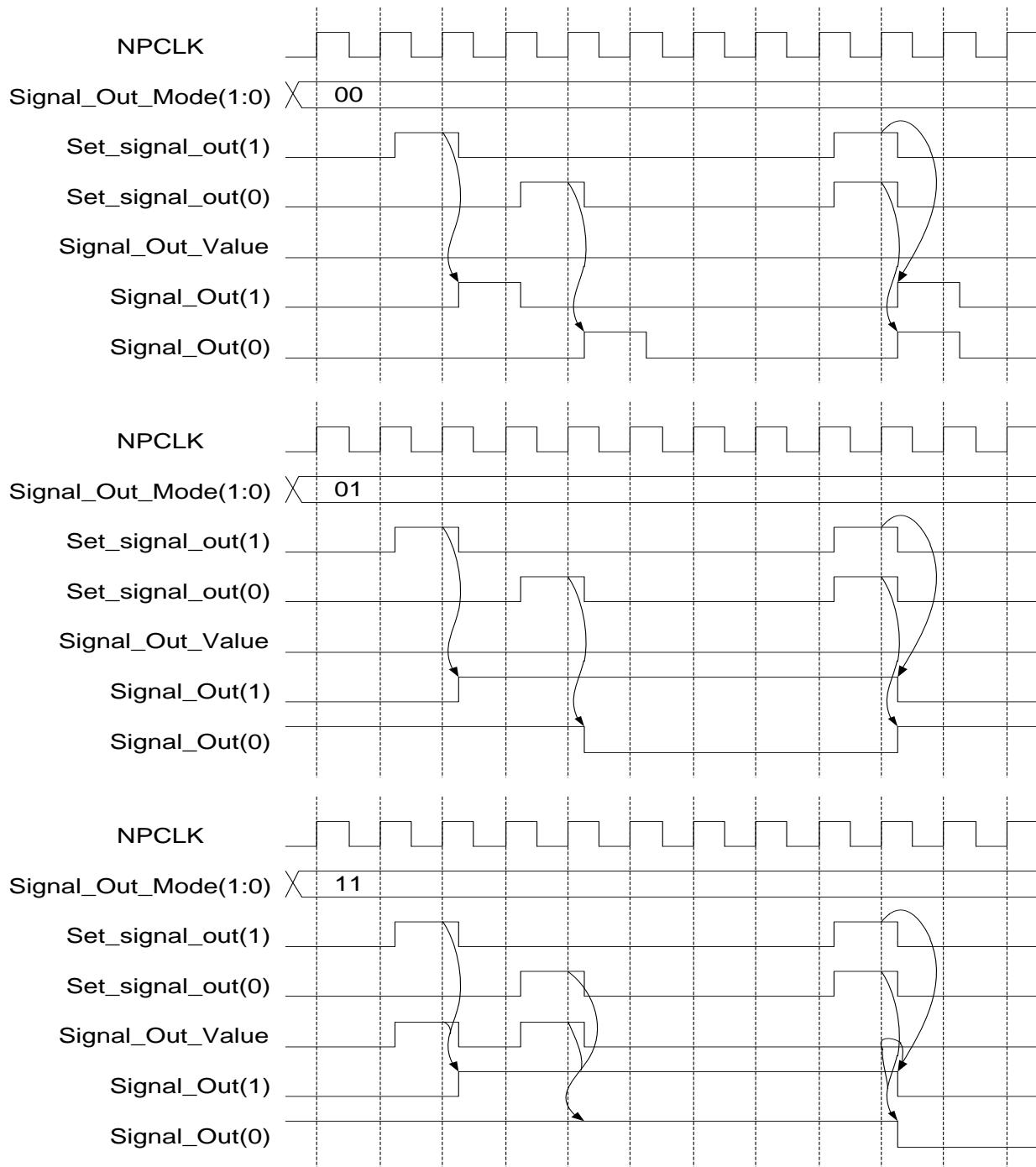
Upon a match event in any clause from any state of the sequencer, a number of output actions can be taken. There are seven external actions: Set\_signal\_out(3:0), Set\_trig\_out, Start\_storage, and Stop\_storage. Other internal actions are all taken upon the sequencer state machine itself, such as going to another state, starting a timer, or incrementing a counter. All actions selected in the clause output control registers will happen simultaneously upon a match event, but some selections are invalid when others are selected, such as starting and stopping a timer in the same cycle.

#### 10.1.3.1 Signal Actions

Up to four Signal\_Out(3:0) actions can be specified when a clause match event occurs. Set\_signal\_out(3:0) will set the Signal\_Out signals that are routed as outputs of the sequencer and can be used to communicate to other circuits or blocks in the SOC where the user can select the signal\_out wires to be pulsed, to toggle, or to signal a level. If the pulsed mode is selected, then the signal\_out wires will be pulsed for one NPCLK on each Set\_signal\_out action. If the toggle mode is selected, then the signal\_out wires will invert on each Set\_signal\_out action. If the level mode is selected, then the Signal\_Out(3:0) wires can be set to the value of the Set\_Signal\_Value action qualifier. The signal\_out wires can be used to set a marker with a timestamp in the trace, but that requires external

implementation specific logic. The following diagrams show the different behaviors of the Signal\_Out modes.

**Figure 37: Timing Diagrams Showing the Signal\_Out Mode Behavior**





### 10.1.3.2 Storage Control Actions

There are Stor\_Qual(7:0) output signals from the CTS to the storage subsystem used to communicate storage control. The Start\_storage action will set the Stor\_Qual output and the Stop\_storage action will clear the Stor\_Qual output. These actions to the Stor\_Qual outputs are sticky, thus Stor\_Qual is defined as level output signaling.

### 10.1.3.3 Trigger Action

The Set\_trig\_out action will pulse the Trig\_Out output signal for one NPCLK cycle.

### 10.1.3.4 Counter Actions

There are two counter actions per counter/timer that can result from a match event within a clause. Upon a match, selected actions can cause the counter to be incremented by 1, or cleared (set to 0). It is not valid to increment and clear from the same clause – at most one may be selected.

### 10.1.3.5 Timer Actions

There are three timer actions per counter/timer that can result from a match event. Upon a match, selected actions can cause the timer to start counting which means it will start from 'h0 and begin incrementing, or start from its current value if it's previously been stopped. The timer can also be stopped with the stop event for subsequent re-start if desired (note the stop action is only available on the LCTs, and not the SCTs). The timer will set the timer\_n\_match bit to the mask and match logic during the cycle that it reaches the match value. The timer will halt when it reaches the match value. The timer can also be reset based off of a match event. This means the timer will be reset to zero. Another match event with start timer selected must be used to start the timer again. It is valid to start the timer and clear the timer from the same clause, so the value loaded should be 'h1 in this case. The LCTs can also be stopped with its current value preserved waiting for another start action to resume counting. This is useful for accumulation of time.

### 10.1.3.6 "Go To" State Actions

Upon a match event for any clause, the output action can be configured to go to any available state, including the current state and the idle state. This allows for the sequencer to be used to look for a sequence of events (eg. 'A' followed by 'B', followed by 'C') among other trigger scenarios. The CTS is specified to be a one-hot state sequencer, that is, there is always one and only one active state. A requirement of the CTS is that each clause in each state must specify the next active state by having the Set\_state(3:0) configured in the action control register. Any value programmed in this register must define a valid entry as there is no hardware support to avoid illegal state transitions. In other words, if 4 states are parameterized & included in a given local CTS instantiation, then the Set\_state(3:0) value must be within the range of 0 – 3 or IDLE. Any illegal state transitions programmed and taken may cause a hang of the CTS. The hardware for the clauses are built as follows:

```
If (condition) then
    Take Actions, and go to state W
Else if (condition) then
    Take Actions and go to state X
Else if (condition) then
```



```
Take Actions and go to state Y
Else if (condition) then
    Take Actions and go to state Z
Else
    Stay in current state and take no actions
```

What is implied and will be supported by hardware is the final "**Else stay in current state and take no actions**" choice. The user need not define this, and the software can assume this action is hard wired.

#### 10.1.4 Sequencer Control and Status

##### 10.1.4.1 Sequencer Control

The Sequencer\_Enable configuration bit has a rising edge detector built on it to form a "start" bit internally. When this start bit is true, the CTS logic will use the Force\_State configuration register setting to inform it if it's a normal start condition where the state machine will proceed to state 0, or if it's a power state re-start condition where all RW\_V shadow status registers will be "jammed" back into the functional registers. These registers jammed are the current state, Stor\_Qual state, signal\_out state, all counter timer values, and the currently running indication of any counter/timer that is programmed into timer mode. Clearing the Sequencer\_Enable bit will reset all the counter and timer resources and the state machine will go to idle. All of the clocks for this block of logic can be gated when the enable is cleared. The enable should be cleared on a cold reset of the part, but if possible should survive a warm reset so that the triggering logic can be used during reset cycling of the part. The enable bit must be set to turn on the clocks to this logic.

##### 10.1.4.2 Sequencer Status

This section describes the status registers available as status in the CTS. All status register can be written by the software to be used to "jam" (force) the sequencer state by using the Force\_State register during a power state restart. During normal operation, an internal connection from the CTS to the register block will be generated that is the Update\_Status configuration bit tied to the update input to the register block. Update is used as a status update enable for all shadow status registers. The software can clear the Update\_Status register in advance of reading all the numerous status registers. This allows for a "snapshot" of the constantly changing state of the CTS to be held for observation while the software reads all the status registers.

###### 10.1.4.2.1 Sequencer State

The four Sequencer\_State(3:0) bits are a binary encoding of the current state the sequencer is in and can be read by software to give a real time update of the current state of the sequencer. There is a maximum of 8 parameterized states available, plus IDLE. The Sequencer\_State(3:0) bits can be read real time reliably because the register access interface is synchronous to NPCLK as is the sequencer itself.

0000 = State 0

0001 = State 1

0010 = State 2



0011 = State 3  
0100 = State 4  
0101 = State 5  
0110 = State 6  
0111 = State 7  
1000 = IDLE

#### 10.1.4.2.2 Trigger Status

The trigger status bit indicates that a trigger action on Trig\_Out has occurred.

#### 10.1.4.2.3 Stor\_Qual Status

The Stor\_Qual\_Initial\_Value status register will be used by the CTS to set the state of the Stor\_Qual output signal upon a sequencer start event as signaled by the rising edge detect logic on the Sequencer\_start configuration bit. Once the CTS has started operation as indicated by the sequencer\_running signal, the Stor\_Qual\_Initial\_Value register will act as a shadow status register of the Stor\_Qual output bit. The Stor\_Qual\_Initial\_Value register is similar to other status registers, but differs in one way, that is, the Stor\_Qual output is always forced with the Stor\_Qual\_Initial\_Value register upon a sequencer start up regardless of the state of the Force\_State register contents.

#### 10.1.4.2.4 Counter/Timer Status

Each of the Large and Small Counter/Timers has a current status register that reflects the real time contents of the counter/timer during operation. An additional status bit is available if in timer mode called \*CT\*\_Running is a status indication that the timer is currently running due to a previous start action. The software can reliably read the counter or timer's current status registers because the register-read interface is also synchronous to NPCLK within this block of logic.

#### 10.1.4.3 Conditions That Affect Use Cases

The following table is a compilation of internal events and a description of how they affect the internal logic relating to usability. The intent is to enumerate the different corner cases and have a table describing what internal actions are taken.

Table 10-1: Corner Case Actions

Condition or Event	Cause of Event	Default Action Taken	Description
Sequencer Start	Sequencer_Enable = 1 & Force_State = 0	State transitions to state 0 All counter/timers reset to 0h Trig_out = 0 Signal_Out (3:0) = 0h Stor_Qual(7:0) = Stor_Qual_Initial_Value(7:0)	This is a normal startup condition.



Condition or Event	Cause of Event	Default Action Taken	Description
Sequencer Start with forced state	Sequencer_Enable = 1 & Force_State = 1	State transitions to contents in Sequencer_State(3:0)  All counter/timers are loaded with contents of the status shadow registers, including the running status  Trig_Out = 0  Signal_out(3:0) = 0h  Stor_Qual(7:0) = Stor_Qual_Initial_Value(7:0)	Restoring the state after a power event. This is identified using the Force_State configuration register.
Sequencer Stops	Clause event match causing next state transition defined as IDLE	All counter/timers stop at current value  Trig_out = 0  Signal_Out(3:0) = 0h  Stor_Qual(7:0) holds the current value, unless there is a stop_storage or start_storage action in this cycle, then it takes that action and holds last value when IDLE.	This is normally due to a trigger condition, and is part of normal operation
Cold_Reset	Cold_Reset Input pin assertion	All resettable registers in the design are reset to their default values. This includes the reset on the flops for all inputs and outputs, they must all reset to 0. The state will reset to IDLE = 1000.	This is a normal cold reset that is normally due to a power cycle event.
Software Reset	Software writes Sequencer_Enable = 0 configuration register	Will not affect the configuration registers except for the status registers. State is reset to IDLE, Counter/Timers will halt and reset to 0, Outputs are all reset to 0.	

### 10.1.5 Clause and State Operation

All states are identical except that the sequencer will always progress to state 0 from IDLE when the Sequencer\_Enable = 1 and Force\_State = 0. This begins normal operation of the trigger sequencer operation. The independent states will share the primary event inputs after the synchronization and edge detect logic. The states will also share the internal events resulting from the counter and timer match or not matching. Each clause of each state has an Input Control Register associated with it that determines which events in combination can cause a TRUE or Logical "1" on the output of its event qualifier. This translates to a TRUE on its "if" or the "else if" clause. Multiple events could be AND'd or OR'd together, but no other combination of AND/OR is possible. This AND/OR function is selected by the OP\_type configuration registers.

The CTS supports AND/OR of all enabled events on a per clause basis. Additionally the inverse of the event can be chosen using the clause event mode registers. Other AND/OR event combinations can be supported with the ELSE/IF implied within the clause logic. For example:

IF ((A AND B) OR (C AND D)) THEN perform action X

is not supported by the CTS, but can be performed with the CTS by programming as follows:



```
IF (A AND B) THEN perform action X          // clause 0
ELSE IF (C AND D) THEN perform action X      // clause 1
ELSE do nothing
// default clause
```

It is important to note that a priority encoder is built to form the "if / else if / else if ..." clause sequence for the state. The clause 0 event qualifier result will be the first "if" clause. If clause 0's conditional match logic (hit) is true, then it will gate off the actions from all later clauses via its inverted value being an input to the later states AND. It follows that clause 1's conditional "else if" operator is only enabled if clause 0's didn't "hit", and if true will disable clause 2 and clause 3, etc. The S\*\_active input indicates that this is the current active state and is used to enable action outputs from this state if the sequencer has this state as its currently active state.

At most one clause can signal a positive high true result to the action qualifier block. In this block, the single input will cause possibly multiple output events to be set. This is all controlled by each clause's Action Control Register settings.

At the top level, each state's action outputs will be OR'd together to create the final action bits that go to either the counter/timers, the sequencer (indication of next state) or the primary outputs.

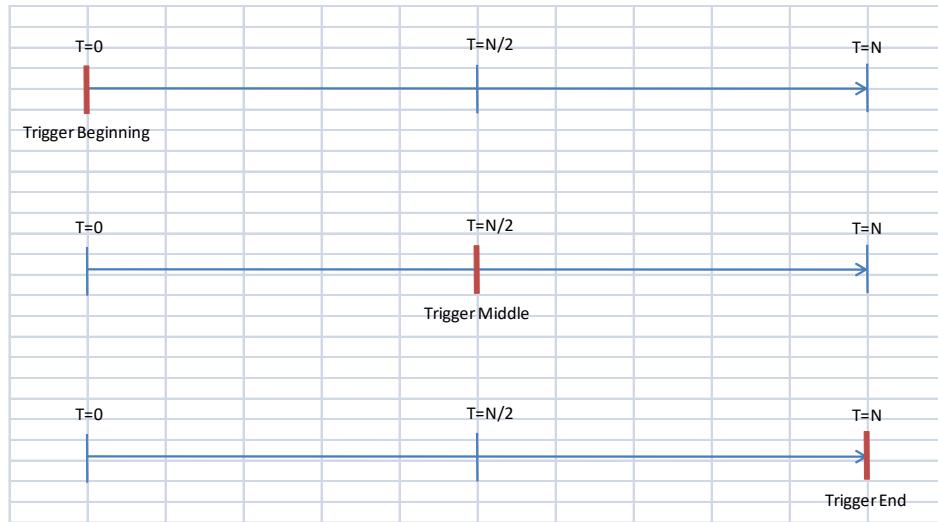
## 10.1.6 Global Triggering Considerations

### 10.1.6.1 Trigger Location Within Trace Capture

In most trace capture situations, the user wants to be able to place a trigger point at some programmable place within the trace. There are three baseline trigger locations that are useful: trigger at the beginning of trace, trigger in the middle, and trigger at the end. CTS supports each of these three, and any variation in between.

Trigger at beginning of trace is when a trigger event occurs and storage starts and fills up the entire selected storage space. Therefore the trace shows a trigger location at time = 0 and the rest of the trace follows. Trigger at the middle of the trace is when a trigger event occurs and storage continues for  $\frac{1}{2}$  the storage size, which allows for  $\frac{1}{2}$  the data to be previous to the event and  $\frac{1}{2}$  the data after the trigger event. Trigger at the end of the trace is when a trigger event occurs and storage halts immediately, such that all data stored is data previous to the trigger event.

**Figure 38: Trigger Locations**



#### 10.1.6.2 Circular Trace Buffer

Trace capture systems normally support the concept of a circular trace buffer, which is a buffer of a given size that is allocated and used in a circular manner until a trigger occurs. The circular trace buffer allows capture and storage of information prior to the trigger (trigger at end of trace), and then optionally continue capturing after the trigger until the post-trigger storage counter has counted down.

#### 10.1.6.3 Trigger or Marker Indications in Trace

The user needs to know the trigger location's position in the trace. The Trig\_Out signal action output signal is used to communicate this trigger from the CTS to the storage logic. Trace Sources can then insert a TRIG packet into their trace stream to indicate the position of the trigger relative to their trace stream.

#### 10.1.6.4 Capture Abort

A mechanism to abort the capture after triggering is required. For cases where post-trigger storage is selected, but due to the nature of the traffic, source selection or filtering, storage is not proceeding or proceeding slowly. The tool would set the bit (or bits, or sequence of actions) necessary to stop the trace collection in response to user GUI selection. This feature is typically used when a post-trigger storage counter value is set but no progress is being made toward filling the storage memory (user knows this as tool indicates fill progress by querying and displaying the value of the post-storage counter). This would be a function of the storage subsystem rather than the CTS.

#### 10.1.6.5 Repetitive Triggering

A typical LA feature, repetitive triggering provides storage at each occurrence of a trigger. After a trigger occurs, the storage completes, and the trigger re-enables. This feature can be simulated using storage qualification by enabling storage and starting a timer based on the event of interest, stopping storage when the timer expires, and then going back to searching for the event of interest. This too would be a function of the storage subsystem. The CTS can support this by configuring the state sequence and matching to provide multiple Trig\_Out indications. This is because the Trig\_Out action is independent of the



next\_state action due to any clause's match. This means that a clause could set Trig\_Out, then go back to state 0 for example (not forced to go to IDLE) to continue sequencing. Subsequent Trig\_Out conditions could then be matched upon.

#### 10.1.6.6 Power State Survival & Power Concerns

There are multiple low power states in a system and they are all implementation-specific. Low power states that reduce the voltage or frequency should be survivable by the CTS. The CTS does not provide the hardware hooks to self-correct from loss of power. Re-starting the CTS after a loss of power requires software or firmware support. Hooks have been implemented in the CTS to allow capturing of the state, forcing back in this state, and re-starting after a power down event.

The procedure would be that the software/firmware will capture the state of the CTS's status registers before power is lost and save the state somewhere in non-volatile memory. A snapshot of the state of the CTS can be captured by setting the Update\_Status register then clearing it. In the cycle that Update\_Status is cleared, the state of the CTS will be held in the shadow status registers as they will quit updating. The software/firmware can then read all the CTS status registers without them changing during the process.

Upon resumption of power, the software/firmware writes the previously-saved state back into CTS. The CTS status registers are unique in that they are shadow copies of the functional registers that act as both readable status registers during normal operation, and as writeable registers that will force the state of the functional registers opportunistically. The software can set the Force\_State bit along with the Sequencer\_Enable register to force the state of the CTS and bypass the normal startup to State 0. Forcing the state of the CTS will also force the state of the counters, timers, and output signals to the value of their status registers. While going through this restore sequence, the Update\_Status register bit must be cleared to ensure the integrity of the status CSRs that will be forced back into the CTS.

This concept of re-starting after a power state is tricky, and there are many corner cases. One caution is that any CTS activity that occurs after the snapshot of the CTS state is taken will not be visible because the status registers quit updating when the Update\_Status bit is cleared. In other words, the state of the CTS will be restored to this snapshot taken of the state.

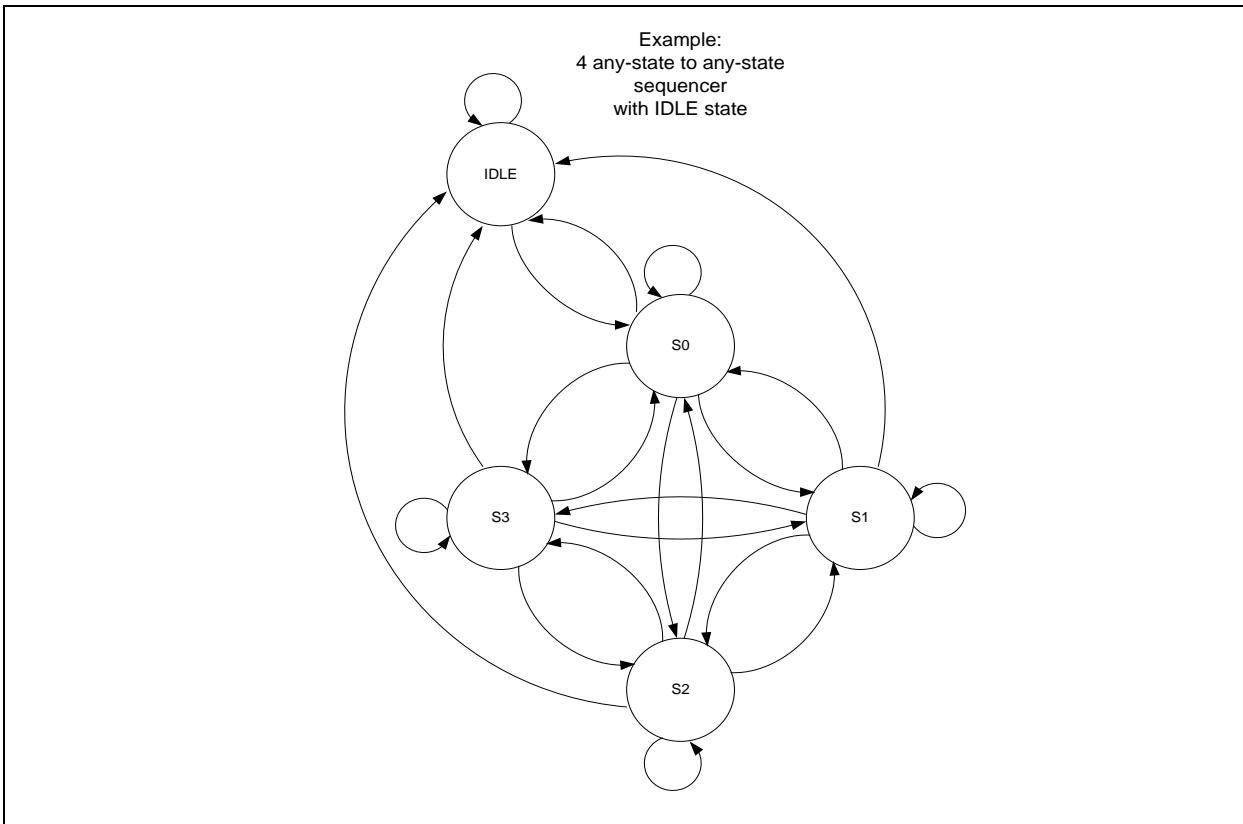
## 10.2 CTS State Machine

The promoted re-use and flexibility of the CTS is supported through modularization and parameterization. The number of states and clauses of the CTS State Machine are parameters that can be set to meet the target requirements. The maximum number of functional states for the CTS is 8 (plus an idle state) and the maximum number of clauses per state is 6 (plus the default clause). The logic for each clause of each state is identical. They are configured in a way to create a prioritized if/then/else if structure per state, as well as a state sequencer at the top level.

Figure 39 shows an example four-state trigger sequencer machine. It is a Finite State Machine programmed through a set of control registers that is used to create complex triggers based on a number of input variable match conditions. Each clause of each state of the sequencer has all input events and all the internal counter/timer events available as input conditions to match against. Upon a match condition, any number of available actions can be taken and any next state can be defined. All states of the sequencer share the global

counter/timers. These counter/timers can be configured to either count events or measure time. Configured as a counter, the counter/timer can be used to count match events before either moving on to the next state of the sequencer, or signaling one of the selectable output actions. Configured as a timer, the counter/timer can be started, stopped or re-started based on any match event and used to measure or accumulate time values (note that stopping and subsequent re-starting of a timer is only available with LCTs). Because of the possibility for trigger scenarios to look for a number of different conditions per state, there will be up to six clauses per sequencer state – an “if event W, else if event X, else if event Y, else if event Z ... etc.”. Because the user will most likely not want to use the same exit condition for each of the clause conditions, each clause has the ability to choose any or all of the actions and next state arc on the clause match (shown as W, X, Y, and Z above). Each clause of each state of the sequencer can go to any other state of the sequencer, including back to the same state upon a match event.

**Figure 39: Sequencer Block Diagram**

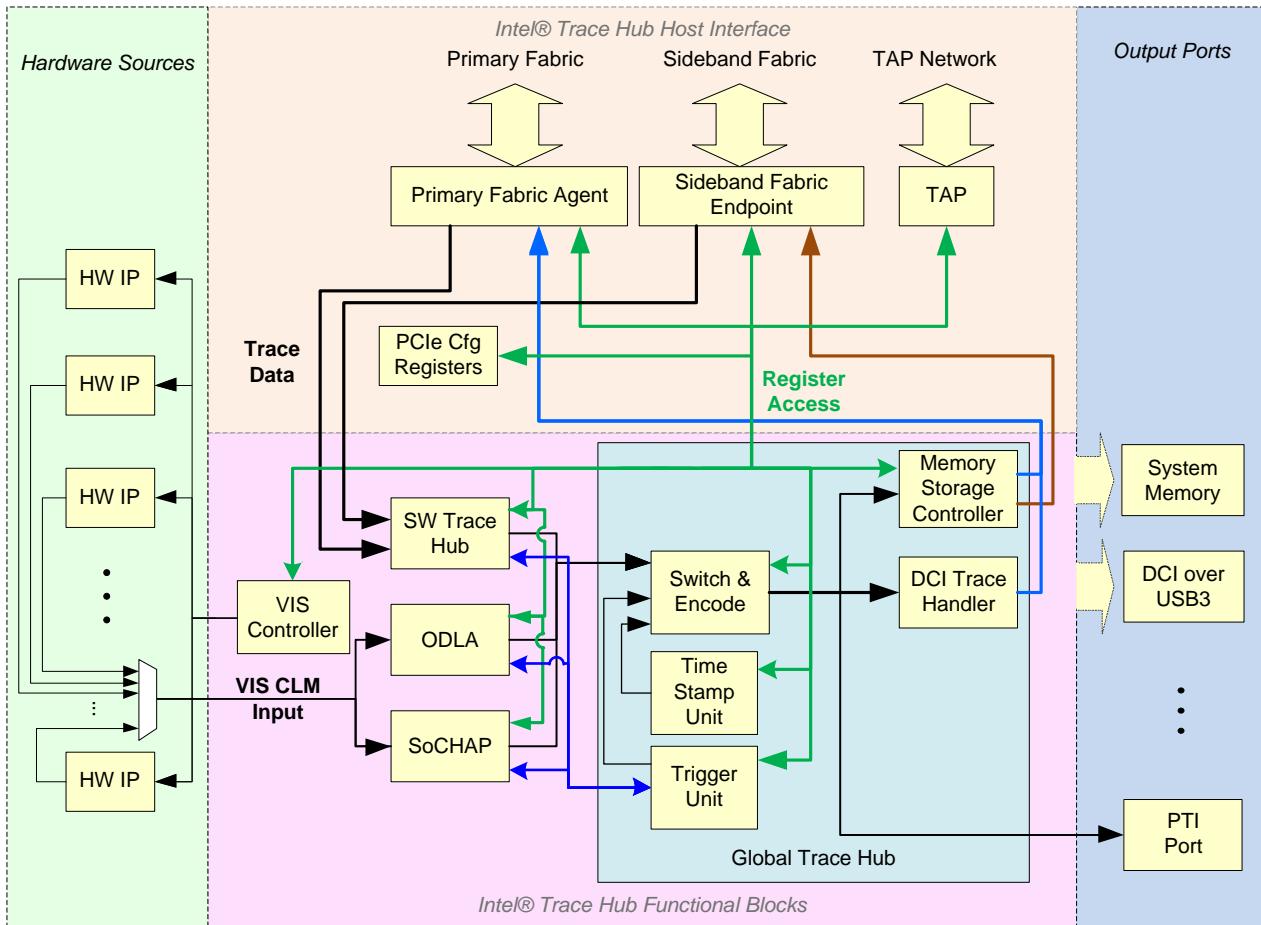


## 11 Parallel Trace Interface Port

The Parallel Trace Interface (PTI) port is an ingredient of the Intel Trace Hub Architecture shown in Figure 40. The Intel Trace Hub architecture is designed to support multiple trace destinations for the collected trace data. Some destinations are on-chip, like system memory, while others are off-chip, like the PTI port which is the topic of this chapter.

The MIPI-PTI is a standard interface defined by the MIPI Alliance<sup>10</sup>; it is composed of a parallel output port used to send trace data off-chip for capture by external hardware tools.

**Figure 40: Intel Trace Hub Architecture High-Level Diagram**



<sup>10</sup> The Parallel Trace Interface standard is defined and maintained by the Mobile Industry Processor Interface (MIPI) alliance working group.



## 11.1 PTI Port Software Support

The Intel Trace Hub's PTI port configuration registers are visible to software through Intel Trace Hub's PCI device. They are accessible through all interfaces supported by the Intel Trace Hub, including the primary fabric interface, sideband fabric interface, and TAP network.

The PTI port can be completely enabled or disabled via software running on target, or through software running a Debug and Test System (DTS) on a JTAG-based interface.

## 11.2 PTI Port Calibration/Training Sequence

Support for MIPI-PTI specified calibration/training patterns generated by an internal pattern generation block to be used for calibrating external capture devices.

## 11.3 Functional Description

### 11.3.1 Intel Trace Hub Trace Transfer Through the PTI Port

The Global Trace Hub (GTH) is the main trace collector for the Intel Trace Hub architecture. It is responsible for collecting data from the various sources, encoding the data into industry standard STPv2.1 (System Trace Protocol version 2.1)<sup>11</sup> and optionally time stamping it with the Time Stamp Correlation Unit. If not already time stamped, it can enable correlation of the occurrence of software and hardware events.

The GTH supports multiple trace-destinations for transferring the collected data to either an on-chip or off-chip location. The trace destination addressed in this chapter is the PTI port. In version 2 and later of the Intel Trace Hub architecture, the PTI Port controller is built-in to the Low Power Path (LPP). Trace data is formatted by the PTI port controller into PTI packets, after which the data is then forwarded to the General Purpose Input/Output (GPIO) pins for transmission. Moreover, the PTI port controller/formatter can also down-scale the Intel Trace Hub clock frequency if required by the specific GPIO technology.

On the Debug and Test System (DTS) side, the debug host PC or workstation is responsible for processing trace data received by the DTS to obtain an actual reconstruction of the system's behavior. It will include all the necessary APIs and the software stack to perform this task.

### 11.3.2 MIPI-PTI Protocol and Terminology

The MIPI-PTI protocol is an industry standard interface defined to export relatively high-bandwidth debug and trace data from the target system to the DTS through the DTC.

A PTI is a unidirectional, Double Data Rate (DDR), interface with a set of *PTI\_BPWIDTH/2* single-ended data lanes. The strobe/clock output in the MIPI-PTI specification is, in the remainder of this document, referred to as a dedicated output called *Trace Clock*. In addition to the above outputs, the Intel Trace Hub PTI defines an additional output known as the 'Trace Valid' signal (*trace\_vld*), which is asserted as long as there is valid data on the

---

<sup>11</sup> The MIPI STPv2 is the industry standard tracing protocol developed by the MIPI Alliance.



output pins. In VIS-bypass mode, the two dedicated outputs are used to drive the two VIS clocks while the data pins drive the VIS data.

It is also important to mention that in most of the use cases the DTS is the master controller for the PTI port and is responsible for providing control and coordination with the PTI port logic through one of the standard interfaces (e.g. JTAG interface, sideband fabric, or primary fabric). There do exist, however, some use cases where the SUT can configure itself for trace collection – in this case the DTS will have to be configured *a priori* to accept all trace data sent to it. Details of how the PTI port controller is configured, however, are beyond the scope of this document.

Finally, the PTI specification defines a *PTI packet* as the data transferred on the data lanes, *trace\_data*, on a single toggle of the trace clock or trace strobe (note that the *trace\_vld* signal is not required for a valid PTI packet).

#### 11.3.2.1 Clock Divider Logic

Since the Intel Trace Hub clock is usually one of the fastest clocks on an SoC, a slower clock may be required for data transfer on the GPIO pins. To this end, a clock divider is embedded in the PTI port controller logic that will generate such a clock. The PTI port controller clock divider supports ratios of 1, 2, 4, and 8.

#### 11.3.2.2 PTI Mode Mux

The Intel Trace Hub's PTI port supports 4 modes where each uses a subset of the GPIO pins to drive the trace data. A summary of these 4 modes and the GTH trace data to pin mapping is presented in Table 11-1.

If the GTH interface width is not an integer multiple of the number of data lanes enabled for trace data transfer, additional NULL packets will be added to the trace data under certain conditions. One such occurrence would be if the GTH is quiescent for a long enough period of time such that the PTI port internal FIFO does not have enough data to fill all of the enabled data lanes.

**Table 11-1: PTI Port Modes and Signal Mapping**

Mode/Pins	PTI 4-bit Mode	PTI 8-bit Mode	PTI 16-bit Mode
trace_data[3:0]	pti_data[3:0] pti_data[7:4] pti_data[11:8] pti_data[15:12] pti_data[19:16] pti_data[23:20] pti_data[27:24] pti_data[31:28]	pti_data[3:0] pti_data[7:4] pti_data[19:16] pti_data[23:20]	pti_data[3:0] pti_data[19:16]
trace_data[7:4]	N/A	pti_data[11:8] pti_data[15:12] pti_data[27:24] pti_data[31:28]	pti_data[7:4] pti_data[23:20]
trace_data[11:8]	N/A	N/A	pti_data[11:8] pti_data[27:24]
trace_data[15:12]	N/A	N/A	pti_data[15:12] pti_data[31:28]

### 11.3.2.3 PTI Transmission Controller

The PTI transmission controller is responsible for controlling the data flow and the data multiplexer based on the programmed configuration. It is responsible for generating the index signals driving the mux and generating the *trace\_vld* signal which is also used for generating the PTI trace clock (or strobe).

### 11.3.2.4 PTI Pattern Generator

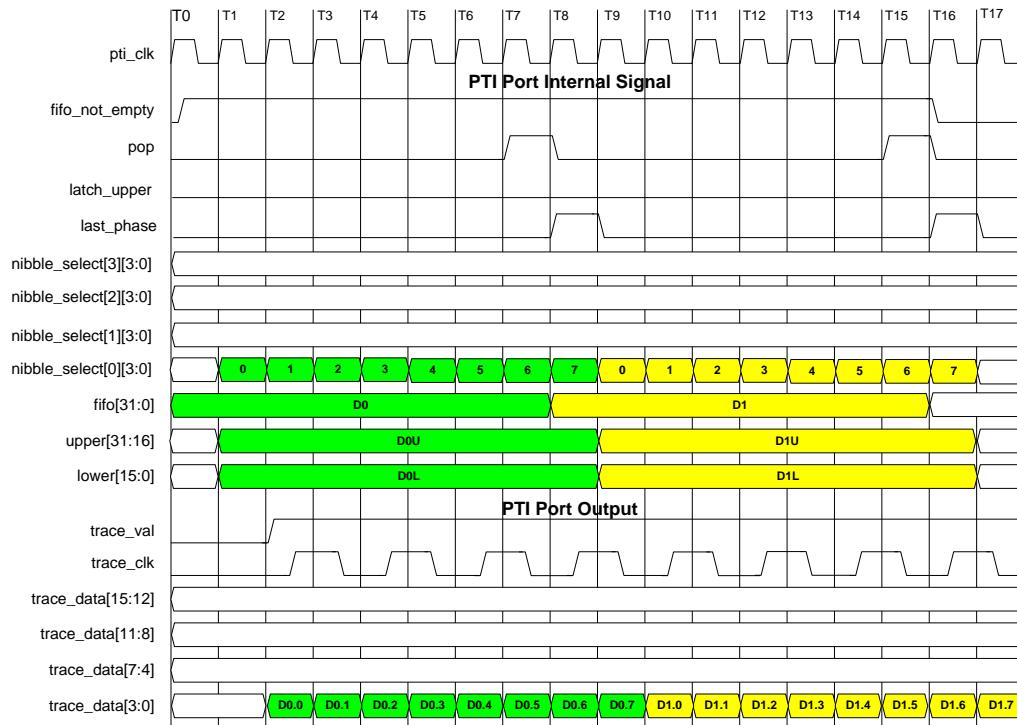
The MIPI-PTI standard requires a minimum of six training/calibration patterns that have to be supported by a PTI port as well as DTS. There is a dedicated block in the PTI port controller that is responsible for generating those five patterns based on the input configuration. A list of those patterns, their configuration bits, and use cases directly copied from MIPI-PTI specification are given in Table 11-2 below. Also note that the pattern will only be driven on the pins corresponding to the current configuration mode.

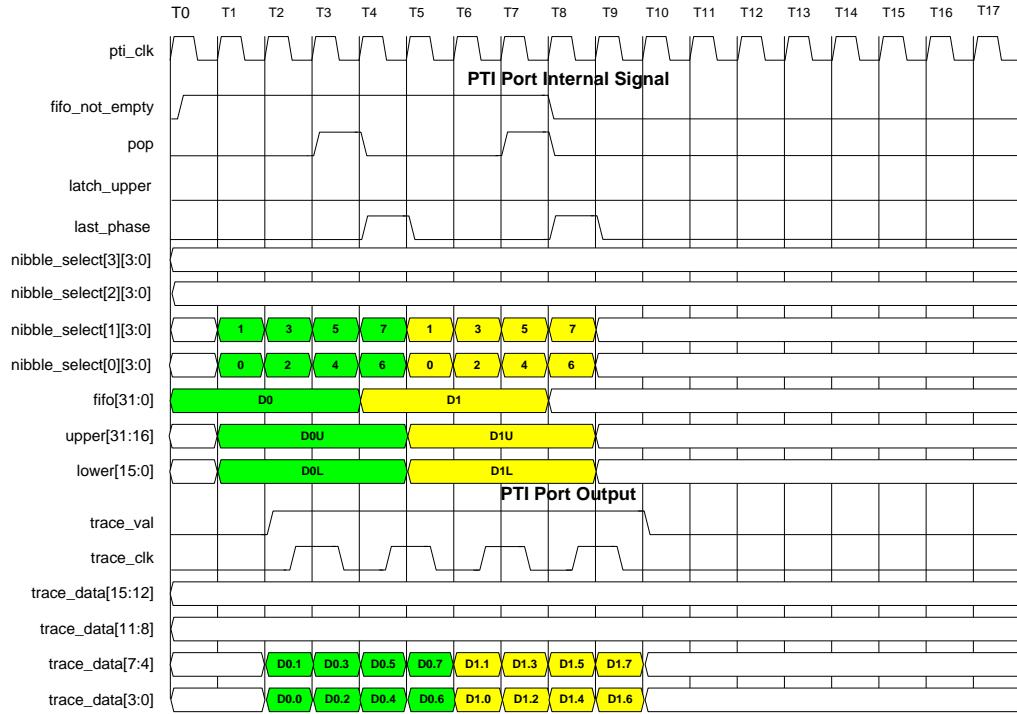
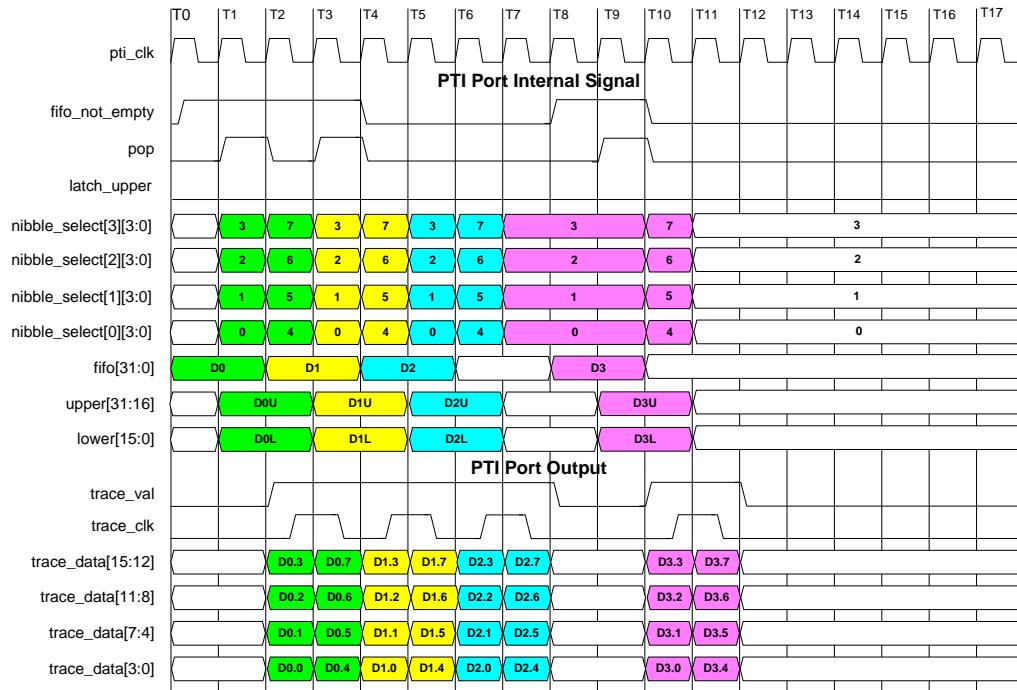
**Table 11-2: MIPI-PTI Training/Calibration Patterns**

<b>Pattern Number</b>	<b>Pattern Generation Config Mode</b>	<b>Packet Sequence</b>	<b>Use Case</b>
0	3'b001	<pre>trace_data[] all ones trace_data[] all ones trace_data[] all zeros trace_data[] all ones trace_data[] all ones trace_data[] all zeros ... </pre>	<p>Stresses power supply/ground of the board/chip by switching from all highs to all lows on the same clock (looking for ground bounces).</p> <p>Stresses pin to pin skew by driving all signals high for at least 2 bit periods (by ensuring the signals attain the maximum "high logic" voltages before a transition).</p>
1	3'b010	<pre>trace_data[] all zeros trace_data[] all ones Invert trace_data[0] Invert trace_data[] Invert trace_data[] Invert trace_data[1] and trace_data[0] Invert trace_data[] Invert trace_data[] Invert trace_data[2] and trace_data[1] Invert trace_data[] Invert trace_data[] ... Invert trace_data[N] and trace_data[N-1] Invert trace_data[] Invert trace_data[] ... </pre>	<p>Stresses chip electrical characteristics by driving a pattern that is dominated by high signals.</p> <p>Verifies signal mapping by walking an inverted bit across the interface.</p>
2	3'b011	<pre>trace_data[] all ones trace_data[] all zeros Invert trace_data[0] Invert trace_data[] Invert trace_data[] Invert trace_data[1] and trace_data[0] Invert trace_data[] Invert trace_data[] Invert trace_data[2] and trace_data[1] Invert trace_data[] </pre>	<p>Stresses chip electrical characteristics by driving a pattern that is dominated by low signals.</p> <p>Verifies signal mapping by walking an inverted bit across the interface.</p>

		Invert trace_data[] ... Invert trace_data[N] and trace_data[N-1] Invert trace_data[] Invert trace_data[] ...	
3	3'b100	trace_data[] all zeros trace_data[] all zeros trace_data[] all ones trace_data[] all zeros trace_data[] all zeros ...	Stresses power supply/ground of the board/chip by switching from all highs to all lows on the same clock (looking for ground bounces).  Stresses pin to pin skew by driving all signals low for at least 2 bit periods (by ensuring the signals attain the minimum "low logic" voltages before a transition).
4	3'b101	Up counter from '0's to all '1's and back to '0's	Pattern is more representative of real data.  Useful for timing training since it's very easy to look at for increments and wrap arounds.
5	3'b110	Alternating odd and even bits set/clear	Every bit switches every clock  Recommended for timing training since the check can be made right at the interface with simple logic.

Figure 41: PTI Packets (4-bit mode)



**Figure 42: PTI Packets (8-bit mode)**

**Figure 43: PTI Packets (16-bit mode)**




### 11.3.3 Programming Model

The PTI port has to be enabled before sending any trace data to it, or before using it to generate any of the calibration patterns. After enabling, it is recommended driving one or more of the six calibration patterns supported to allow the DTC to adjust its delays for data integrity on the PTI port. Once calibrated, the PTI port is ready to transfer data between the GTH and the DTC.

### 11.3.4 Performance/Bandwidth Analysis

The PTI port is capable of operating at the Intel Trace Hub clock frequency for a maximum possible bandwidth of  $(PTI\_BPBWID/2)$ -bits per Intel Trace Hub clock cycles. For example, if running the Intel Trace Hub clock at 400MHz with 16 data lanes, the PTI port can transfer data up to a bandwidth of 800MB/s. However, the actual output bandwidth of the PTI port is dependent on the GPIO capabilities and technology.



## 12 Host Interface

The Host Interface block provides the PCI device presence and memory-mapped IO (MMIO) interface of the Intel Trace Hub device. The primary fabric interface provides a high-bandwidth interface to the SOC, for both receiving high-bandwidth trace data, and writing STP-encoded trace data to memory (or to the USB3 DbC.Trace endpoint), while the sideband fabric provides an alternative, though lower-bandwidth, interface. The Host Interface block also provides a TAP block, allowing an external JTAG host device access to the same resources (PCI Configuration space, and MMIO space).

### 12.1 Fabric Multi-Root Space Support

The internal fabric (internal bus) provides support for multiple independent root spaces with their independent address spaces. In this context, a root space is a collection of resources using an independent address space that is orthogonal to all other root spaces. A typical agent is referred to as a single-root space aware is only accessible in its own root space and can only communicate with agents in that root space. On the other hand, an agent that is multi-root space aware is accessible from multiple root spaces (i.e. is mapped to each of the orthogonal address spaces), and can communicate with agent across root spaces. The mechanism by which an agent communicates with other agents across root spaces and how a transaction is routed is beyond the scope of this specification. The Intel Trace Hub is a multi-root space accessible device, accessible in two root spaces, with different addresses in each space.

For the *Host Space*<sup>12</sup> or Root Space 0 (RS0), the Intel Trace Hub implements four programmable BARs summarized in Table 12-1. For Root Space 1 (RS1), the Intel Trace Hub can be accessed via two fixed memory windows ("BARs") as summarized in Table 12-2. In addition, the Intel Trace Hub can target (perform memory writes) to devices in root space 0. Specifically, the Intel Trace Hub is designed to write to system memory, and also to a USB controller with a DbC.Trace endpoint. Because of this functionality, the Intel Trace Hub can function as a "bridge" between RS1 and RS0. This may have security or isolation implications, in the case of running multiple virtual machines on a single physical machine. Operating Systems and Virtual Machine Monitors should take appropriate precautions when assigning the Intel Trace Hub to a single virtual machine.

---

<sup>12</sup> The Host Space is the name used to denote the main IA CPU root space.

**Table 12-1: Host Space Base Address Registers**

<b>Intel Trace Hub MMIO Region</b>	<b>Host Space Base Address</b>
<b>CSR and MSC Trace Buffer (MTB) Region:</b> The CSR/MTB region is a fixed 1MB of MMIO space that includes the memory mapped CSRs, plus the MSC Trace Buffers (see Figure 2 for more details).	CSR_MTB_BAR programmable configuration space register – <b>BAR0</b>
<b>Software Trace Memory Region (STMR):</b> the STMR is a variable size (project specific) MMIO space used by software/firmware to send debug messages to the Intel Trace Hub. For more details please refer to Chapter 4	SW_BAR programmable configuration space register – <b>BAR1</b>
<b>Intel PT Trace Memory Region (RTMR):</b> RTMR is a variable size (project specific) MMIO space used by Intel PT hardware to send debug messages to the Intel Trace Hub. For more details please refer to Chapter 4 .	RTIT_BAR programmable configuration space register – <b>BAR2</b>
<b>Firmware Trace Memory Region (FTMR):</b> FTMR is a variable size (project specific) MMIO space used by firmware to send debug messages to the Intel Trace Hub. For more details please refer to Chapter 4	FW_BAR programmable configuration space register – <b>BAR3</b>

**Table 12-2: Root-Space 1 (RS1) Base Address Registers**

<b>Intel Trace Hub MMIO Region</b>	<b>RS1 Space Base Address</b>
<b>CSR and MSC Trace Buffer (MTB) Region:</b> The CSR/MTB region is a fixed 1MB of MMIO space that includes the memory mapped CSRs, plus the MSC Trace Buffers (see Figure 2 for more details).	0xF740_0000 ( <b>BAR0</b> )
<b>Firmware Trace Memory Region (FTMR):</b> the FTMR is a variable size (project specific) MMIO space used by firmware to send debug messages to the Intel Trace Hub. For more details please refer to Chapter 4 .	0xF700_0000 ( <b>BAR3</b> )

## 12.2 Error Handling and Reporting

The Intel Trace Hub is a standard PCI device that supports error logging and reporting per the PCI and internal bus specifications. In this section, error handling and reporting are presented.

From a transaction-level point of view, the Intel Trace Hub is responsible for handling three types of erroneous transaction types. Those three types can be summarized as follows:

1. **Unsupported transactions that are always unsupported:** Any transaction that does not have the *type*, *format*, and *traffic class*, supported by the Intel Trace Hub is considered always unsupported.
2. **Supported transactions that violate the Intel Trace Hub programming model:** These transactions are supported and have the correct type, format, and traffic class, but might be violating the Intel Trace Hub programming model or some specific Intel Trace Hub requirement based on the current Intel Trace Hub device configuration and state.
3. **Erroneous downstream completions:** This is a completion transaction received by the Intel Trace Hub in response to an outbound non-posted request with a completion status other than successful indicating some error has occurred.

Depending on which of the above transaction type is received, the Intel Trace Hub will respond and handle the exceptions differently as explained below.



### 12.2.1 Always Unsupported Requests

Unsupported Requests (UR)s are the category of transactions that are always unsupported by the Intel Trace Hub independent of the state of the device. The action taken by the Intel Trace Hub in response to the fabric issuing a command put for an always unsupported request depends on the transaction type as listed in Table 12-3.

Examples of always unsupported requests:

1. A command put for an IOWr or IORd transaction.
2. A command put for a transaction with an IMPS larger than 64B.
3. A command put for a CfgWr transaction with wrong bus/device/function numbers.

**Table 12-3: Unsupported Request Error Handling**

Request Type	Error Condition	Error Logging	Error Reporting
Posted	Unsupported Request	Sets Unsupported Request Detected (URD) bit in Device Specific PCI Status register	If both the Unsupported Request Reporting Enable (URRE) bit in Device Specific Control register and the SERR Enable bit in PCI Command register are set, then the Signaled System Error (SSE) bit in PCI Status register will be set and a DO_SERR message will be sent on the sideband interface
Non-Posted	Unsupported Request	No	The Completion will be returned with an Unsupported Request (UR) status

### 12.2.2 Programming Model Unsupported Requests

If the Intel Trace Hub receives a command put for a transaction targeting the Intel Trace Hub (it asserts hit or would have asserted hit if using target decode) but the transaction violates the Intel Trace Hub's programming model, it will have to be handled as a Completer Abort (CA).

Examples of supported requests that violate the Intel Trace Hub programming model:

1. A command put for a Cfg transaction with a payload larger than 1DW.
2. A command put for a MWr/MRd transaction with MSE not set.
3. A command put for a MWr/MRd to a 32-bit CSR that is not DW aligned.

It is also important to note that even when using target decode a transaction that violates the Intel Trace Hub's programming model might still be received due to the target pipelined architecture. For example, consider two back-to-back transactions: the first is a CfgWr that clears the MSE bit, and the second is a MWr. Due to the pipelined nature of the agent, both requests would be accepted, but by the time the device is ready to service the MWr, the memory space-enable bit would have been cleared and the second transaction has to be a CA. The action taken by the Intel Trace Hub in response to a transaction that violates its programming model depends on the type of the transaction as listed in Table 12-4.

**Table 12-4: Completer Abort Error Handling**

Request Type	Error Condition	Error Logging	Error Reporting
Posted	Completer Abort	Sets Signaled Target Abort (STA) bit in PCI Status register	In PCI Command register, if SERR Enable bit is set, then the Signaled System Error (SSE) bit in PCI Status register will be set and a DO_SERR message will be sent on the sideband interface



Request Type	Error Condition	Error Logging	Error Reporting
Non-Posted	Completer Abort	Sets Signaled Target Abort (STA) bit in PCI Status register	The Completion will be returned with a Completer Abort (CA) status

### 12.2.3 Erroneous Downstream Completions

The third type of erroneous transaction that is handled by the Intel Trace Hub relates to downstream completion transactions (generated in response to outbound non-posted transactions from the MSU). In this case, any completion with a status other than "successful" is considered erroneous and has to be handled by the Intel Trace Hub Host Interface. This is necessary to avoid corrupting system memory and even possibly hanging the entire system.

To this end, once a completion transaction is received by the Intel Trace Hub in response to an issued non-posted transaction with any status other than successful, the primary fabric target agent will immediately signal the MSU to take the appropriate action to prevent any catastrophic events from occurring. The detailed MSU behavior is documented in the MSU chapter of this specification. From an Intel Trace Hub device perspective, Table 12-5 shows how such an error will be logged in PCI configuration space based on the completion status.

**Table 12-5: Unsuccessful Completion Error Handling as a Requestor**

Request Type	Completion Status	Error Logging	Error Reporting
Completion	Completion Abort	Sets Receive Target Abort (RTA) bit in PCI Status register.	MSU specific. Please consult the MSU specification for a detailed description.
Completion	Unsupported Request	Sets Receive Master Abort (RMA) bit in PCI Status register.	MSU specific. Please consult the MSU specification for a detailed description.

### 12.2.4 Miscellaneous Downstream Erroneous Transactions

If a downstream transaction does not fall in any of the above categories but can still cause some functional problem, like trying to access a non-existent register or reading a write-only register, the internal bus and PCI specifications require that the device still service the transaction. If this erroneous transaction is a posted transaction, it will silently be dropped. If it is a non-posted transaction, then a data payload of zero will be returned along with a status of Successful Completion (SC).

Examples of miscellaneous erroneous transactions:

1. A MWr transaction targeting the RO MSC Trace Buffer (MTB).
2. A MRd to an empty region in one of the Intel Trace Hub's MMIO regions.

To deal with the above types of erroneous transactions, a different approach was needed; the internal fabric agent does not have this level of knowledge about the other Intel Trace Hub components. For example, the primary fabric target agent does not know if the MRd transaction it received is targeting a non-existent memory region. Hence, the Intel Trace Hub implements a timeout mechanism that is used to comply with the PCI specification requirements, while also acting as a survivability feature that can prevent internal hangs and blockings.



The timeout mechanism is composed of two timers at the main points of arbitration on the Intel Trace Hub internal bus architecture. The first is located at the Inbound Switch arbiter and the second at the Inbound Clock Crossing Bridge arbiter.

Once a request is put on the corresponding bus (IPCLK Inbound Bus for the Inbound Switch and the NPCLK Inbound Bus for the Inbound CCB), the timer starts to be decremented. For posted requests, if the timer expires before the targeted block accepts the request, it is immediately dropped to free the bus and accept the next transaction. For non-posted requests if the timer expires before the response has been returned by the targeted unit, the timer will return a successful completion on behalf of the targeted unit and accept the next non-posted request (since there can only be one outstanding non-posted request at any given time).

Both timers are programmable through the Intel Trace Hub PCI configuration space to accommodate for different functional block latencies. For example, the VIS Register Controller block can be running much slower than other units and might require a much longer time to respond to a given request. For this reason it is crucial that software users make sure to program the values for those timeout timers to a large enough value that would prevent any false timeouts, yet a small enough value that wouldn't unnecessarily degrade the overall system performance.

For some use cases, the Intel Trace Hub needs to backpressure the internal bus fabric which, in turn, can backpressure the source of the transaction. In this case, the Intel Trace Hub also provides the ability to turn off one or both of those timeout timers by programming them to zero.

It is also important to note that the larger the values programmed into those timeout timers, the larger performance degradation that will be sensed by software. This is, however, a good indication that software is not performing the correct operation and should be debugged.



# 13 Programming Model

A detailed programming model of all of Intel Trace Hub features for all use cases is beyond the scope of this document. In this section, we present a suggested programming sequence.

In general, the recommended programming/configuration sequence for the Intel Trace Hub hardware is:

1. Trace destinations.
2. Trace sources.
3. Map sources to destinations (configure Global Trace Hub).
4. Configure triggering.

More details for each of the above steps are given in the following subsections.

## 13.1 Configuring Trace Destinations

Depending on the use-case, one or more Intel Trace Hub tracing destinations must be configured as described below.

### 13.1.1 Trace to PTI

Configuring PTI as a trace source is relatively simple. The user will select the operational mode (i.e. 4-bit, 8-bit, or 16-bit), set the SMU maintenance packet frequency for this port, GTH data retention policy, NULL packet generation policy, and then enable PTI port.

If training is also required, the PTI port controller supports generating some test patterns for testing the capture tool. Training is a fully manual operation. That is, there is not a pre-defined training sequence (in hardware terms) for PTI. The user/software can select a training pattern to send to the PTI port. The user would then verify proper capture of that training pattern. Once verified, the user can select a different training pattern. Once the user is satisfied with PTI port training, he/she can disable training and proceed with trace capture.

### 13.1.2 Trace to DCI, BSSB Mode

If trace data will be sent through BSSB, the Intel Trace Hub DCI Handler has to be configured to BSSB mode (refer to chapter 9 for details). The BSSB-related hardware (DCI Bridge and USB PHY) must also be initialized, though that is out of scope for this document. Both MSCs must be put into DCI mode, and enabled, since they control the source side of the local trace buffer that stores the trace data. As with other trace destinations, the SMU packet frequency should be selected, as well as NULL packet generation and GTH data retention policy (drop/no-drop).

Generally, the configuration sequence for this mode is:

1. Configure the DCI Bridge and BSSB components to stream trace out of the BSSB port (details are outside the scope of this specification).
2. Program MTB Base pointer and DCI TH Credit return register offsets into the appropriate registers of the DCI Bridge.
3. Configure the DCI TH with DCI Base pointers / offsets.
  - a. EXIBASEHI, EXIBASELO registers



- b. STREAMCFG1, STREAMCFG2 registers
4. Set DCI TH Timeout values
5. Configure both MSCs for DCI Mode
6. Clear the SETSTRMMODE bit (write 0), and set the ENABLE bit in the STREAMCFG1 register

### 13.1.3 Trace to DCI, DbC (USB) Mode

When tracing to USB (specifically, the DbC.Trace endpoint within the xHCI controller), the USB controller should be discovered and configured first. Next, the Intel Trace Hub DCI Trace Handler must be enabled to run in DbC.Trace mode, and initialized with pointers to the DbC.Trace endpoint; see Section 9.2.1 for details. As with other trace destinations, the SMU packet frequency should be selected, as well as NULL packet generation and GTH data retention policy (drop/no-drop).

Generally, the configuration sequence for this mode is:

1. Configure the XHCI.DBC Controller for operation as a DbC.Trace endpoint (details are outside the scope of this specification)
2. Write MTB Base address pointer, DCI TH Credit return register offset into the appropriate registers of the XHCI.Dbc Controller.
3. Configure the DCI TH with DBC Base pointers / offsets
  - a. DBCBASEHI, DBCBASELO registers
  - b. STREAMCFG1, STREAMCFG2 registers
4. Set DCI TH Timeout values
5. Configure both MSCs for DCI Mode
6. Set the SETSTRMMODE bit and set the ENABLE bit in the STREAMCFG1 register.

### 13.1.4 Trace to System Memory, Single-Block/CSR-Driven Mode

1. When tracing to system memory, the first thing that must be done is to allocate system DRAM. For single-buffer, or CSR-driven, mode, a single large, physically contiguous block of system memory must be allocated (typically by the system BIOS).

---

**NOTE:** For highest performance, this version of the Intel Trace Hub writes trace data to memory with the “no-snoop” attribute set. In order to properly accommodate this system performance feature, the agent allocating memory for the Intel Trace Hub must allocate non-cacheable memory. Alternatively, the agent can allocate cacheable memory, but then must ensure the CPU’s cache is invalidated before the Intel Trace Hub writes data to that memory.

---

2. For each MSC, software should write the base address and buffer size for its corresponding memory block in the MSCnBAR and MSCnSIZE register.
3. If interrupt-on-complete operation is desired, software must set up an ISR for the MSU error handling MSI/INTx.
4. As with other trace destinations, the SMU packet frequency should be selected, as well as NULL packet generation and GTH data retention policy (drop/no-drop).
5. Enable the desired MSC(s) by setting the MSCnEN bit, and writing the corresponding value for single-block/CSR-driven mode to the MSCnMODE bits. Options such as wrap\_enable may also be programmed prior/simultaneous to setting the enable.



6. When the msu\_interrupt is received indicating capture done, software should read the current memory write pointer value from CMWPn and the wrap status from the WRAPSTATn bit to determine the valid trace data in the memory block.

### 13.1.5 Trace to System Memory, Multi-Block/Linked-List Mode

1. Software should allocate one or more regions of memory to hold the memory blocks for the trace data/linked-list. Software writes the first 32 bytes in the header to configure the address, size and other attributes for each memory block. Software should also clear to 0 the 32 bytes of the hardware header for each memory block so that it can easily identify non-zero values as those written by the MSU.

**NOTE:** For highest performance, this version of the Intel Trace Hub writes trace data to memory with the “no-snoop” attribute set. In order to properly accommodate this system performance feature, the agent allocating memory for the Intel Trace Hub (e.g., a PCI device driver) must allocate non-cacheable memory. Alternatively, the agent can allocate cacheable memory, but then must ensure the CPU’s cache is invalidated before the Intel Trace Hub writes data to that memory.

There may be further attributes or conditions required for a particular OS to ensure that the Intel TH PCI Driver and Intel TH hardware are able to both utilize the physical address of the memory block(s) allocated (e.g., disallowing paging, locking memory, etc.). Such details are beyond the scope of this specification.

2. Software should write the address for the first memory block of the first memory window (the head of the linked-list) to the MSCnBAR register.
3. If interrupt-on-complete operation is desired, software must set up an ISR for the MSU error handling MSI/INTx.
4. As with other trace destinations, the SMU packet frequency should be selected, as well as NULL packet generation and GTH data retention policy (drop/no-drop).
5. After setting up all the block headers in memory and programming the first memory block address, software should enable one or more desired MSC(s) by setting the MSCnEN bit, and writing the corresponding value for multi-block/linked-list mode to the MSCnMODE bits.
6. When the msu\_interrupt is received indicating capture done, software can read the hardware header in each block of each linked-list/window to determine where the valid trace data and other attributes have been linked.

### 13.1.6 Internal Buffer Mode

1. Enable the desired MSC(s) by setting the MSCnEN bit, and writing the corresponding value for internal buffer mode to the MSCnMODE bits. Options such as wrap\_enable may also be programmed prior/simultaneous to setting the enable.
2. When the capture is complete, software can read the current write pointer from the MSCnTBWP register and the wrap status from the WRAPSTATn bit to determine the valid trace data in each MTB. Software can then directly read the trace data content from each MTB (which is memory-mapped).
3. In internal buffer mode, the MSC responds to inbound read transactions with the addressed trace buffer data. The MSC supports a burst length of 1 on these read transactions (that is, the MSC will not respond to a burst read to the trace buffer). Each trace buffer is 256 entries deep. Each entry is either 8 bytes or 16 bytes wide, depending on MD\_WIDTH. If MD\_WIDTH = 64 (8 bytes), then MTB0 is addressed at

address offset 0x000 to 0x7FF and MTB1 is addressed at address offset 0x8000 to 0x87FF. If MD\_WIDTH = 128 (16 bytes), then MTB0 is addressed at address offset 0x000 to 0xFFFF and MTB1 is addressed at address offset 0x8000 to 0x8FFF. Note there are offsets not shown that are based on the CSR\_MTB\_BAR (see Chapter 12 ).

---

**NOTE:** Reading the MTBs in internal buffer mode while trace data is being stored in the MTBs is not supported. There is no way to prevent the MTB from under-run (reading more data than is actually present). Furthermore, the WRAPSTAT function will not function properly, due to the fact that reading from the MTB changes the read pointer value, which is used in the full and empty calculations.

---

## 13.2 Configuring Trace Sources

Depending on the use-case, one or more Intel Trace Hub trace sources must be enabled and configured to capture/forward trace data to the desired trace destination.

### 13.2.1 Visualization of Internal Signals (VIS)

The VIS Register Controller is used to configure the VIS mux tree to route signals from their sources into the Intel Trace Hub.

### 13.2.2 On-Die Logic Analyzer (ODLA)

The ODLA captures VIS signals into ODLA packets.

### 13.2.3 SoC CHAP (SoCHAP)

The SoCHAPs take VIS signals as input set up in a previous step, and count them according to how they are configured

### 13.2.4 Software Trace Hub (STH)

There is little to set up in the STH to enable tracing software and firmware trace sources; most of the "enabling" is done when configuring the trigger unit. Event generation is the key item to configure, as it will inform the trigger unit of interesting data coming through the STH.

The user/software should set event match/mask registers as needed for event(s) of interest that will come into the STH. Then set up the TRIG\_TS Packet payload (TRIGTSPP), if such will be utilized. Thirdly, set up the Backpressure Duration Counter (see STH chapter for more details). Finally, clear the DPLC.DPLCO if necessary.

## 13.3 Mapping Trace Sources to Trace Destinations

Mapping sources to destinations involves two fundamental steps: choosing which source(s) to send to which destination(s), and then writing SWDEST bits in GTH CSRs accordingly. The algorithm or method by which software does this is outside the scope of this specification.

Within the GTH itself, the TSCU destination needs to be chosen to be useful to decode software: it should be sent to same destination as software trace data that will use the



TSCU time correlation packets (i.e., trace data that is time-stamped with the CPU's time stamp).

Next, Enable or Disable trace sources for masters 0-256, as appropriate for the debug scenario.

Set Grant Duration for each source. Also set the Low Water Mark for each source; the LWM must be set to one less than Grant Duration.

Set storage mode for each source (single or multi-buffer).

Set BPB Transmit Threshold (BPBTXLV) for each output port.

## 13.4 Configuring the Trigger Unit

Configuring the Trigger unit can be as simple or as complex as the user desires, from a simple enable-all-sources-immediately, to a complex trigger with multiple start and stop conditions. A full treatise on Trigger Unit configuration is outside the scope of this document. Nonetheless, the following is a brief summary of the two basic methods:

### 13.4.1 Using the Trigger Unit

Configure the states, events, and clauses, according to the sequence needed for achieving the desired trigger condition. Actions in the various states and clauses will likely include enabling one or more trace sources, counting a certain number of clocks before or after taking a certain action, and counting clocks after a trigger to limit the amount of post-trigger data that is captured, among others.

If storing trace data to memory or USB.DbC.Trace, ensure the "Capture Done" signal (store qualifier #0) is asserted at the end of tracing, so that all trace data can be flushed out of the Intel Trace Hub's internal pipe and buffers, and sent to the destination. In the case of memory, captureDone assertion can also interrupt the main CPU, indicating it is time to post-process and display the captured data.

### 13.4.2 Manual Override

The "manual override" mode is the equivalent of setting the trigger specification to "If Anything, start storing immediately", and "stop storing when I tell you." This is accomplished by software setting the *storeEn* override bits (GTH register SCR) for each trace source to be enabled, and starts the flow of the trace data.

Stopping the flow of trace data is most easily accomplished by software setting the appropriate ForceStoreEnOff bits in the SCR2 register. Using this method also allows software to manually assert the "Capture Done" signal, which will cause all data in the Intel Trace Hub pipeline and buffers to be flushed to the destination.

See section 6.2.9.1.1 for more information.

### 13.4.3 Intel Trace Hub Software Reset

The Intel Trace Hub supports a software-visible configuration register that enables resetting the Intel Trace Hub logic to an uninitialized state through a software command. The logic blocks that will be reset if this register is written by software are the STH, GTH, Trigger Unit, TSCU, ODLA, SoCHAP, and portions of the MSU.



The software reset can be performed through any of the three interfaces, the primary fabric, the sideband fabric, or TAP. Once the “reset command” is received by the Intel Trace Hub, the reset takes place immediately.

To summarize, the following will occur when a software reset is detected:

- Reset all MMIO control registers.
- Reset all functional logic in the Intel Trace Hub Functional Blocks.
- Reset all the PCI configuration space registers except the following:
  - MTB\_CSR\_LBAR and MTB\_CSR\_UBAR registers.
  - SW\_LBAR and SW\_UBAR registers.
  - RTIT\_LBAR and RTIT\_UBAR registers.
  - FW\_LBAR and FW\_UBAR registers.
  - Intel Trace Hub Device Specific Control (NPKSDC) register.

Please note that the user of the software reset needs to re-program the Intel Trace Hub entirely, including its PCI command register, to re-enable it. This software reset is mainly for debug purposes and should not be visible to the Intel Trace Hub PCI driver since it changes the PCI header contents.

# 14 Register Description

This chapter provides details for all the registers within the Intel Trace Hub, including PCI Configuration space registers, memory-mapped I/O registers, and TAP data registers.

## 14.1 Register Attributes Definitions

The register attributes used in this chapter are listed in Table 14-1.

Most registers within the Intel Trace Hub are reset to their hardware default value by the main power-up reset, called npk\_rst\_b. The PCI configuration space registers are generally reset by the main system or CPU reset, called “host\_rst\_b”. The host\_rst\_b signal is typically only asserted when the “host” (main IA CPU) subsystem experiences a reset (e.g., warm reset).

**Table 14-1: Register Attributes**

Tag	Name	Description
RW	Read/Write	Bit is readable and writable
WO	Write Only	Bit is only writeable with reads returning zeros.
RO	Read Only	Bit is only readable, but writes have no effects. Read-only should not be used for reserved fields.
RV, Rsvd	Reserved	Bit is reserved. Reads will return 0, and writes have no effect. Reserved tag may be either RV or Rsvd. The “Rsvd” is used to reduce potential confusion with RW because a ‘V’ and ‘W’ may be mistaken when the font is small and .pdf compresses the image of the text.
RW/1C	Read/Clear	Reads will return current value. A write with logic 1 causes the bit to be cleared. Writes with logic 0 have no effect
W+R	Write with optional Read	The primary operation for the register is a write operation. Write data must be provided (it is not optional). The hardware also provides read data, which can be consumed or ignored. This register type applies to TAP registers, where write data must be provided.

## 14.2 Global Trace Hub Registers

### 14.2.1 Global Trace Hub Registers Summary

Global Trace Hub Registers			
Offset Start	Offset End	Symbol	Register Name/Function
0	3	GTHOPT0	GTH Output Ports 0-3
4	7	GTHOPT1	GTH Output Ports 4-7
8	B	SWDEST_0	Switching Destination [0]
C	F	SWDEST_1	Switching Destination [1]
10	13	SWDEST_2	Switching Destination [2]
14	17	SWDEST_3	Switching Destination [3]
18	1B	SWDEST_4	Switching Destination [4]
1C	1F	SWDEST_5	Switching Destination [5]
20	23	SWDEST_6	Switching Destination [6]
24	27	SWDEST_7	Switching Destination [7]
28	2B	SWDEST_8	Switching Destination [8]
2C	2F	SWDEST_9	Switching Destination [9]
30	33	SWDEST_10	Switching Destination [10]
34	37	SWDEST_11	Switching Destination [11]
38	3B	SWDEST_12	Switching Destination [12]
3C	3F	SWDEST_13	Switching Destination [13]
40	43	SWDEST_14	Switching Destination [14]
44	47	SWDEST_15	Switching Destination [15]
48	4B	SWDEST_16	Switching Destination [16]
4C	4F	SWDEST_17	Switching Destination [17]
50	53	SWDEST_18	Switching Destination [18]
54	57	SWDEST_19	Switching Destination [19]
58	5B	SWDEST_20	Switching Destination [20]
5C	5F	SWDEST_21	Switching Destination [21]
60	63	SWDEST_22	Switching Destination [22]
64	67	SWDEST_23	Switching Destination [23]
68	6B	SWDEST_24	Switching Destination [24]
6C	6F	SWDEST_25	Switching Destination [25]
70	73	SWDEST_26	Switching Destination [26]

Global Trace Hub Registers			
Offset Start	Offset End	Symbol	Register Name/Function
74	77	SWDEST_27	Switching Destination [27]
78	7B	SWDEST_28	Switching Destination [28]
7C	7F	SWDEST_29	Switching Destination [29]
80	83	SWDEST_30	Switching Destination [30]
84	87	SWDEST_31	Switching Destination [31]
88	8B	GSWDEST	General Software Trace Destination
8C	8F	LWMO	Low WaterMark for Sources 0-7
90	93	LWM1	Low WaterMark for Sources 7-15
94	97	GTH_INFO_1	GTH Parameter Info 1
98	9B	GTH_MISC	GTH Miscellaneous Register
9C	9F	SMCRO	STP Maintenance Control Register 0
A0	A3	SMCR1	STP Maintenance Control Register 1
A4	A7	SMCR2	STP Maintenance Control Register 2
A8	AB	SMCR3	STP Maintenance Control Register 3
AC	AF	PGDO	Programmable Grant Duration Register 0
B0	B3	PGD1	Programmable Grant Duration Register 1
C8	CB	SCR	Source Control Register
CC	CF	GTHFRQ	GTH Frequency Register
D0	D3	BPB_FRAME_SIZE	Byte Packing Buffer max frame size
D4	D7	GTHSTAT	GTH Status Register
D8	DB	SCR2	Source Control Register 2
DC	DF	DESTOVR	Destination Override Register
E0	E3	SCRPD0	ScratchPad[0] Register
E4	E7	SCRPD1	ScratchPad[1] Register
E8	EB	SCRPD2	ScratchPad[2] Register
EC	EF	SCRPD3	ScratchPad[3] Register
F0	F3	NDB_CTRL	NPK DTF Control Register
120	123	BPB_CSR0	BPB Control/Status Register 0
124	127	BPB_CSR1	BPB Control/Status Register 0
124	127	BPB_CSR1	BPB Control/Status Register 1
1C00	1C03	LPP_CTL	LPP Control Register



## 14.2.2 GTHOPT0: GTH Output Ports 0-3

CSR Register Name: GTHOPT0: GTH Output Ports 0-3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0	Offset End: 3
Bits	Access	Default	Label	Bit Description	
31	RW	0	P3FLUSH	<b>Port 3 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 3	
30	RW	0	RSVD	Reserved for future use	
29	RO/V	0	P3RST	<b>Port 3 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.	
28	RW	0	P3DRP	GTH Data Retention Policy for Port 3. See PODRP for details.	
27	RW	0	P3NULL	NULL Packet Generation for output port 3	
26:24	RO	0	P3TYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved	
23	RW	0	P2FLUSH	<b>Port 2 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 2	
22	RW	0	RSVD	Reserved for future use	
21	RO/V	0	P2RST	<b>Port 2 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.	
20	RW	0	P2DRP	GTH Data Retention Policy for Port 2. See PODRP for details.	
19	RW	0	P2NULL	NULL Packet Generation for output port 2.	
18:16	RO	2	P2TYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved	
15	RW	0	P1FLUSH	<b>Port 1 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 1.	
14	RW	0	RSVD	Reserved for future use	
13	RO/V	0	P1RST	<b>Port 1 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.	



CSR Register Name: GTHOPT0: GTH Output Ports 0-3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0	Offset End: 3
Bits	Access	Default	Label	Bit Description	
12	RW	0	P1DRP	GTH Data Retention Policy for Port 1. See P0DRP for details.	
11	RW	0	P1NULL	NULL Packet Generation for output port 1.	
10:8	RO	1	P1TYPE	<b>GTH Output Port 1 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved	
7	RW	0	POFLUSH	<b>Port 0 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 0.	
6	RW	0	RSVD	Reserved for future use	
5	RO/V	0	PORST	<b>Port 0 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.	
4	RW	0	PODRP	<b>GTH Data Retention Policy for Port 0.</b> This bit defines the behavior of the GTH when Port 0 is in a not ready or reset condition. Specifically, it is when portReset is asserted. The conditions under which the portReset signal is asserted for a given port is defined by the port's logic, and is outside the scope of this specification. This condition might occur when the output port is unconfigured, held in reset, or otherwise not fully operational. When the portReset is asserted, the BPB will take action based on the setting of the PnDRP bit. Specifically: 0: Hold/retain data. In this mode, the GTH will hold, or retain, all the trace data it has. The Byte Packing Buffer will very quickly fill up, and stall its data path. When the affected Input Buffers fill up, they will de-assert their get signal, indicating to their trace source(s) that they cannot accept any more input data. 1: Drop data. In this mode, the Byte Packing Buffer will ignore the deassertion of its get input, behaving as if it is asserted continuously. This will have the net effect of unloading data from the BPB and dropping it on the floor (it is permanently lost).	
3	RW	0	PONULL	<b>NULL Packet Generation for output port 0.</b> 0: NULL Packets are suppressed. 1: NULL Packets are generated.	
2:0	RO	1	POTYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved	



### 14.2.3 GTHOPT1: GTH Output Ports 4-7

CSR Register Name: GTHOPT1: GTH Output Ports 4-7				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 4
Bits	Access	Default	Label	Bit Description
31	RW	0	P7FLUSH	<b>Port 7 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 7.
30	RW	0	RSVD	Reserved for future use
29	RO/V	0	P7RST	<b>Port 7 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.
28	RW	0	P7DRP	<b>GTH Data Retention Policy for Port 7.</b> This bit defines the behavior of the GTH when Port 7 is in a not ready or reset condition. Specifically, it is when portReset is asserted. The conditions under which the portReset signal is asserted for a given port is defined by the port's logic, and is outside the scope of this specification. This condition might occur when the output port is unconfigured, held in reset, or otherwise not fully operational. When the portReset is asserted, the BPB will take action based on the setting of the PnDRP bit. Specifically: 0: Hold/retain data. In this mode, the GTH will hold, or retain, all the trace data it has. The Byte Packing Buffer will very quickly fill up, and stall its data path. When the affected Input Buffers fill up, they will de-assert their get signal, indicating to their trace source(s) that they cannot accept any more input data. 1: Drop data. In this mode, the Byte Packing Buffer will ignore the deassertion of its get input, behaving as if it is asserted continuously. This will have the net effect of unloading data from the BPB and dropping it on the floor (it is permanently lost).
27	RW	0	P7NULL	NULL Packet Generation for output port 7 0: NULL Packets are suppressed 1: NULL Packets are generated
26:24	RO	0	P7TYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved
23	RW	0	P6FLUSH	<b>Port 6 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 6.
22	RW	0	RSVD	Reserved for future use



CSR Register Name: GTHOPT1: GTH Output Ports 4-7				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 4
Bits	Access	Default	Label	Bit Description
21	RO/V	0	P6RST	<b>Port 6 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.
20	RW	0	P6DRP	<b>GTH Data Retention Policy for Port 6.</b> This bit defines the behavior of the GTH when Port 6 is in a not ready or reset condition. Specifically, it is when portReset is asserted. The conditions under which the portReset signal is asserted for a given port is defined by the port's logic, and is outside the scope of this specification. This condition might occur when the output port is unconfigured, held in reset, or otherwise not fully operational. When the portReset is asserted, the BPB will take action based on the setting of the PnDRP bit. Specifically: 0: Hold/retain data. In this mode, the GTH will hold, or retain, all the trace data it has. The Byte Packing Buffer will very quickly fill up, and stall its data path. When the affected Input Buffers fill up, they will de-assert their get signal, indicating to their trace source(s) that they cannot accept any more input data. 1: Drop data. In this mode, the Byte Packing Buffer will ignore the deassertion of its get input, behaving as if it is asserted continuously. This will have the net effect of unloading data from the BPB and dropping it on the floor (it is permanently lost).
19	RW	0	P6NULL	NULL Packet Generation for output port 6 0: NULL Packets are suppressed 1: NULL Packets are generated
18:16	RO	0	P6TYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved
15	RW	0	P5FLUSH	<b>Port 5 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 5.
14	RW	0	RSVD	Reserved for future use
13	RO/V	0	P5RST	<b>Port 5 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.

CSR Register Name: GTHOPT1: GTH Output Ports 4-7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 4	Offset End: 7
Bits	Access	Default	Label	Bit Description	
12	RW	0	P5DRP	<b>GTH Data Retention Policy for Port 5.</b> This bit defines the behavior of the GTH when Port 5 is in a not ready or reset condition. Specifically, it is when portReset is asserted. The conditions under which the portReset signal is asserted for a given port is defined by the port's logic, and is outside the scope of this specification. This condition might occur when the output port is unconfigured, held in reset, or otherwise not fully operational. When the portReset is asserted, the BPB will take action based on the setting of the PnDRP bit. Specifically: 0: Hold/retain data. In this mode, the GTH will hold, or retain, all the trace data it has. The Byte Packing Buffer will very quickly fill up, and stall its data path. When the affected Input Buffers fill up, they will de-assert their get signal, indicating to their trace source(s) that they cannot accept any more input data. 1: Drop data. In this mode, the Byte Packing Buffer will ignore the deassertion of its get input, behaving as if it is asserted continuously. This will have the net effect of unloading data from the BPB and dropping it on the floor (it is permanently lost).	
11	RW	0	P5NULL	NULL Packet Generation for output port 5 0: NULL Packets are suppressed 1: NULL Packets are generated	
10:8	RO	0	P5TYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved	
7	RW	0	P4FLUSH	<b>Port 4 Flush.</b> Setting this bit will assert the flush signal to the byte packing buffer for port 4.	
6	RW	0	RSVD	Reserved for future use	
5	RO/V	0	P4RST	<b>Port 4 in Reset.</b> When asserted, indicates the port is in reset, and is not accepting data. When cleared, indicates the port is operational, and can accept trace data.	



CSR Register Name: GTHOPT1: GTH Output Ports 4-7				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 4
Bits	Access	Default	Label	Bit Description
4	RW	0	P4DRP	<b>GTH Data Retention Policy for Port 4.</b> This bit defines the behavior of the GTH when Port 4 is in a not ready or reset condition. Specifically, it is when portReset is asserted. The conditions under which the portReset signal is asserted for a given port is defined by the port's logic, and is outside the scope of this specification. This condition might occur when the output port is unconfigured, held in reset, or otherwise not fully operational. When the portReset is asserted, the BPB will take action based on the setting of the PnDRP bit. Specifically: 0: Hold/retain data. In this mode, the GTH will hold, or retain, all the trace data it has. The Byte Packing Buffer will very quickly fill up, and stall its data path. When the affected Input Buffers fill up, they will de-assert their get signal, indicating to their trace source(s) that they cannot accept any more input data. 1: Drop data. In this mode, the Byte Packing Buffer will ignore the deassertion of its get input, behaving as if it is asserted continuously. This will have the net effect of unloading data from the BPB and dropping it on the floor (it is permanently lost).
3	RW	0	P4NULL	NULL Packet Generation for output port 4 0: NULL Packets are suppressed 1: NULL Packets are generated
2:0	RO	0	P4TYPE	<b>GTH Output Port 0 type.</b> 0h: No port 1h - System Memory / USB (DCI) 2h - MIPI-HTI 3h - Low Power Path (PTI, External Bridge) 4h - MIPI-PTI Others - reserved

#### 14.2.4 SWDEST\_0: Switching Destination [0]

CSR Register Name: SWDEST_0: Switching Destination [0]				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 8
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST7EN	<b>Master 7 Enable.</b> See MAST0EN for definition.
30:28	RW	0	MAST7DEST	<b>Master 7 Destination.</b> See MAST1DEST for definition.
27	RW	1	MAST6EN	<b>Master 6 Enable.</b> See MAST0EN for definition.
26:24	RW	0	MAST6DEST	<b>Master 6 Destination.</b> See MAST1DEST for definition.
23	RW	1	MAST5EN	<b>Master 5 Enable.</b> See MAST0EN for definition.

CSR Register Name: SWDEST_0: Switching Destination [0]				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 8
Bits	Access	Default	Label	Bit Description
22:20	RW	0	MAST5DEST	<b>Master 4 Enable.</b> See MASTOEN for definition.
19	RW	1	MAST4EN	<b>Master 4 Enable.</b> See MASTOEN for definition.
18:16	RW	0	MAST4DEST	<b>Master 4 Destination.</b> See MAST1DEST for definition.
15	RW	1	MAST3EN	<b>Master 3 Enable.</b> See MASTOEN for definition.
14:12	RW	0	MAST3DEST	<b>Master 3 Destination.</b> See MAST1DEST for definition.
11	RW	1	MAST2EN	<b>Master 2 Enable.</b> See MASTOEN for definition.
10:8	RW	0	MAST2DEST	<b>Master 2 Destination.</b> See MAST0DEST for definition.
7	RW	1	MAST1EN	<b>Master 1 Enable.</b> See MASTOEN for definition.
6:4	RW	0	MAST1DEST	<b>Master 1 Destination.</b> See MAST0DEST for definition.
3	RW	1	MASTOEN	<b>Master 0 Enable.</b> Enables tracing for Master 0. 0: tracing for Master 0 is disabled. All data received from Master 0 is received at the Input Buffer and immediately dropped. 1: tracing for Master 0 is enabled (Default)
2:0	RW	0	MAST0DEST	<b>Master 0 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 0 trace data. 0x0: Master 0 is routed to Output Port 0 0x1: Master 0 is routed to Output Port 1 ... 0x7: Master 0 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

#### 14.2.5 SWDEST\_1: Switching Destination [1]

CSR Register Name: SWDEST_1: Switching Destination [1]				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: C
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST15EN	<b>Master 15 Enable.</b> See MAST8EN for definition.
30:28	RW	0	MAST15DEST	<b>Master 15 Destination.</b> See MAST8DEST for definition.
27	RW	1	MAST14EN	<b>Master 14 Enable.</b> See MAST8EN for definition.
26:24	RW	0	MAST14DEST	<b>Master 14 Destination.</b> See MAST8DEST for definition.
23	RW	1	MAST13EN	<b>Master 13 Enable.</b> See MAST8EN for definition.

<b>CSR Register Name: SWDEST_1: Switching Destination [1]</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: C</b>	<b>Offset End: F</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
22:20	RW	0	MAST13DEST	<b>Master 13 Destination.</b> See MAST8DEST for definition.	
19	RW	1	MAST12EN	<b>Master 12 Enable.</b> See MAST8EN for definition.	
18:16	RW	0	MAST12DEST	<b>Master 12 Destination.</b> See MAST8DEST for definition.	
15	RW	1	MAST11EN	<b>Master 11 Enable.</b> See MAST8EN for definition.	
14:12	RW	0	MAST11DEST	<b>Master 11 Destination.</b> See MAST8DEST for definition.	
11	RW	1	MAST10EN	<b>Master 10 Enable.</b> See MAST8EN for definition.	
10:8	RW	0	MAST10DEST	<b>Master 10 Destination.</b> See MAST8DEST for definition.	
7	RW	1	MAST9EN	<b>Master 9 Enable.</b> See MAST8EN for definition.	
6:4	RW	0	MAST9DEST	<b>Master 9 Destination.</b> See MAST8DEST for definition.	
3	RW	1	MAST8EN	<b>Master 8 Enable.</b> Enables tracing for Master 8. 0: tracing for Master 8 is disabled. All data received from Master 8 is received at the Input Buffer and immediately dropped. 1: tracing for Master 8 is enabled (Default)	
2:0	RW	0	MAST8DEST	<b>Master 8 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 8 trace data. 0x0: Master 8 is routed to Output Port 0 0x1: Master 8 is routed to Output Port 1 ... 0x7: Master 8 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.	

#### 14.2.6 SWDEST\_2: Switching Destination [2]

<b>CSR Register Name: SWDEST_2: Switching Destination [2]</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 10</b>	<b>Offset End: 13</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31	RW	1	MAST23EN	<b>Master 23 Enable.</b> See MAST16EN for definition.	
30:28	RW	0	MAST23DEST	<b>Master 23 Destination.</b> See MAST16DEST for definition.	
27	RW	1	MAST22EN	<b>Master 22 Enable.</b> See MAST16EN for definition.	
26:24	RW	0	MAST22DEST	<b>Master 22 Destination.</b> See MAST16DEST for definition.	

<b>CSR Register Name: SWDEST_2: Switching Destination [2]</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 10</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
23	RW	1	MAST21EN	<b>Master 21 Enable.</b> See MAST16EN for definition.
22:20	RW	0	MAST21DEST	<b>Master 21 Destination.</b> See MAST16DEST for definition.
19	RW	1	MAST20EN	<b>Master 20 Enable.</b> See MAST16EN for definition.
18:16	RW	0	MAST20DEST	<b>Master 20 Destination.</b> See MAST16DEST for definition.
15	RW	1	MAST19EN	<b>Master 19 Enable.</b> See MAST16EN for definition.
14:12	RW	0	MAST19DEST	<b>Master 19 Destination.</b> See MAST16DEST for definition.
11	RW	1	MAST18EN	<b>Master 18 Enable.</b> See MAST16EN for definition.
10:8	RW	0	MAST18DEST	<b>Master 18 Destination.</b> See MAST16DEST for definition.
7	RW	1	MAST17EN	<b>Master 17 Enable.</b> See MAST16EN for definition.
6:4	RW	0	MAST17DEST	<b>Master 17 Destination.</b> See MAST16DEST for definition.
3	RW	1	MAST16EN	<b>Master 16 Enable.</b> Enables tracing for Master 16. 0: tracing for Master 16 is disabled. All data received from Master 16 is received at the Input Buffer and immediately dropped. 1: tracing for Master 16 is enabled (Default)
2:0	RW	0	MAST16DEST	<b>Master 16 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 16 trace data. 0x0: Master 16 is routed to Output Port 0 0x1: Master 16 is routed to Output Port 1 ... 0x7: Master 16 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

#### 14.2.7 SWDEST\_3: Switching Destination [3]

<b>CSR Register Name: SWDEST_3: Switching Destination [3]</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 14</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31	RW	1	MAST31EN	<b>Master 31 Enable.</b> See MAST24EN for definition.
30:28	RW	0	MAST31DEST	<b>Master 31 Destination.</b> See MAST24DEST for definition.
27	RW	1	MAST30EN	<b>Master 30 Enable.</b> See MAST24EN for definition.

<b>CSR Register Name: SWDEST_3: Switching Destination [3]</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 14</b>	<b>Offset End: 17</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
26:24	RW	0	MAST30DEST	<b>Master 30 Destination.</b> See MAST24DEST for definition.	
23	RW	1	MAST29EN	<b>Master 29 Enable.</b> See MAST24EN for definition.	
22:20	RW	0	MAST29DEST	<b>Master 29 Destination.</b> See MAST24DEST for definition.	
19	RW	1	MAST28EN	<b>Master 28 Enable.</b> See MAST24EN for definition.	
18:16	RW	0	MAST28DEST	<b>Master 28 Destination.</b> See MAST24DEST for definition.	
15	RW	1	MAST27EN	<b>Master 27 Enable.</b> See MAST24EN for definition.	
14:12	RW	0	MAST27DEST	<b>Master 27 Destination.</b> See MAST24DEST for definition.	
11	RW	1	MAST26EN	<b>Master 26 Enable.</b> See MAST24EN for definition.	
10:8	RW	0	MAST26DEST	<b>Master 26 Destination.</b> See MAST24DEST for definition.	
7	RW	1	MAST25EN	<b>Master 25 Enable.</b> See MAST24EN for definition.	
6:4	RW	0	MAST25DEST	<b>Master 25 Destination.</b> See MAST24DEST for definition.	
3	RW	1	MAST24EN	<b>Master 24 Enable.</b> Enables tracing for Master 24. 0: tracing for Master 24 is disabled. All data received from Master 24 is received at the Input Buffer and immediately dropped. 1: tracing for Master 24 is enabled (Default)	
2:0	RW	0	MAST24DEST	<b>Master 24 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 24 trace data. 0x0: Master 24 is routed to Output Port 0 0x1: Master 24 is routed to Output Port 1 ... 0x7: Master 24 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.	

#### 14.2.8 SWDEST\_4: Switching Destination [4]

<b>CSR Register Name: SWDEST_4: Switching Destination [4]</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 18</b>	<b>Offset End: 1B</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31	RW	1	MAST39EN	<b>Master 39 Enable.</b> See MAST32EN for definition.	
30:28	RW	0	MAST39DEST	<b>Master 39 Destination.</b> See MAST32DEST for definition.	

CSR Register Name: SWDEST_4: Switching Destination [4]				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 18
Bits	Access	Default	Label	Bit Description
27	RW	1	MAST38EN	<b>Master 38 Enable.</b> See MAST32EN for definition.
26:24	RW	0	MAST38DEST	<b>Master 38 Destination.</b> See MAST32DEST for definition.
23	RW	1	MAST37EN	<b>Master 37 Enable.</b> See MAST32EN for definition.
22:20	RW	0	MAST37DEST	<b>Master 37 Destination.</b> See MAST32DEST for definition.
19	RW	1	MAST36EN	<b>Master 36 Enable.</b> See MAST32EN for definition.
18:16	RW	0	MAST36DEST	<b>Master 36 Destination.</b> See MAST32DEST for definition.
15	RW	1	MAST35EN	<b>Master 35 Enable.</b> See MAST32EN for definition.
14:12	RW	0	MAST35DEST	<b>Master 35 Destination.</b> See MAST32DEST for definition.
11	RW	1	MAST34EN	<b>Master 34 Enable.</b> See MAST32EN for definition.
10:8	RW	0	MAST34DEST	<b>Master 34 Destination.</b> See MAST32DEST for definition.
7	RW	1	MAST33EN	<b>Master 33 Enable.</b> See MAST32EN for definition.
6:4	RW	0	MAST33DEST	<b>Master 33 Destination.</b> See MAST32DEST for definition.
3	RW	1	MAST32EN	<b>Master 32 Enable.</b> Enables tracing for Master 32. 0: tracing for Master 32 is disabled. All data received from Master 32 is received at the Input Buffer and immediately dropped. 1: tracing for Master 32 is enabled (Default)
2:0	RW	0	MAST32DEST	<b>Master 32 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 32 trace data. 0x0: Master 32 is routed to Output Port 0 0x1: Master 32 is routed to Output Port 1 ... 0x7: Master 32 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

#### 14.2.9 SWDEST\_5: Switching Destination [5]

CSR Register Name: SWDEST_5: Switching Destination [5]				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 1C
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST47EN	<b>Master 47 Enable.</b> See MAST40EN for definition.

<b>CSR Register Name: SWDEST_5: Switching Destination [5]</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 1C</b>	<b>Offset End: 1F</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
30:28	RW	0	MAST47DEST	<b>Master 47 Destination.</b> See MAST40DEST for definition.	
27	RW	1	MAST46EN	<b>Master 46 Enable.</b> See MAST40EN for definition.	
26:24	RW	0	MAST46DEST	<b>Master 46 Destination.</b> See MAST40DEST for definition.	
23	RW	1	MAST45EN	<b>Master 45 Enable.</b> See MAST40EN for definition.	
22:20	RW	0	MAST45DEST	<b>Master 45 Destination.</b> See MAST40DEST for definition.	
19	RW	1	MAST44EN	<b>Master 44 Enable.</b> See MAST40EN for definition.	
18:16	RW	0	MAST44DEST	<b>Master 44 Destination.</b> See MAST40DEST for definition.	
15	RW	1	MAST43EN	<b>Master 43 Enable.</b> See MAST40EN for definition.	
14:12	RW	0	MAST43DEST	<b>Master 43 Destination.</b> See MAST40DEST for definition.	
11	RW	1	MAST42EN	<b>Master 42 Enable.</b> See MAST40EN for definition.	
10:8	RW	0	MAST42DEST	<b>Master 42 Destination.</b> See MAST40DEST for definition.	
7	RW	1	MAST41EN	<b>Master 41 Enable.</b> See MAST40EN for definition.	
6:4	RW	0	MAST41DEST	<b>Master 41 Destination.</b> See MAST40DEST for definition.	
3	RW	1	MAST40EN	<b>Master 40 Enable.</b> Enables tracing for Master 40. 0: tracing for Master 40 is disabled. All data received from Master 40 is received at the Input Buffer and immediately dropped. 1: tracing for Master 40 is enabled (Default)	
2:0	RW	0	MAST40DEST	<b>Master 40 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 40 trace data. 0x0: Master 40 is routed to Output Port 0 0x1: Master 40 is routed to Output Port 1 ... 0x7: Master 40 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.	

### 14.2.10 SWDEST\_6: Switching Destination [6]

CSR Register Name: SWDEST_6: Switching Destination [6]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 20	Offset End: 23	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST55EN	<b>Master 55 Enable.</b> See MAST48EN for definition.
30:28	RW	0	MAST55DEST	<b>Master 55 Destination.</b> See MAST48DEST for definition.
27	RW	1	MAST54EN	<b>Master 54 Enable.</b> See MAST48EN for definition.
26:24	RW	0	MAST54DEST	<b>Master 54 Destination.</b> See MAST48DEST for definition.
23	RW	1	MAST53EN	<b>Master 53 Enable.</b> See MAST48EN for definition.
22:20	RW	0	MAST53DEST	<b>Master 53 Destination.</b> See MAST48DEST for definition.
19	RW	1	MAST52EN	<b>Master 52 Enable.</b> See MAST48EN for definition.
18:16	RW	0	MAST52DEST	<b>Master 52 Destination.</b> See MAST48DEST for definition.
15	RW	1	MAST51EN	<b>Master 51 Enable.</b> See MAST48EN for definition.
14:12	RW	0	MAST51DEST	<b>Master 51 Destination.</b> See MAST48DEST for definition.
11	RW	1	MAST50EN	<b>Master 50 Enable.</b> See MAST48EN for definition.
10:8	RW	0	MAST50DEST	<b>Master 50 Destination.</b> See MAST48DEST for definition.
7	RW	1	MAST49EN	<b>Master 49 Enable.</b> See MAST48EN for definition.
6:4	RW	0	MAST49DEST	<b>Master 49 Destination.</b> See MAST48DEST for definition.
3	RW	1	MAST48EN	<b>Master 48 Enable.</b> Enables tracing for Master 48. 0: tracing for Master 48 is disabled. All data received from Master 48 is received at the Input Buffer and immediately dropped. 1: tracing for Master 48 is enabled (Default)
2:0	RW	0	MAST48DEST	<b>Master 48 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 48 trace data. 0x0: Master 48 is routed to Output Port 0 0x1: Master 48 is routed to Output Port 1 ... 0x7: Master 48 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.11 SWDEST\_7: Switching Destination [7]

CSR Register Name: SWDEST_7: Switching Destination [7]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 24	Offset End: 27	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST63EN	<b>Master 63 Enable.</b> See MAST56EN for definition.
30:28	RW	0	MAST63DEST	<b>Master 63 Destination.</b> See MAST56DEST for definition.
27	RW	1	MAST62EN	<b>Master 62 Enable.</b> See MAST56EN for definition.
26:24	RW	0	MAST62DEST	<b>Master 62 Destination.</b> See MAST56DEST for definition.
23	RW	1	MAST61EN	<b>Master 61 Enable.</b> See MAST56EN for definition.
22:20	RW	0	MAST61DEST	<b>Master 61 Destination.</b> See MAST56DEST for definition.
19	RW	1	MAST60EN	<b>Master 60 Enable.</b> See MAST56EN for definition.
18:16	RW	0	MAST60DEST	<b>Master 60 Destination.</b> See MAST56DEST for definition.
15	RW	1	MAST59EN	<b>Master 59 Enable.</b> See MAST56EN for definition.
14:12	RW	0	MAST59DEST	<b>Master 59 Destination.</b> See MAST56DEST for definition.
11	RW	1	MAST58EN	<b>Master 58 Enable.</b> See MAST56EN for definition.
10:8	RW	0	MAST58DEST	<b>Master 58 Destination.</b> See MAST56DEST for definition.
7	RW	1	MAST57EN	<b>Master 57 Enable.</b> See MAST56EN for definition.
6:4	RW	0	MAST57DEST	<b>Master 57 Destination.</b> See MAST56DEST for definition.
3	RW	1	MAST56EN	<b>Master 56 Enable.</b> Enables tracing for Master 56. 0: tracing for Master 56 is disabled. All data received from Master 56 is received at the Input Buffer and immediately dropped. 1: tracing for Master 56 is enabled (Default)
2:0	RW	0	MAST56DEST	<b>Master 56 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 56 trace data. 0x0: Master 56 is routed to Output Port 0 0x1: Master 56 is routed to Output Port 1 ... 0x7: Master 56 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.12 SWDEST\_8: Switching Destination [8]

CSR Register Name: SWDEST_8: Switching Destination [8]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 28	Offset End: 2B	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST71EN	<b>Master 71 Enable.</b> See MAST64EN for definition.
30:28	RW	0	MAST71DEST	<b>Master 71 Destination.</b> See MAST64DEST for definition.
27	RW	1	MAST70EN	<b>Master 70 Enable.</b> See MAST64EN for definition.
26:24	RW	0	MAST70DEST	<b>Master 70 Destination.</b> See MAST64DEST for definition.
23	RW	1	MAST69EN	<b>Master 69 Enable.</b> See MAST64EN for definition.
22:20	RW	0	MAST69DEST	<b>Master 69 Destination.</b> See MAST64DEST for definition.
19	RW	1	MAST68EN	<b>Master 68 Enable.</b> See MAST64EN for definition.
18:16	RW	0	MAST68DEST	<b>Master 68 Destination.</b> See MAST64DEST for definition.
15	RW	1	MAST67EN	<b>Master 67 Enable.</b> See MAST64EN for definition.
14:12	RW	0	MAST67DEST	<b>Master 67 Destination.</b> See MAST64DEST for definition.
11	RW	1	MAST66EN	<b>Master 66 Enable.</b> See MAST64EN for definition.
10:8	RW	0	MAST66DEST	<b>Master 66 Destination.</b> See MAST64DEST for definition.
7	RW	1	MAST65EN	<b>Master 65 Enable.</b> See MAST64EN for definition.
6:4	RW	0	MAST65DEST	<b>Master 65 Destination.</b> See MAST64DEST for definition.
3	RW	1	MAST64EN	<b>Master 64 Enable.</b> Enables tracing for Master 64. 0: tracing for Master 64 is disabled. All data received from Master 64 is received at the Input Buffer and immediately dropped. 1: tracing for Master 64 is enabled (Default)
2:0	RW	0	MAST64DEST	<b>Master 64 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 64 trace data. 0x0: Master 64 is routed to Output Port 0 0x1: Master 64 is routed to Output Port 1 ... 0x7: Master 64 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.13 SWDEST\_9: Switching Destination [9]

CSR Register Name: SWDEST_9: Switching Destination [9]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 2C	Offset End: 2F	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST79EN	<b>Master 79 Enable.</b> See MAST72EN for definition.
30:28	RW	0	MAST79DEST	<b>Master 79 Destination.</b> See MAST72DEST for definition.
27	RW	1	MAST78EN	<b>Master 78 Enable.</b> See MAST72EN for definition.
26:24	RW	0	MAST78DEST	<b>Master 78 Destination.</b> See MAST72DEST for definition.
23	RW	1	MAST77EN	<b>Master 77 Enable.</b> See MAST72EN for definition.
22:20	RW	0	MAST77DEST	<b>Master 77 Destination.</b> See MAST72DEST for definition.
19	RW	1	MAST76EN	<b>Master 76 Enable.</b> See MAST72EN for definition.
18:16	RW	0	MAST76DEST	<b>Master 76 Destination.</b> See MAST72DEST for definition.
15	RW	1	MAST75EN	<b>Master 75 Enable.</b> See MAST72EN for definition.
14:12	RW	0	MAST75DEST	<b>Master 75 Destination.</b> See MAST72DEST for definition.
11	RW	1	MAST74EN	<b>Master 74 Enable.</b> See MAST72EN for definition.
10:8	RW	0	MAST74DEST	<b>Master 74 Destination.</b> See MAST72DEST for definition.
7	RW	1	MAST73EN	<b>Master 73 Enable.</b> See MAST72EN for definition.
6:4	RW	0	MAST73DEST	<b>Master 73 Destination.</b> See MAST72DEST for definition.
3	RW	1	MAST72EN	<b>Master 72 Enable.</b> Enables tracing for Master 72. 0: tracing for Master 72 is disabled. All data received from Master 72 is received at the Input Buffer and immediately dropped. 1: tracing for Master 72 is enabled (Default)
2:0	RW	0	MAST72DEST	<b>Master 72 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 72 trace data. 0x0: Master 72 is routed to Output Port 0 0x1: Master 72 is routed to Output Port 1 ... 0x7: Master 72 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

#### 14.2.14 SWDEST\_10: Switching Destination [10]

CSR Register Name: SWDEST_10: Switching Destination [10]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 30	Offset End: 33	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST87EN	<b>Master 87 Enable.</b> See MAST80EN for definition.
30:28	RW	0	MAST87DEST	<b>Master 87 Destination.</b> See MAST80DEST for definition.
27	RW	1	MAST86EN	<b>Master 86 Enable.</b> See MAST80EN for definition.
26:24	RW	0	MAST86DEST	<b>Master 86 Destination.</b> See MAST80DEST for definition.
23	RW	1	MAST85EN	<b>Master 85 Enable.</b> See MAST80EN for definition.
22:20	RW	0	MAST85DEST	<b>Master 85 Destination.</b> See MAST80DEST for definition.
19	RW	1	MAST84EN	<b>Master 84 Enable.</b> See MAST80EN for definition.
18:16	RW	0	MAST84DEST	<b>Master 84 Destination.</b> See MAST80DEST for definition.
15	RW	1	MAST83EN	<b>Master 83 Enable.</b> See MAST80EN for definition.
14:12	RW	0	MAST83DEST	<b>Master 83 Destination.</b> See MAST80DEST for definition.
11	RW	1	MAST82EN	<b>Master 82 Enable.</b> See MAST80EN for definition.
10:8	RW	0	MAST82DEST	<b>Master 82 Destination.</b> See MAST80DEST for definition.
7	RW	1	MAST81EN	<b>Master 81 Enable.</b> See MAST80EN for definition.
6:4	RW	0	MAST81DEST	<b>Master 81 Destination.</b> See MAST80DEST for definition.
3	RW	1	MAST80EN	<b>Master 80 Enable.</b> Enables tracing for Master 80. 0: tracing for Master 80 is disabled. All data received from Master 80 is received at the Input Buffer and immediately dropped. 1: tracing for Master 80 is enabled (Default)
2:0	RW	0	MAST80DEST	<b>Master 80 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 80 trace data. 0x0: Master 80 is routed to Output Port 0 0x1: Master 80 is routed to Output Port 1 ... 0x7: Master 80 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.15 SWDEST\_11: Switching Destination [11]

CSR Register Name: SWDEST_11: Switching Destination [11]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 34	Offset End: 37	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST95EN	<b>Master 95 Enable.</b> See MAST88EN for definition.
30:28	RW	0	MAST95DEST	<b>Master 95 Destination.</b> See MAST88DEST for definition.
27	RW	1	MAST94EN	<b>Master 94 Enable.</b> See MAST88EN for definition.
26:24	RW	0	MAST94DEST	<b>Master 94 Destination.</b> See MAST88DEST for definition.
23	RW	1	MAST93EN	<b>Master 93 Enable.</b> See MAST88EN for definition.
22:20	RW	0	MAST93DEST	<b>Master 93 Destination.</b> See MAST88DEST for definition.
19	RW	1	MAST92EN	<b>Master 92 Enable.</b> See MAST88EN for definition.
18:16	RW	0	MAST92DEST	<b>Master 92 Destination.</b> See MAST88DEST for definition.
15	RW	1	MAST91EN	<b>Master 91 Enable.</b> See MAST88EN for definition.
14:12	RW	0	MAST91DEST	<b>Master 91 Destination.</b> See MAST88DEST for definition.
11	RW	1	MAST90EN	<b>Master 90 Enable.</b> See MAST88EN for definition.
10:8	RW	0	MAST90DEST	<b>Master 90 Destination.</b> See MAST88DEST for definition.
7	RW	1	MAST89EN	<b>Master 89 Enable.</b> See MAST88EN for definition.
6:4	RW	0	MAST89DEST	<b>Master 89 Destination.</b> See MAST88DEST for definition.
3	RW	1	MAST88EN	<b>Master 88 Enable.</b> Enables tracing for Master 88. 0: tracing for Master 88 is disabled. All data received from Master 88 is received at the Input Buffer and immediately dropped. 1: tracing for Master 88 is enabled (Default)
2:0	RW	0	MAST88DEST	<b>Master 88 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 88 trace data. 0x0: Master 88 is routed to Output Port 0 0x1: Master 88 is routed to Output Port 1 ... 0x7: Master 88 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.16 SWDEST\_12: Switching Destination [12]

CSR Register Name: SWDEST_12: Switching Destination [12]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 38	Offset End: 3B	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST103EN	<b>Master 103 Enable.</b> See MAST96EN for definition.
30:28	RW	0	MAST103DEST	<b>Master 103 Destination.</b> See MAST96DEST for definition.
27	RW	1	MAST102EN	<b>Master 102 Enable.</b> See MAST96EN for definition.
26:24	RW	0	MAST102DEST	<b>Master 102 Destination.</b> See MAST96DEST for definition.
23	RW	1	MAST101EN	<b>Master 101 Enable.</b> See MAST96EN for definition.
22:20	RW	0	MAST101DEST	<b>Master 101 Destination.</b> See MAST96DEST for definition.
19	RW	1	MAST100EN	<b>Master 100 Enable.</b> See MAST96EN for definition.
18:16	RW	0	MAST100DEST	<b>Master 100 Destination.</b> See MAST96DEST for definition.
15	RW	1	MAST99EN	<b>Master 99 Enable.</b> See MAST96EN for definition.
14:12	RW	0	MAST99DEST	<b>Master 99 Destination.</b> See MAST96DEST for definition.
11	RW	1	MAST98EN	<b>Master 98 Enable.</b> See MAST96EN for definition.
10:8	RW	0	MAST98DEST	<b>Master 98 Destination.</b> See MAST96DEST for definition.
7	RW	1	MAST97EN	<b>Master 97 Enable.</b> See MAST96EN for definition.
6:4	RW	0	MAST97DEST	<b>Master 97 Destination.</b> See MAST96DEST for definition.
3	RW	1	MAST96EN	<b>Master 96 Enable.</b> Enables tracing for Master 96. 0: tracing for Master 96 is disabled. All data received from Master 96 is received at the Input Buffer and immediately dropped. 1: tracing for Master 96 is enabled (Default)
2:0	RW	0	MAST96DEST	<b>Master 96 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 96 trace data. 0x0: Master 96 is routed to Output Port 0 0x1: Master 96 is routed to Output Port 1 ... 0x7: Master 96 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.17 SWDEST\_13: Switching Destination [13]

CSR Register Name: SWDEST_13: Switching Destination [13]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 3C	Offset End: 3F	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST111EN	<b>Master 111 Enable.</b> See MAST104EN for definition.
30:28	RW	0	MAST111DEST	<b>Master 111 Destination.</b> See MAST104DEST for definition.
27	RW	1	MAST110EN	<b>Master 110 Enable.</b> See MAST104EN for definition.
26:24	RW	0	MAST110DEST	<b>Master 110 Destination.</b> See MAST104DEST for definition.
23	RW	1	MAST109EN	<b>Master 109 Enable.</b> See MAST104EN for definition.
22:20	RW	0	MAST109DEST	<b>Master 109 Destination.</b> See MAST104DEST for definition.
19	RW	1	MAST108EN	<b>Master 108 Enable.</b> See MAST104EN for definition.
18:16	RW	0	MAST108DEST	<b>Master 108 Destination.</b> See MAST104DEST for definition.
15	RW	1	MAST107EN	<b>Master 107 Enable.</b> See MAST104EN for definition.
14:12	RW	0	MAST107DEST	<b>Master 107 Destination.</b> See MAST104DEST for definition.
11	RW	1	MAST106EN	<b>Master 106 Enable.</b> See MAST104EN for definition.
10:8	RW	0	MAST106DEST	<b>Master 106 Destination.</b> See MAST104DEST for definition.
7	RW	1	MAST105EN	<b>Master 105 Enable.</b> See MAST104EN for definition.
6:4	RW	0	MAST105DEST	<b>Master 105 Destination.</b> See MAST104DEST for definition.
3	RW	1	MAST104EN	<b>Master 104 Enable.</b> Enables tracing for Master 104. 0: tracing for Master 104 is disabled. All data received from Master 104 is received at the Input Buffer and immediately dropped. 1: tracing for Master 104 is enabled (Default)
2:0	RW	0	MAST104DEST	<b>Master 104 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 104 trace data. 0x0: Master 104 is routed to Output Port 0 0x1: Master 104 is routed to Output Port 1 ... 0x7: Master 104 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.18 SWDEST\_14: Switching Destination [14]

CSR Register Name: SWDEST_14: Switching Destination [14]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 40	Offset End: 43	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST119EN	<b>Master 119 Enable.</b> See MAST112EN for definition.
30:28	RW	0	MAST119DEST	<b>Master 119 Destination.</b> See MAST112DEST for definition.
27	RW	1	MAST118EN	<b>Master 118 Enable.</b> See MAST112EN for definition.
26:24	RW	0	MAST118DEST	<b>Master 118 Destination.</b> See MAST112DEST for definition.
23	RW	1	MAST117EN	<b>Master 117 Enable.</b> See MAST112EN for definition.
22:20	RW	0	MAST117DEST	<b>Master 117 Destination.</b> See MAST112DEST for definition.
19	RW	1	MAST116EN	<b>Master 116 Enable.</b> See MAST112EN for definition.
18:16	RW	0	MAST116DEST	<b>Master 116 Destination.</b> See MAST112DEST for definition.
15	RW	1	MAST115EN	<b>Master 115 Enable.</b> See MAST112EN for definition.
14:12	RW	0	MAST115DEST	<b>Master 115 Destination.</b> See MAST112DEST for definition.
11	RW	1	MAST114EN	<b>Master 114 Enable.</b> See MAST112EN for definition.
10:8	RW	0	MAST114DEST	<b>Master 114 Destination.</b> See MAST112DEST for definition.
7	RW	1	MAST113EN	<b>Master 113 Enable.</b> See MAST112EN for definition.
6:4	RW	0	MAST113DEST	<b>Master 113 Destination.</b> See MAST112DEST for definition.
3	RW	1	MAST112EN	<b>Master 112 Enable.</b> Enables tracing for Master 112. 0: tracing for Master 112 is disabled. All data received from Master 112 is received at the Input Buffer and immediately dropped. 1: tracing for Master 112 is enabled (Default)
2:0	RW	0	MAST112DEST	<b>Master 112 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 112 trace data. 0x0: Master 112 is routed to Output Port 0 0x1: Master 112 is routed to Output Port 1 ... 0x7: Master 112 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.19 SWDEST\_15: Switching Destination [15]

CSR Register Name: SWDEST_15: Switching Destination [15]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 44	Offset End: 47	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST127EN	<b>Master 127 Enable.</b> See MAST120EN for definition.
30:28	RW	0	MAST127DEST	<b>Master 127 Destination.</b> See MAST120DEST for definition.
27	RW	1	MAST126EN	<b>Master 126 Enable.</b> See MAST120EN for definition.
26:24	RW	0	MAST126DEST	<b>Master 126 Destination.</b> See MAST120DEST for definition.
23	RW	1	MAST125EN	<b>Master 125 Enable.</b> See MAST120EN for definition.
22:20	RW	0	MAST125DEST	<b>Master 125 Destination.</b> See MAST120DEST for definition.
19	RW	1	MAST124EN	<b>Master 124 Enable.</b> See MAST120EN for definition.
18:16	RW	0	MAST124DEST	<b>Master 124 Destination.</b> See MAST120DEST for definition.
15	RW	1	MAST123EN	<b>Master 123 Enable.</b> See MAST120EN for definition.
14:12	RW	0	MAST123DEST	<b>Master 123 Destination.</b> See MAST120DEST for definition.
11	RW	1	MAST122EN	<b>Master 122 Enable.</b> See MAST120EN for definition.
10:8	RW	0	MAST122DEST	<b>Master 122 Destination.</b> See MAST120DEST for definition.
7	RW	1	MAST121EN	<b>Master 121 Enable.</b> See MAST120EN for definition.
6:4	RW	0	MAST121DEST	<b>Master 121 Destination.</b> See MAST120DEST for definition.
3	RW	1	MAST120EN	<b>Master 120 Enable.</b> Enables tracing for Master 120. 0: tracing for Master 120 is disabled. All data received from Master 120 is received at the Input Buffer and immediately dropped. 1: tracing for Master 120 is enabled (Default)
2:0	RW	0	MAST120DEST	<b>Master 120 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 120 trace data. 0x0: Master 120 is routed to Output Port 0 0x1: Master 120 is routed to Output Port 1 ... 0x7: Master 120 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



#### 14.2.20 SWDEST\_16: Switching Destination [16]

CSR Register Name: SWDEST_16: Switching Destination [16]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 48	Offset End: 4B	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST135EN	<b>Master 135 Enable.</b> See MAST128EN for definition.
30:28	RW	0	MAST135DEST	<b>Master 135 Destination.</b> See MAST128DEST for definition.
27	RW	1	MAST134EN	<b>Master 134 Enable.</b> See MAST128EN for definition.
26:24	RW	0	MAST134DEST	<b>Master 134 Destination.</b> See MAST128DEST for definition.
23	RW	1	MAST133EN	<b>Master 133 Enable.</b> See MAST128EN for definition.
22:20	RW	0	MAST133DEST	<b>Master 133 Destination.</b> See MAST128DEST for definition.
19	RW	1	MAST132EN	<b>Master 132 Enable.</b> See MAST128EN for definition.
18:16	RW	0	MAST132DEST	<b>Master 132 Destination.</b> See MAST128DEST for definition.
15	RW	1	MAST131EN	<b>Master 131 Enable.</b> See MAST128EN for definition.
14:12	RW	0	MAST131DEST	<b>Master 131 Destination.</b> See MAST128DEST for definition.
11	RW	1	MAST130EN	<b>Master 130 Enable.</b> See MAST128EN for definition.
10:8	RW	0	MAST130DEST	<b>Master 130 Destination.</b> See MAST128DEST for definition.
7	RW	1	MAST129EN	<b>Master 129 Enable.</b> See MAST128EN for definition.
6:4	RW	0	MAST129DEST	<b>Master 129 Destination.</b> See MAST128DEST for definition.
3	RW	1	MAST128EN	<b>Master 128 Enable.</b> Enables tracing for Master 128. 0: tracing for Master 128 is disabled. All data received from Master 128 is received at the Input Buffer and immediately dropped. 1: tracing for Master 128 is enabled (Default)
2:0	RW	0	MAST128DEST	<b>Master 128 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 128 trace data. 0x0: Master 128 is routed to Output Port 0 0x1: Master 128 is routed to Output Port 1 ... 0x7: Master 128 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.21 SWDEST\_17: Switching Destination [17]

CSR Register Name: SWDEST_17: Switching Destination [17]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 4C	Offset End: 4F	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST143EN	<b>Master 143 Enable.</b> See MAST136EN for definition.
30:28	RW	0	MAST143DEST	<b>Master 143 Destination.</b> See MAST136DEST for definition.
27	RW	1	MAST142EN	<b>Master 142 Enable.</b> See MAST136EN for definition.
26:24	RW	0	MAST142DEST	<b>Master 142 Destination.</b> See MAST136DEST for definition.
23	RW	1	MAST141EN	<b>Master 141 Enable.</b> See MAST136EN for definition.
22:20	RW	0	MAST141DEST	<b>Master 141 Destination.</b> See MAST136DEST for definition.
19	RW	1	MAST140EN	<b>Master 140 Enable.</b> See MAST136EN for definition.
18:16	RW	0	MAST140DEST	<b>Master 140 Destination.</b> See MAST136DEST for definition.
15	RW	1	MAST139EN	<b>Master 139 Enable.</b> See MAST136EN for definition.
14:12	RW	0	MAST139DEST	<b>Master 139 Destination.</b> See MAST136DEST for definition.
11	RW	1	MAST138EN	<b>Master 138 Enable.</b> See MAST136EN for definition.
10:8	RW	0	MAST138DEST	<b>Master 138 Destination.</b> See MAST136DEST for definition.
7	RW	1	MAST137EN	<b>Master 137 Enable.</b> See MAST136EN for definition.
6:4	RW	0	MAST137DEST	<b>Master 137 Destination.</b> See MAST136DEST for definition.
3	RW	1	MAST136EN	<b>Master 136 Enable.</b> Enables tracing for Master 136. 0: tracing for Master 136 is disabled. All data received from Master 136 is received at the Input Buffer and immediately dropped. 1: tracing for Master 136 is enabled (Default)
2:0	RW	0	MAST136DEST	<b>Master 136 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 136 trace data. 0x0: Master 136 is routed to Output Port 0 0x1: Master 136 is routed to Output Port 1 ... 0x7: Master 136 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.22 SWDEST\_18: Switching Destination [18]

CSR Register Name: SWDEST_18: Switching Destination [18]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 50	Offset End: 53	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST151EN	<b>Master 151 Enable.</b> See MAST144EN for definition.
30:28	RW	0	MAST151DEST	<b>Master 151 Destination.</b> See MAST144DEST for definition.
27	RW	1	MAST150EN	<b>Master 150 Enable.</b> See MAST144EN for definition.
26:24	RW	0	MAST150DEST	<b>Master 150 Destination.</b> See MAST144DEST for definition.
23	RW	1	MAST149EN	<b>Master 149 Enable.</b> See MAST144EN for definition.
22:20	RW	0	MAST149DEST	<b>Master 149 Destination.</b> See MAST144DEST for definition.
19	RW	1	MAST148EN	<b>Master 148 Enable.</b> See MAST144EN for definition.
18:16	RW	0	MAST148DEST	<b>Master 148 Destination.</b> See MAST144DEST for definition.
15	RW	1	MAST147EN	<b>Master 147 Enable.</b> See MAST144EN for definition.
14:12	RW	0	MAST147DEST	<b>Master 147 Destination.</b> See MAST144DEST for definition.
11	RW	1	MAST146EN	<b>Master 146 Enable.</b> See MAST144EN for definition.
10:8	RW	0	MAST146DEST	<b>Master 146 Destination.</b> See MAST144DEST for definition.
7	RW	1	MAST145EN	<b>Master 145 Enable.</b> See MAST144EN for definition.
6:4	RW	0	MAST145DEST	<b>Master 145 Destination.</b> See MAST144DEST for definition.
3	RW	1	MAST144EN	<b>Master 144 Enable.</b> Enables tracing for Master 144. 0: tracing for Master 144 is disabled. All data received from Master 144 is received at the Input Buffer and immediately dropped. 1: tracing for Master 144 is enabled (Default)
2:0	RW	0	MAST144DEST	<b>Master 144 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 144 trace data. 0x0: Master 144 is routed to Output Port 0 0x1: Master 144 is routed to Output Port 1 ... 0x7: Master 144 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.23 SWDEST\_19: Switching Destination [19]

CSR Register Name: SWDEST_19: Switching Destination [19]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 54	Offset End: 57	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST159EN	<b>Master 159 Enable.</b> See MAST152EN for definition.
30:28	RW	0	MAST159DEST	<b>Master 159 Destination.</b> See MAST152DEST for definition.
27	RW	1	MAST158EN	<b>Master 158 Enable.</b> See MAST152EN for definition.
26:24	RW	0	MAST158DEST	<b>Master 158 Destination.</b> See MAST152DEST for definition.
23	RW	1	MAST157EN	<b>Master 157 Enable.</b> See MAST152EN for definition.
22:20	RW	0	MAST157DEST	<b>Master 157 Destination.</b> See MAST152DEST for definition.
19	RW	1	MAST156EN	<b>Master 156 Enable.</b> See MAST152EN for definition.
18:16	RW	0	MAST156DEST	<b>Master 156 Destination.</b> See MAST152DEST for definition.
15	RW	1	MAST155EN	<b>Master 155 Enable.</b> See MAST152EN for definition.
14:12	RW	0	MAST155DEST	<b>Master 155 Destination.</b> See MAST152DEST for definition.
11	RW	1	MAST154EN	<b>Master 154 Enable.</b> See MAST152EN for definition.
10:8	RW	0	MAST154DEST	<b>Master 154 Destination.</b> See MAST152DEST for definition.
7	RW	1	MAST153EN	<b>Master 153 Enable.</b> See MAST152EN for definition.
6:4	RW	0	MAST153DEST	<b>Master 153 Destination.</b> See MAST152DEST for definition.
3	RW	1	MAST152EN	<b>Master 152 Enable.</b> Enables tracing for Master 152. 0: tracing for Master 152 is disabled. All data received from Master 152 is received at the Input Buffer and immediately dropped. 1: tracing for Master 152 is enabled (Default)
2:0	RW	0	MAST152DEST	<b>Master 152 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 152 trace data. 0x0: Master 152 is routed to Output Port 0 0x1: Master 152 is routed to Output Port 1 ... 0x7: Master 152 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.24 SWDEST\_20: Switching Destination [20]

CSR Register Name: SWDEST_20: Switching Destination [20]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 58	Offset End: 5B	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST167EN	<b>Master 167 Enable.</b> See MAST160EN for definition.
30:28	RW	0	MAST167DEST	<b>Master 167 Destination.</b> See MAST160DEST for definition.
27	RW	1	MAST166EN	<b>Master 166 Enable.</b> See MAST160EN for definition.
26:24	RW	0	MAST166DEST	<b>Master 166 Destination.</b> See MAST160DEST for definition.
23	RW	1	MAST165EN	<b>Master 165 Enable.</b> See MAST160EN for definition.
22:20	RW	0	MAST165DEST	<b>Master 165 Destination.</b> See MAST160DEST for definition.
19	RW	1	MAST164EN	<b>Master 164 Enable.</b> See MAST160EN for definition.
18:16	RW	0	MAST164DEST	<b>Master 164 Destination.</b> See MAST160DEST for definition.
15	RW	1	MAST163EN	<b>Master 163 Enable.</b> See MAST160EN for definition.
14:12	RW	0	MAST163DEST	<b>Master 163 Destination.</b> See MAST160DEST for definition.
11	RW	1	MAST162EN	<b>Master 162 Enable.</b> See MAST160EN for definition.
10:8	RW	0	MAST162DEST	<b>Master 162 Destination.</b> See MAST160DEST for definition.
7	RW	1	MAST161EN	<b>Master 161 Enable.</b> See MAST160EN for definition.
6:4	RW	0	MAST161DEST	<b>Master 161 Destination.</b> See MAST160DEST for definition.
3	RW	1	MAST160EN	<b>Master 160 Enable.</b> Enables tracing for Master 160. 0: tracing for Master 160 is disabled. All data received from Master 160 is received at the Input Buffer and immediately dropped. 1: tracing for Master 160 is enabled (Default)
2:0	RW	0	MAST160DEST	<b>Master 160 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 160 trace data. 0x0: Master 160 is routed to Output Port 0 0x1: Master 160 is routed to Output Port 1 ... 0x7: Master 160 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.25 SWDEST\_21: Switching Destination [21]

CSR Register Name: SWDEST_21: Switching Destination [21]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5C	Offset End: 5F	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST175EN	<b>Master 175 Enable.</b> See MAST168EN for definition.
30:28	RW	0	MAST175DEST	<b>Master 175 Destination.</b> See MAST168DEST for definition.
27	RW	1	MAST174EN	<b>Master 174 Enable.</b> See MAST168EN for definition.
26:24	RW	0	MAST174DEST	<b>Master 174 Destination.</b> See MAST168DEST for definition.
23	RW	1	MAST173EN	<b>Master 173 Enable.</b> See MAST168EN for definition.
22:20	RW	0	MAST173DEST	<b>Master 173 Destination.</b> See MAST168DEST for definition.
19	RW	1	MAST172EN	<b>Master 172 Enable.</b> See MAST168EN for definition.
18:16	RW	0	MAST172DEST	<b>Master 172 Destination.</b> See MAST168DEST for definition.
15	RW	1	MAST171EN	<b>Master 171 Enable.</b> See MAST168EN for definition.
14:12	RW	0	MAST171DEST	<b>Master 171 Destination.</b> See MAST168DEST for definition.
11	RW	1	MAST170EN	<b>Master 170 Enable.</b> See MAST168EN for definition.
10:8	RW	0	MAST170DEST	<b>Master 170 Destination.</b> See MAST168DEST for definition.
7	RW	1	MAST169EN	<b>Master 169 Enable.</b> See MAST168EN for definition.
6:4	RW	0	MAST169DEST	<b>Master 169 Destination.</b> See MAST168DEST for definition.
3	RW	1	MAST168EN	<b>Master 168 Enable.</b> Enables tracing for Master 168. 0: tracing for Master 168 is disabled. All data received from Master 168 is received at the Input Buffer and immediately dropped. 1: tracing for Master 168 is enabled (Default)
2:0	RW	0	MAST168DEST	<b>Master 168 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 168 trace data. 0x0: Master 168 is routed to Output Port 0 0x1: Master 168 is routed to Output Port 1 ... 0x7: Master 168 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.26 SWDEST\_22: Switching Destination [22]

CSR Register Name: SWDEST_22: Switching Destination [22]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 60	Offset End: 63	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST183EN	<b>Master 183 Enable.</b> See MAST176EN for definition.
30:28	RW	0	MAST183DEST	<b>Master 183 Destination.</b> See MAST176DEST for definition.
27	RW	1	MAST182EN	<b>Master 182 Enable.</b> See MAST176EN for definition.
26:24	RW	0	MAST182DEST	<b>Master 182 Destination.</b> See MAST176DEST for definition.
23	RW	1	MAST181EN	<b>Master 181 Enable.</b> See MAST176EN for definition.
22:20	RW	0	MAST181DEST	<b>Master 181 Destination.</b> See MAST176DEST for definition.
19	RW	1	MAST180EN	<b>Master 180 Enable.</b> See MAST176EN for definition.
18:16	RW	0	MAST180DEST	<b>Master 180 Destination.</b> See MAST176DEST for definition.
15	RW	1	MAST179EN	<b>Master 179 Enable.</b> See MAST176EN for definition.
14:12	RW	0	MAST179DEST	<b>Master 179 Destination.</b> See MAST176DEST for definition.
11	RW	1	MAST178EN	<b>Master 178 Enable.</b> See MAST176EN for definition.
10:8	RW	0	MAST178DEST	<b>Master 178 Destination.</b> See MAST176DEST for definition.
7	RW	1	MAST177EN	<b>Master 177 Enable.</b> See MAST176EN for definition.
6:4	RW	0	MAST177DEST	<b>Master 177 Destination.</b> See MAST176DEST for definition.
3	RW	1	MAST176EN	<b>Master 176 Enable.</b> Enables tracing for Master 176. 0: tracing for Master 176 is disabled. All data received from Master 176 is received at the Input Buffer and immediately dropped. 1: tracing for Master 176 is enabled (Default)
2:0	RW	0	MAST176DEST	<b>Master 176 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 176 trace data. 0x0: Master 176 is routed to Output Port 0 0x1: Master 176 is routed to Output Port 1 ... 0x7: Master 176 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.27 SWDEST\_23: Switching Destination [23]

CSR Register Name: SWDEST_23: Switching Destination [23]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 64	Offset End: 67	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST191EN	<b>Master 191 Enable.</b> See MAST184EN for definition.
30:28	RW	0	MAST191DEST	<b>Master 191 Destination.</b> See MAST184DEST for definition.
27	RW	1	MAST190EN	<b>Master 190 Enable.</b> See MAST184EN for definition.
26:24	RW	0	MAST190DEST	<b>Master 190 Destination.</b> See MAST184DEST for definition.
23	RW	1	MAST189EN	<b>Master 189 Enable.</b> See MAST184EN for definition.
22:20	RW	0	MAST189DEST	<b>Master 189 Destination.</b> See MAST184DEST for definition.
19	RW	1	MAST188EN	<b>Master 188 Enable.</b> See MAST184EN for definition.
18:16	RW	0	MAST188DEST	<b>Master 188 Destination.</b> See MAST184DEST for definition.
15	RW	1	MAST187EN	<b>Master 187 Enable.</b> See MAST184EN for definition.
14:12	RW	0	MAST187DEST	<b>Master 187 Destination.</b> See MAST184DEST for definition.
11	RW	1	MAST186EN	<b>Master 186 Enable.</b> See MAST184EN for definition.
10:8	RW	0	MAST186DEST	<b>Master 186 Destination.</b> See MAST184DEST for definition.
7	RW	1	MAST185EN	<b>Master 185 Enable.</b> See MAST184EN for definition.
6:4	RW	0	MAST185DEST	<b>Master 185 Destination.</b> See MAST184DEST for definition.
3	RW	1	MAST184EN	<b>Master 184 Enable.</b> Enables tracing for Master 184. 0: tracing for Master 184 is disabled. All data received from Master 184 is received at the Input Buffer and immediately dropped. 1: tracing for Master 184 is enabled (Default)
2:0	RW	0	MAST184DEST	<b>Master 184 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 184 trace data. 0x0: Master 184 is routed to Output Port 0 0x1: Master 184 is routed to Output Port 1 ... 0x7: Master 184 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.28 SWDEST\_24: Switching Destination [24]

CSR Register Name: SWDEST_24: Switching Destination [24]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 68	Offset End: 6B	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST199EN	<b>Master 199 Enable.</b> See MAST192EN for definition.
30:28	RW	0	MAST199DEST	<b>Master 199 Destination.</b> See MAST192DEST for definition.
27	RW	1	MAST198EN	<b>Master 198 Enable.</b> See MAST192EN for definition.
26:24	RW	0	MAST198DEST	<b>Master 198 Destination.</b> See MAST192DEST for definition.
23	RW	1	MAST197EN	<b>Master 197 Enable.</b> See MAST192EN for definition.
22:20	RW	0	MAST197DEST	<b>Master 197 Destination.</b> See MAST192DEST for definition.
19	RW	1	MAST196EN	<b>Master 196 Enable.</b> See MAST192EN for definition.
18:16	RW	0	MAST196DEST	<b>Master 196 Destination.</b> See MAST192DEST for definition.
15	RW	1	MAST195EN	<b>Master 195 Enable.</b> See MAST192EN for definition.
14:12	RW	0	MAST195DEST	<b>Master 195 Destination.</b> See MAST192DEST for definition.
11	RW	1	MAST194EN	<b>Master 194 Enable.</b> See MAST192EN for definition.
10:8	RW	0	MAST194DEST	<b>Master 194 Destination.</b> See MAST192DEST for definition.
7	RW	1	MAST193EN	<b>Master 193 Enable.</b> See MAST192EN for definition.
6:4	RW	0	MAST193DEST	<b>Master 193 Destination.</b> See MAST192DEST for definition.
3	RW	1	MAST192EN	<b>Master 192 Enable.</b> Enables tracing for Master 192. 0: tracing for Master 192 is disabled. All data received from Master 192 is received at the Input Buffer and immediately dropped. 1: tracing for Master 192 is enabled (Default)
2:0	RW	0	MAST192DEST	<b>Master 192 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 192 trace data. 0x0: Master 192 is routed to Output Port 0 0x1: Master 192 is routed to Output Port 1 ... 0x7: Master 192 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.29 SWDEST\_25: Switching Destination [25]

CSR Register Name: SWDEST_25: Switching Destination [25]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 6C	Offset End: 6F	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST207EN	<b>Master 207 Enable.</b> See MAST200EN for definition.
30:28	RW	0	MAST207DEST	<b>Master 207 Destination.</b> See MAST200DEST for definition.
27	RW	1	MAST206EN	<b>Master 206 Enable.</b> See MAST200EN for definition.
26:24	RW	0	MAST206DEST	<b>Master 206 Destination.</b> See MAST200DEST for definition.
23	RW	1	MAST205EN	<b>Master 205 Enable.</b> See MAST200EN for definition.
22:20	RW	0	MAST205DEST	<b>Master 205 Destination.</b> See MAST200DEST for definition.
19	RW	1	MAST204EN	<b>Master 204 Enable.</b> See MAST200EN for definition.
18:16	RW	0	MAST204DEST	<b>Master 204 Destination.</b> See MAST200DEST for definition.
15	RW	1	MAST203EN	<b>Master 203 Enable.</b> See MAST200EN for definition.
14:12	RW	0	MAST203DEST	<b>Master 203 Destination.</b> See MAST200DEST for definition.
11	RW	1	MAST202EN	<b>Master 202 Enable.</b> See MAST200EN for definition.
10:8	RW	0	MAST202DEST	<b>Master 202 Destination.</b> See MAST200DEST for definition.
7	RW	1	MAST201EN	<b>Master 201 Enable.</b> See MAST200EN for definition.
6:4	RW	0	MAST201DEST	<b>Master 201 Destination.</b> See MAST200DEST for definition.
3	RW	1	MAST200EN	<b>Master 200 Enable.</b> Enables tracing for Master 200. 0: tracing for Master 200 is disabled. All data received from Master 200 is received at the Input Buffer and immediately dropped. 1: tracing for Master 200 is enabled (Default)
2:0	RW	0	MAST200DEST	<b>Master 200 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 200 trace data. 0x0: Master 200 is routed to Output Port 0 0x1: Master 200 is routed to Output Port 1 ... 0x7: Master 200 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.30 SWDEST\_26: Switching Destination [26]

CSR Register Name: SWDEST_26: Switching Destination [26]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 70	Offset End: 73	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST215EN	<b>Master 215 Enable.</b> See MAST208EN for definition.
30:28	RW	0	MAST215DEST	<b>Master 215 Destination.</b> See MAST208DEST for definition.
27	RW	1	MAST214EN	<b>Master 214 Enable.</b> See MAST208EN for definition.
26:24	RW	0	MAST214DEST	<b>Master 214 Destination.</b> See MAST208DEST for definition.
23	RW	1	MAST213EN	<b>Master 213 Enable.</b> See MAST208EN for definition.
22:20	RW	0	MAST213DEST	<b>Master 213 Destination.</b> See MAST208DEST for definition.
19	RW	1	MAST212EN	<b>Master 212 Enable.</b> See MAST208EN for definition.
18:16	RW	0	MAST212DEST	<b>Master 212 Destination.</b> See MAST208DEST for definition.
15	RW	1	MAST211EN	<b>Master 211 Enable.</b> See MAST208EN for definition.
14:12	RW	0	MAST211DEST	<b>Master 211 Destination.</b> See MAST208DEST for definition.
11	RW	1	MAST210EN	<b>Master 210 Enable.</b> See MAST208EN for definition.
10:8	RW	0	MAST210DEST	<b>Master 210 Destination.</b> See MAST208DEST for definition.
7	RW	1	MAST209EN	<b>Master 209 Enable.</b> See MAST208EN for definition.
6:4	RW	0	MAST209DEST	<b>Master 209 Destination.</b> See MAST208DEST for definition.
3	RW	1	MAST208EN	<b>Master 208 Enable.</b> Enables tracing for Master 208. 0: tracing for Master 208 is disabled. All data received from Master 208 is received at the Input Buffer and immediately dropped. 1: tracing for Master 208 is enabled (Default)
2:0	RW	0	MAST208DEST	<b>Master 208 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 208 trace data. 0x0: Master 208 is routed to Output Port 0 0x1: Master 208 is routed to Output Port 1 ... 0x7: Master 208 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.31 SWDEST\_27: Switching Destination [27]

CSR Register Name: SWDEST_27: Switching Destination [27]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 74	Offset End: 77	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST223EN	<b>Master 223 Enable.</b> See MAST216EN for definition.
30:28	RW	0	MAST223DEST	<b>Master 223 Destination.</b> See MAST216DEST for definition.
27	RW	1	MAST222EN	<b>Master 222 Enable.</b> See MAST216EN for definition.
26:24	RW	0	MAST222DEST	<b>Master 222 Destination.</b> See MAST216DEST for definition.
23	RW	1	MAST221EN	<b>Master 221 Enable.</b> See MAST216EN for definition.
22:20	RW	0	MAST221DEST	<b>Master 221 Destination.</b> See MAST216DEST for definition.
19	RW	1	MAST220EN	<b>Master 220 Enable.</b> See MAST216EN for definition.
18:16	RW	0	MAST220DEST	<b>Master 220 Destination.</b> See MAST216DEST for definition.
15	RW	1	MAST219EN	<b>Master 219 Enable.</b> See MAST216EN for definition.
14:12	RW	0	MAST219DEST	<b>Master 219 Destination.</b> See MAST216DEST for definition.
11	RW	1	MAST218EN	<b>Master 218 Enable.</b> See MAST216EN for definition.
10:8	RW	0	MAST218DEST	<b>Master 218 Destination.</b> See MAST216DEST for definition.
7	RW	1	MAST217EN	<b>Master 217 Enable.</b> See MAST216EN for definition.
6:4	RW	0	MAST217DEST	<b>Master 217 Destination.</b> See MAST216DEST for definition.
3	RW	1	MAST216EN	<b>Master 216 Enable.</b> Enables tracing for Master 216. 0: tracing for Master 216 is disabled. All data received from Master 216 is received at the Input Buffer and immediately dropped. 1: tracing for Master 216 is enabled (Default)
2:0	RW	0	MAST216DEST	<b>Master 216 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 216 trace data. 0x0: Master 216 is routed to Output Port 0 0x1: Master 216 is routed to Output Port 1 ... 0x7: Master 216 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.32 SWDEST\_28: Switching Destination [28]

CSR Register Name: SWDEST_28: Switching Destination [28]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 78	Offset End: 7B	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST231EN	<b>Master 231 Enable.</b> See MAST224EN for definition.
30:28	RW	0	MAST231DEST	<b>Master 231 Destination.</b> See MAST224DEST for definition.
27	RW	1	MAST230EN	<b>Master 230 Enable.</b> See MAST224EN for definition.
26:24	RW	0	MAST230DEST	<b>Master 230 Destination.</b> See MAST224DEST for definition.
23	RW	1	MAST229EN	<b>Master 229 Enable.</b> See MAST224EN for definition.
22:20	RW	0	MAST229DEST	<b>Master 229 Destination.</b> See MAST224DEST for definition.
19	RW	1	MAST228EN	<b>Master 228 Enable.</b> See MAST224EN for definition.
18:16	RW	0	MAST228DEST	<b>Master 228 Destination.</b> See MAST224DEST for definition.
15	RW	1	MAST227EN	<b>Master 227 Enable.</b> See MAST224EN for definition.
14:12	RW	0	MAST227DEST	<b>Master 227 Destination.</b> See MAST224DEST for definition.
11	RW	1	MAST226EN	<b>Master 226 Enable.</b> See MAST224EN for definition.
10:8	RW	0	MAST226DEST	<b>Master 226 Destination.</b> See MAST224DEST for definition.
7	RW	1	MAST225EN	<b>Master 225 Enable.</b> See MAST224EN for definition.
6:4	RW	0	MAST225DEST	<b>Master 225 Destination.</b> See MAST224DEST for definition.
3	RW	1	MAST224EN	<b>Master 224 Enable.</b> Enables tracing for Master 224. 0: tracing for Master 224 is disabled. All data received from Master 224 is received at the Input Buffer and immediately dropped. 1: tracing for Master 224 is enabled (Default)
2:0	RW	0	MAST224DEST	<b>Master 224 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 224 trace data. 0x0: Master 224 is routed to Output Port 0 0x1: Master 224 is routed to Output Port 1 ... 0x7: Master 224 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.33 SWDEST\_29: Switching Destination [29]

CSR Register Name: SWDEST_29: Switching Destination [29]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 7C	Offset End: 7F	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST239EN	<b>Master 239 Enable.</b> See MAST232EN for definition.
30:28	RW	0	MAST239DEST	<b>Master 239 Destination.</b> See MAST232DEST for definition.
27	RW	1	MAST238EN	<b>Master 238 Enable.</b> See MAST232EN for definition.
26:24	RW	0	MAST238DEST	<b>Master 238 Destination.</b> See MAST232DEST for definition.
23	RW	1	MAST237EN	<b>Master 237 Enable.</b> See MAST232EN for definition.
22:20	RW	0	MAST237DEST	<b>Master 237 Destination.</b> See MAST232DEST for definition.
19	RW	1	MAST236EN	<b>Master 236 Enable.</b> See MAST232EN for definition.
18:16	RW	0	MAST236DEST	<b>Master 236 Destination.</b> See MAST232DEST for definition.
15	RW	1	MAST235EN	<b>Master 235 Enable.</b> See MAST232EN for definition.
14:12	RW	0	MAST235DEST	<b>Master 235 Destination.</b> See MAST232DEST for definition.
11	RW	1	MAST234EN	<b>Master 234 Enable.</b> See MAST232EN for definition.
10:8	RW	0	MAST234DEST	<b>Master 234 Destination.</b> See MAST232DEST for definition.
7	RW	1	MAST233EN	<b>Master 233 Enable.</b> See MAST232EN for definition.
6:4	RW	0	MAST233DEST	<b>Master 233 Destination.</b> See MAST232DEST for definition.
3	RW	1	MAST232EN	<b>Master 232 Enable.</b> Enables tracing for Master 232. 0: tracing for Master 232 is disabled. All data received from Master 232 is received at the Input Buffer and immediately dropped. 1: tracing for Master 232 is enabled (Default)
2:0	RW	0	MAST232DEST	<b>Master 232 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 232 trace data. 0x0: Master 232 is routed to Output Port 0 0x1: Master 232 is routed to Output Port 1 ... 0x7: Master 232 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



## 14.2.34 SWDEST\_30: Switching Destination [30]

CSR Register Name: SWDEST_30: Switching Destination [30]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 80	Offset End: 83	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST247EN	<b>Master 247 Enable.</b> See MAST240EN for definition.
30:28	RW	0	MAST247DEST	<b>Master 247 Destination.</b> See MAST240DEST for definition.
27	RW	1	MAST246EN	<b>Master 246 Enable.</b> See MAST240EN for definition.
26:24	RW	0	MAST246DEST	<b>Master 246 Destination.</b> See MAST240DEST for definition.
23	RW	1	MAST245EN	<b>Master 245 Enable.</b> See MAST240EN for definition.
22:20	RW	0	MAST245DEST	<b>Master 245 Destination.</b> See MAST240DEST for definition.
19	RW	1	MAST244EN	<b>Master 244 Enable.</b> See MAST240EN for definition.
18:16	RW	0	MAST244DEST	<b>Master 244 Destination.</b> See MAST240DEST for definition.
15	RW	1	MAST243EN	<b>Master 243 Enable.</b> See MAST240EN for definition.
14:12	RW	0	MAST243DEST	<b>Master 243 Destination.</b> See MAST240DEST for definition.
11	RW	1	MAST242EN	<b>Master 242 Enable.</b> See MAST240EN for definition.
10:8	RW	0	MAST242DEST	<b>Master 242 Destination.</b> See MAST240DEST for definition.
7	RW	1	MAST241EN	<b>Master 241 Enable.</b> See MAST240EN for definition.
6:4	RW	0	MAST241DEST	<b>Master 241 Destination.</b> See MAST240DEST for definition.
3	RW	1	MAST240EN	<b>Master 240 Enable.</b> Enables tracing for Master 240. 0: tracing for Master 240 is disabled. All data received from Master 240 is received at the Input Buffer and immediately dropped. 1: tracing for Master 240 is enabled (Default)
2:0	RW	0	MAST240DEST	<b>Master 240 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 240 trace data. 0x0: Master 240 is routed to Output Port 0 0x1: Master 240 is routed to Output Port 1 ... 0x7: Master 240 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.

### 14.2.35 SWDEST\_31: Switching Destination [31]

CSR Register Name: SWDEST_31: Switching Destination [31]				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 84	Offset End: 87	
Bits	Access	Default	Label	Bit Description
31	RW	1	MAST255EN	<b>Master 255 Enable.</b> See MAST248EN for definition.
30:28	RW	0	MAST255DEST	<b>Master 255 Destination.</b> See MAST248DEST for definition.
27	RW	1	MAST254EN	<b>Master 254 Enable.</b> See MAST248EN for definition.
26:24	RW	0	MAST254DEST	<b>Master 254 Destination.</b> See MAST248DEST for definition.
23	RW	1	MAST253EN	<b>Master 253 Enable.</b> See MAST248EN for definition.
22:20	RW	0	MAST253DEST	<b>Master 253 Destination.</b> See MAST248DEST for definition.
19	RW	1	MAST252EN	<b>Master 252 Enable.</b> See MAST248EN for definition.
18:16	RW	0	MAST252DEST	<b>Master 252 Destination.</b> See MAST248DEST for definition.
15	RW	1	MAST251EN	<b>Master 251 Enable.</b> See MAST248EN for definition.
14:12	RW	0	MAST251DEST	<b>Master 251 Destination.</b> See MAST248DEST for definition.
11	RW	1	MAST250EN	<b>Master 250 Enable.</b> See MAST248EN for definition.
10:8	RW	0	MAST250DEST	<b>Master 250 Destination.</b> See MAST248DEST for definition.
7	RW	1	MAST249EN	<b>Master 249 Enable.</b> See MAST248EN for definition.
6:4	RW	0	MAST249DEST	<b>Master 249 Destination.</b> See MAST248DEST for definition.
3	RW	1	MAST248EN	<b>Master 248 Enable.</b> Enables tracing for Master 248. 0: tracing for Master 248 is disabled. All data received from Master 248 is received at the Input Buffer and immediately dropped. 1: tracing for Master 248 is enabled (Default)
2:0	RW	0	MAST248DEST	<b>Master 248 Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for Master 248 trace data. 0x0: Master 248 is routed to Output Port 0 0x1: Master 248 is routed to Output Port 1 ... 0x7: Master 248 is routed to Output Port 0x7 The default value of this bit field is controlled by the corresponding MDEST[N] parameter.



### 14.2.36 GSWDEST: General Software Trace Destination

CSR Register Name: GSWDEST: General Software Trace Destination				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 88	Offset End: 8B
Bits	Access	Default	Label	Bit Description
31:4	RO	0	Reserved	GSWDEST Reserved
3	RW	1	MAST256EN	<b>Master 256 (and higher) Enable.</b> Enables tracing for Masters numbered 256 and higher. 0: tracing for Masters 256 and higher is disabled. All data received from these masters is received at the Input Buffer and immediately dropped. 1: tracing for Masters 256 and higher is enabled (Default)
2:0	RW	0	MAST256DEST	<b>Master 256 (and higher) Destination.</b> Specifies the destination port number (refer to GTHOPT register for details) for trace data from Masters numbered 256 and higher. 0x0: Trace data is routed to Output Port 0 0x1: Trace data is routed to Output Port 1 ... 0x7: Trace data is routed to Output Port 0x7

### 14.2.37 LWMO: Low WaterMark for Sources 0-7

CSR Register Name: LWMO: Low WaterMark for Sources 0-7				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 8C	Offset End: 8F
Bits	Access	Default	Label	Bit Description
31:28	RW	0	LWM7	Low Water Mark for Trace Source #7. See LWMO.
27:24	RW	0	LWM6	Low Water Mark for Trace Source #6. See LWMO.
23:20	RW	0	LWM5	Low Water Mark for Trace Source #5. See LWMO.
19:16	RW	0	LWM4	Low Water Mark for Trace Source #4. See LWMO.
15:12	RW	0	LWM3	Low Water Mark for Trace Source #3. See LWMO.
11:8	RW	0	LWM2	Low Water Mark for Trace Source #2. See LWMO.
7:4	RW	0	LWM1	Low Water Mark for Trace Source #1. See LWMO.
3:0	RW	0	LWMO	<b>Low Water Mark for Trace Source #0.</b> Indicates the maximum level to which the input buffer for trace source #0 can be filled before the lowWaterMark signal will be de-asserted. That is, the lowWaterMark signal is asserted when the input buffer has LWMO entries or less. Conversely, when greater than LWMO entries are filled, lowWaterMark is deasserted. With a default value of 0, the lowWaterMark signal serves as an empty signal. In practice the LWM value for a trace source should be set to the grant duration minus 1 (GrantDur-1).

#### 14.2.38 LWM1: Low WaterMark for Sources 7-15

CSR Register Name: LWM1: Low WaterMark for Sources 7-15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 90	Offset End: 93
Bits	Access	Default	Label	Bit Description	
31:28	RW	0	LWM15	Low Water Mark for Trace Source #15. See LWM0.	
27:24	RW	0	LWM14	Low Water Mark for Trace Source #14. See LWM0.	
23:20	RW	0	LWM13	Low Water Mark for Trace Source #13. See LWM0.	
19:16	RW	0	LWM12	Low Water Mark for Trace Source #12. See LWM0.	
15:12	RW	0	LWM11	Low Water Mark for Trace Source #11. See LWM0.	
11:8	RW	0	LWM10	Low Water Mark for Trace Source #10. See LWM0.	
7:4	RW	0	LWM9	Low Water Mark for Trace Source #9. See LWM0.	
3:0	RW	0	LWM8	Low Water Mark for Trace Source #8. See LWM0.	

#### 14.2.39 GTH\_INFO\_1: GTH Parameter Info 1

CSR Register Name: GTH_INFO_1: GTH Parameter Info 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 94	Offset End: 97
Bits	Access	Default	Label	Bit Description	
3:0	RO	4	CTS_SIGNAL_WIDTH	<b>CTS External Signal Width.</b> Indicates the number of external (external to NPK, not the SoC) signals available to the SOC for its own use (implementation specific use).	

#### 14.2.40 GTH\_MISC: GTH Miscellaneous Register

CSR Register Name: GTH_MISC: GTH Miscellaneous Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 98	Offset End: 9B
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	RSVD	Reserved	
15:12	RW	0	TUEVT7SRC	<b>Trigger Unit Event 7 Source.</b> 000: VER7 001: bpb_data_xfer[0] (MSC0) 010: bpb_data_xfer[1] (MSC1) 011: bpb_data_xfer[2] 100: bpb_data_xfer[3] ... 111: bpb_data_xfer[6] Note that not all values are valid for all implementations. Only values that correspond to the number of trace destinations are valid. All other values will alias.	



CSR Register Name: GTH_MISC: GTH Miscellaneous Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 98
Bits	Access	Default	Label	Bit Description
11:8	RW	0	TUEVT6SRC	<b>Trigger Unit Event 6 Source.</b> 000: VER6 001: bpb_data_xfer[0] (MSC0) 010: bpb_data_xfer[1] (MSC1) 011: bpb_data_xfer[2] 100: bpb_data_xfer[3] ... 111: bpb_data_xfer[6] Note that not all values are valid for all implementations. Only values that correspond to the number of trace destinations are valid. All other values will alias.
7:1	RO	0	RSVD2	Reserved
0	RW	0	RSVD	Reserved for future use

#### 14.2.41 SMCRO: STP Maintenance Control Register 0

CSR Register Name: SMCRO: STP Maintenance Control Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 9C
Bits	Access	Default	Label	Bit Description
31:16	RW	8000	SYNCF1	<b>Sync Packet Frequency for Port 1.</b> Specifies the number data sets between Maintenance packets for port 1. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.
15:0	RW	8000	SYNCFO	<b>Sync Packet Frequency for Port 0.</b> Specifies the number of data sets between Maintenance packets for port 0. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.

#### 14.2.42 SMCR1: STP Maintenance Control Register 1

CSR Register Name: SMCR1: STP Maintenance Control Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0
Bits	Access	Default	Label	Bit Description
31:16	RW	8000	SYNCF3	<b>Sync Packet Frequency for Port 3.</b> Specifies the number of data sets between Maintenance packets for port 3. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.
15:0	RW	8000	SYNCF2	<b>Sync Packet Frequency for Port 2.</b> Specifies the number of data sets between Maintenance packets for port 2. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.

#### 14.2.43 SMCR2: STP Maintenance Control Register 2

CSR Register Name: SMCR2: STP Maintenance Control Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A4
Bits	Access	Default	Label	Bit Description
31:16	RW	8000	SYNCF5	<b>Sync Packet Frequency for Port 5.</b> Specifies the number of data sets between Maintenance packets for port 5. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.
15:0	RW	8000	SYNCF4	<b>Sync Packet Frequency for Port 4.</b> Specifies the number of data sets between Maintenance packets for port 4. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.



#### 14.2.44 SMCR3: STP Maintenance Control Register 3

CSR Register Name: SMCR3: STP Maintenance Control Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A8	Offset End: AB
Bits	Access	Default	Label	Bit Description	
31:16	RW	8000	SYNCF7	<b>Sync Packet Frequency for Port 7.</b> Specifies the number of data sets between Maintenance packets for port 5. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.	
15:0	RW	8000	SYNCF6	<b>Sync Packet Frequency for Port 6.</b> Specifies the number of data sets between Maintenance packets for port 4. A value of 0 (zero) turns off Maintenance packet generation for the port. However, an initial maintenance packet will be sent at the start of the trace so that reconstruction software can successfully reconstruct the trace data.	

#### 14.2.45 PGDO: Programmable Grant Duration Register 0

CSR Register Name: PGDO: Programmable Grant Duration Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: AC	Offset End: AF
Bits	Access	Default	Label	Bit Description	
31:28	RW	1	GNTDUR_7	<b>Programmable Grant Duration for Input Port 7.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
27:24	RW	1	GNTDUR_6	<b>Programmable Grant Duration for Input Port 6.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
23:20	RW	1	GNTDUR_5	<b>Programmable Grant Duration for Input Port 5.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
19:16	RW	1	GNTDUR_4	<b>Programmable Grant Duration for Input Port 4.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
15:12	RW	1	GNTDUR_3	<b>Programmable Grant Duration for Input Port 3.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
11:8	RW	1	GNTDUR_2	<b>Programmable Grant Duration for Input Port 2.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	

<b>CSR Register Name:</b> PGD0: Programmable Grant Duration Register 0					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> AC	<b>Offset End:</b> AF
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
7:4	RW	1	GNTDUR_1	<b>Programmable Grant Duration for Input Port 1.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
3:0	RW	1	GNTDUR_0	<b>Programmable Grant Duration for Input Port 0.</b> The value in this register specifies the maximum number of consecutive grants (or 'get's) that can be given to GTH Input Port 0 (SMU).	

#### 14.2.46 PGD1: Programmable Grant Duration Register 1

<b>CSR Register Name:</b> PGD1: Programmable Grant Duration Register 1					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> B0	<b>Offset End:</b> B3
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31:28	RW	1	GNTDUR_15	<b>Programmable Grant Duration for Input Port 15.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
27:24	RW	1	GNTDUR_14	<b>Programmable Grant Duration for Input Port 14.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
23:20	RW	1	GNTDUR_13	<b>Programmable Grant Duration for Input Port 13.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
19:16	RW	1	GNTDUR_12	<b>Programmable Grant Duration for Input Port 12.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
15:12	RW	1	GNTDUR_11	<b>Programmable Grant Duration for Input Port 11.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
11:8	RW	1	GNTDUR_10	<b>Programmable Grant Duration for Input Port 10.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	
7:4	RW	1	GNTDUR_9	<b>Programmable Grant Duration for Input Port 9.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist	



CSR Register Name: PGD1: Programmable Grant Duration Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: B0
Bits	Access	Default	Label	Bit Description
3:0	RW	1	GNTDUR_8	<b>Programmable Grant Duration for Input Port 8.</b> See definition for GNTDUR_0 for details. These bits are read-only as 0h if the trace source (input port) does not exist

#### 14.2.47 SCR: Source Control Register

CSR Register Name: SCR: Source Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: C8
Bits	Access	Default	Label	Bit Description
31:24	RO	0	Reserved	SCR Reserved
23	RW	0	StoreEnOvrd7	<b>Storage Enable Override 7.</b> See StoreEnOvrd2 for details.
22	RW	0	StoreEnOvrd6	<b>Storage Enable Override 6.</b> See StoreEnOvrd2 for details.
21	RW	0	StoreEnOvrd5	<b>Storage Enable Override 5.</b> See StoreEnOvrd2 for details.
20	RW	0	StoreEnOvrd4	<b>Storage Enable Override 4.</b> See StoreEnOvrd2 for details.
19	RW	0	StoreEnOvrd3	<b>Storage Enable Override 3.</b> See StoreEnOvrd2 for details.
18	RW	0	StoreEnOvrd2	<b>Storage Enable Override 2.</b> Under normal operation the CTS gasket controls the StoreEn[N:0] signals to the trace sources. This Storage Enable Override configuration bit is a method to force the StoreEn[N:0] signals without requiring programming of the CTS or the gasket. The StoreEnOvrd is logically ORed together with the StoreEn from the gasket before being sent to the trace sources.
17:8	RO	0	Reserved	SCR Reserved
7	RW	0	STM7	Storage Mode control bit for Source 7. See STM0 for details.
6	RW	0	STM6	Storage Mode control bit for Source 6. See STM0 for details.
5	RW	0	STM5	Storage Mode control bit for Source 5. See STM0 for details.
4	RW	0	STM4	Storage Mode control bit for Source 4. See STM0 for details.
3	RW	0	STM3	Storage Mode control bit for Source 3. See STM0 for details.



CSR Register Name: SCR: Source Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: C8
Bits	Access	Default	Label	Bit Description
2	RW	0	STM2	Storage Mode control bit for Source 2. See STM0 for details.
1	RW	0	STM1	Storage Mode control bit for Source 1. See STM0 for details.
0	RW	0	STM0	<b>Storage Mode control bit for Source 0.</b> 0: Normal mode. Trace source is stored in single logical buffer. 1: Multi-buffer. Trace source is stored in separate logical buffers with wrapping around trigger assertion

#### 14.2.48 GTHFRQ: GTH Frequency Register

CSR Register Name: GTHFRQ: GTH Frequency Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: CC
Bits	Access	Default	Label	Bit Description
31:0	RW	B71B0000	GTH_FREQ	<b>GTH Operating Frequency.</b> Specifies the apparent operating frequency of the GTH, in Hertz. This is also clock with which timestamps are generated.

#### 14.2.49 BPB\_FRAME\_SIZE: Byte Packing Buffer max frame size

CSR Register Name: BPB_FRAME_SIZE: Byte Packing Buffer max frame size				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: D0
Bits	Access	Default	Label	Bit Description
31:24	RW	0	FRAME_SZ3	<b>BPB Max Frame Size for output port 3.</b> Specifies the maximum frame size for output port 0, in terms of number of BPB rows. To determine number bytes, multiply by the width of the BPB. 0: Normal Mode (default) 1-255: Framing Mode.
23:16	RW	0	FRAME_SZ2	<b>BPB Max Frame Size for output port 2.</b> Specifies the maximum frame size for output port 0, in terms of number of BPB rows. To determine number bytes, multiply by the width of the BPB. 0: Normal Mode (default) 1-255: Framing Mode.
15:8	RW	0	FRAME_SZ1	<b>BPB Max Frame Size for output port 1.</b> Specifies the maximum frame size for output port 0, in terms of number of BPB rows. To determine number bytes, multiply by the width of the BPB. 0: Normal Mode (default) 1-255: Framing Mode.

<b>CSR Register Name:</b> BPB_FRAME_SIZE: Byte Packing Buffer max frame size				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> D0
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
7:0	RW	0	FRAME_SZ0	<b>BPB Max Frame Size for output port 0.</b> Specifies the maximum frame size for output port 0, in terms of number of BPB rows. To determine number bytes, multiply by the width of the BPB. 0: Normal Mode (default) 1-255: Framing Mode.

#### 14.2.50 GTHSTAT: GTH Status Register

<b>CSR Register Name:</b> GTHSTAT: GTH Status Register				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> D4
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:24	RO	0	Rsvd	Rsvd
23	RO	0	TUSE7	<b>TU Store Enable, Trace Source 7.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
22	RO	0	TUSE6	<b>TU Store Enable, Trace Source 6.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
21	RO	0	TUSE5	<b>TU Store Enable, Trace Source 5.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
20	RO	0	TUSE4	<b>TU Store Enable, Trace Source 4.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
19	RO	0	TUSE3	<b>TU Store Enable, Trace Source 3.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
18	RO	0	TUSE2	<b>TU Store Enable, Trace Source 2.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
17	RO	0	TUSE1	<b>TU Store Enable, Trace Source 1.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.
16	RO	0	TUSE0	<b>TU Store Enable, Trace Source 0.</b> Indicates whether the trigger unit Store Enable signal is asserted or de-asserted. See the CTS Gasket detail drawing for more information.

<b>CSR Register Name: GTHSTAT: GTH Status Register</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: D4</b>	<b>Offset End: D7</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
15	RO	1	IBE7	Input Buffer Empty, Trace Source 7. See IBE0 for details.	
14	RO	1	IBE6	Input Buffer Empty, Trace Source 6. See IBE0 for details.	
13	RO	1	IBE5	Input Buffer Empty, Trace Source 5. See IBE0 for details.	
12	RO	1	IBE4	Input Buffer Empty, Trace Source 4. See IBE0 for details.	
11	RO	1	IBE3	Input Buffer Empty, Trace Source 3. See IBE0 for details.	
10	RO	1	IBE2	Input Buffer Empty, Trace Source 2. See IBE0 for details.	
9	RO	1	IBE1	Input Buffer Empty, Trace Source 1. See IBE0 for details.	
8	RO	1	IBE0	<b>Input Buffer Empty, Trace Source 0.</b> Indicates that the input buffer for trace source 0 is empty, and has absolutely no data in it. This information will be used to determine whether or not a trace source (e.g., STH, ODLA) is inactive and drained of data.	
7	RO	1	PLE7	<b>Pipe Line Empty, Output port 7.</b> See PLE0 for details.	
6	RO	1	PLE6	<b>Pipe Line Empty, Output port 6.</b> See PLE0 for details.	
5	RO	1	PLE5	<b>Pipe Line Empty, Output port 5.</b> See PLE0 for details.	
4	RO	1	PLE4	<b>Pipe Line Empty, Output port 4.</b> See PLE0 for details.	
3	RO	1	PLE3	<b>Pipe Line Empty, Output port 3.</b> See PLE0 for details.	
2	RO	1	PLE2	<b>Pipe Line Empty, Output port 2.</b> See PLE0 for details.	
1	RO	1	PLE1	<b>Pipe Line Empty, Output port 1.</b> See PLE0 for details	
0	RO	1	PLE0	<b>Pipe Line Empty, Output port 0.</b> Indicates that the data pipeline through GTH, from Input Buffer to Byte Packing Buffer, is empty. Software can read this bit to determine if the pipeline is empty. Along with an equivalent bit from the output port controller (e.g., MSU, PTI, HTI), software can determine if all the data has been sent to the destination, and processing of the data can begin.	



### 14.2.51 SCR2: Source Control Register 2

CSR Register Name: SCR2: Source Control Register 2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: D8	Offset End: DB
Bits	Access	Default	Label	Bit Description
31:24	RO	0	Reserved2	Reserved
23	RW	0	FSQOFF7	<b>Force StoreQual Off for Trace Source 7.</b> Overrides StoreQual[N] signal from the CTS, to force it off.
22	RW	0	FSQOFF6	<b>Force StoreQual Off for Trace Source 6.</b> Overrides StoreQual[N] signal from the CTS, to force it off.
21	RW	0	FSQOFF5	<b>Force StoreQual Off for Trace Source 5.</b> Overrides StoreQual[N] signal from the CTS, to force it off.
20	RW	0	FSQOFF4	<b>Force StoreQual Off for Trace Source 4.</b> Overrides StoreQual[N] signal from the CTS, to force it off.
19	RW	0	FSQOFF3	<b>Force StoreQual Off for Trace Source 2.</b> Overrides StoreQual[N] signal from the CTS, to force it off.
18	RW	0	FSQOFF2	<b>Force StoreQual Off for Trace Source 2.</b> Overrides StoreQual[N] signal from the CTS, to force it off.
17:8	RO	0	Reserved1	SCR2 Reserved
7	RW	0	FSEOFF7	<b>Force StoreEn Off for Trace Source 7.</b> See FSEOFF2 for details.
6	RW	0	FSEOFF6	<b>Force StoreEn Off for Trace Source 6.</b> See FSEOFF2 for details.
5	RW	0	FSEOFF5	<b>Force StoreEn Off for Trace Source 5.</b> See FSEOFF2 for details.
4	RW	0	FSEOFF4	<b>Force StoreEn Off for Trace Source 4.</b> See FSEOFF2 for details.
3	RW	0	FSEOFF3	<b>Force StoreEn Off for Trace Source 3.</b> See FSEOFF2 for details.
2	RW	0	FSEOFF2	<b>Force StoreEn Off for Trace Source 2.</b> Overrides StoreEn[N] signal to force it off. This will override the StoreEnOvrd setting, forcing the storeEn to off (see CTS Gasket Detail drawing for more information)
1	RO	0	Rsvd	SCR2 Reserved
0	RW	0	FCD	<b>Force Capture Done.</b> When set, this bit will force the captureDone signal going from CTS to the CTS Gasket to be asserted.



#### 14.2.52 DESTOVR: Destination Override Register

CSR Register Name: DESTOVR: Destination Override Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: DC	Offset End: DF
Bits	Access	Default	Label	Bit Description	
31:4	RO	0	Rsvd	DESTOVR Reserved	
3	RW	0	DESTOVREN	<b>Destination Override Enable.</b> Setting this bit activates the Destination Override function. All trace data from all masters will be sent to the output port specified by bits 2:0.	
2:0	RW	0	DESTOVR	<b>Destination Override.</b> Overrides all Mast[N]Dest bits in the SWDEST and GSWDEST registers with the value specified in this register field. This has the net effect of sending all trace data to the port specified by these bits. These bits use the same encoding as MAST[N]DEST bits in SWDEST registers.	

#### 14.2.53 SCR PDO: ScratchPad[0] Register

CSR Register Name: SCR PDO: ScratchPad[0] Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: E0	Offset End: E3
Bits	Access	Default	Label	Bit Description	
31:25	RW	0	rsvd	<b>Scratch Pad bits, for use by software/firmware.</b> These bits have no defined usage or function, but may have a defined usage or function in the future. Software/firmware is recommended to treat these as "reserved for future use"	
24	RW	0	DebuggerInUse	<b>Debugger In Use.</b> Used during initialization flows to indicate to BIOS/IA firmware that an external host debugger is actively using this Intel® Trace Hub instance	
23:19	RW	0	Rsvd1	Scratch Pad bits, for use by software/firmware.	
18	RW	0	CtpIsPrimDest	Used during save/restore flows to indicate to PMU that the Intel® Trace Hub primary tracing destination is HTI	
17	RW	0	VERIsEnabled	Indicates that the VER is enabled/active	
16	RW	0	NPKHIsEnabled	Indicates that the DCI TH is enabled/active	
15	RW	0	STHIsEnabled	Indicates that the STH is enabled/active	
14	RW	0	SoCHAPIsEnabled	Indicates that the SoCHAP is enabled/active	
13	RW	0	ODLAIsEnabled	Indicates that the ODLA is enabled/active	
12	RW	0	TriggerIsEnabled	Indicates that a trigger is enabled/active	

<b>CSR Register Name: SCR PDO: ScratchPad[0] Register</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: E0</b>	<b>Offset End: E3</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
11	RW	0	TraceSxExit	Used to indicate to driver that a trace session has been active across an Sx event	
10	RW	0	MSC1IsEnabled	Indicates that MSC1 is enabled/active	
9	RW	0	MSC0IsEnabled	Indicates that MSC0 is enabled/active	
8	RW	0	S5Exit	Used to communicate with driver that a S5 event occurred	
7	RW	0	S4Exit	Used to communicate with driver that a S4 event occurred	
6	RW	0	DeepSxExit	Used to communicate with driver that a DeepSx event occurred	
5	RW	0	BssbIsAltDest	Used during save/restore flows to indicate to PMU that BSSB should be used as an alternate destination	
4	RW	0	PtiIsAltDest	Used during save/restore flows to indicate to PMU that PTI should be used as an alternate destination	
3	RW	0	BssbIsPrimDest	Used during save/restore flows to indicate to PMU that the Intel® Trace Hub primary tracing destination is BSSB	
2	RW	0	PtiIsPrimDest	Used during save/restore flows to indicate to PMU that Intel® Trace Hub primary tracing destination is PTI	
1	RW	0	DbCIsPrimDest	Used during save/restore flows to indicate to PMU that Intel® Trace Hub primary tracing destination is USB DbC	
0	RW	0	MemIsPrimDest	Used during save/restore flows to indicate to PMU that Intel® Trace Hub primary tracing destination is system memory	

#### 14.2.54 SCR PD1: ScratchPad[1] Register

<b>CSR Register Name: SCR PD1: ScratchPad[1] Register</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: E4</b>	<b>Offset End: E7</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31:0	RW	0	BlockAddress	If BIOS used the MTB to store boot data and later relocated that data to system memory (post-MRC), it can save the physical memory block address in this register to communicate it with software driver	

#### 14.2.55 SCRPD2: ScratchPad[2] Register

CSR Register Name: SCRPD2: ScratchPad[2] Register																												
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: E8																								
Bits	Access	Default	Label	Bit Description																								
31:4	RW	0	scpd2	<p><b>Scratch Pad bits, for use by software/firmware.</b> These bits have no defined usage or function, but may have a defined usage or function in the future. Software/firmware is recommended to treat these as "reserved for future use"</p>																								
3:0	RW	0	IAFW_CTRL	<p><b>IA Firmware (BIOS) tracing verbosity and enabling.</b> These bits are used by debug and trace tools to communicate to IA Firmware (BIOS) the desired verbosity level, and whether or not tracing to the Intel® Trace Hub is enabled.</p> <table border="1"> <thead> <tr> <th>Bit</th><th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>3</td><td>x</td><td>Reserved for future use</td></tr> <tr> <td>2:1</td><td>00</td><td>Errors only</td></tr> <tr> <td></td><td>01</td><td>Errors and warnings only</td></tr> <tr> <td></td><td>10</td><td>Errors, warnings, and Info only</td></tr> <tr> <td></td><td>11</td><td>Errors, Warnings, Info, and Verbose (all messages)</td></tr> <tr> <td>0</td><td>0</td><td>Disable Trace</td></tr> <tr> <td></td><td>1</td><td>Enable Trace</td></tr> </tbody> </table>	Bit	Value	Description	3	x	Reserved for future use	2:1	00	Errors only		01	Errors and warnings only		10	Errors, warnings, and Info only		11	Errors, Warnings, Info, and Verbose (all messages)	0	0	Disable Trace		1	Enable Trace
Bit	Value	Description																										
3	x	Reserved for future use																										
2:1	00	Errors only																										
	01	Errors and warnings only																										
	10	Errors, warnings, and Info only																										
	11	Errors, Warnings, Info, and Verbose (all messages)																										
0	0	Disable Trace																										
	1	Enable Trace																										

#### 14.2.56 SCRPD3: ScratchPad[3] Register

CSR Register Name: SCRPD3: ScratchPad[3] Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: EC
Bits	Access	Default	Label	Bit Description
31:0	RW	0	scpd3	<p><b>Scratch Pad bits, for use by software/firmware.</b> These bits have no defined usage or function, but may have a defined usage or function in the future. Software/firmware is recommended to treat these as "reserved for future use"</p>



#### 14.2.57 NDB\_CTRL: NPK DTF Control Register

CSR Register Name: NDB_CTRL: NPK DTF Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: F0	Offset End: F3
Bits	Access	Default	Label	Bit Description	
17	RW	1	NDBMW1	<b>MSC1MemWrap -&gt; dtf_sync.</b> 0: MSC 1MemWrap signal assertion will not cause dtf_sync to assert,  1: MS1MemWrap signal assertion will cause dtf_sync to assert.	
16	RW	1	NDBMW0	<b>MSC0MemWrap -&gt; dtf_sync.</b> 0: MSC 0MemWrap signal assertion will not cause dtf_sync to assert,  1: MS0MemWrap signal assertion will cause dtf_sync to assert.	
15:0	RW	0	NDB_WAITEMPTY	<b>NDB Wait for Empty.</b> Specifies the number of npclk periods the NDB will wait after a storeEndeassertion before it considers the DTF to be empty, and forcibly deassert its valid and sourceActive outputs.	

#### 14.2.58 BPB\_CSRO: BPB Control/Status Register 0

CSR Register Name: BPB_CSRO: BPB Control/Status Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 120	Offset End: 123
Bits	Access	Default	Label	Bit Description	
24	RW	0	BPB_FM1	<b>BPB1 Framing Mode.</b>  0: Maximum Size framing mode,  1: Fixed Size framing mode	
23:16	RW	0	BPB_FTO1	See BPB_FTO0.	
8	RW	0	BPB_FMO	<b>BPB0 Framing Mode.</b>  0: Maximum Size framing mode,  1: Fixed Size framing mode	
7:0	RW	0	BPB_FTO0	<b>BPB0 Frame Timeout.</b> For trace destinations used in framing mode, this register specifies the number of clocks the BPB will wait while not sending data, before forcibly closing the frame. 00h: no timeout. 01h to FFh: The frame will be forcibly closed after BPB_FTO0 clocks have passed with no data being sent.	
24	RW	0	BPB_FM3	<b>BPB3 Framing Mode.</b>  0: Maximum Size framing mode,  1: Fixed Size framing mode	



#### 14.2.59 BPB\_CSR1: BPB Control/Status Register 0

CSR Register Name: BPB_CSR1: BPB Control/Status Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 124	Offset End: 127
Bits	Access	Default	Label	Bit Description	
8	RW	0	BPB_FM2	<b>BPB2 Framing Mode.</b> 0: Maximum Size framing mode, 1: Fixed Size framing mode	

#### 14.2.60 BPB\_CSR1: BPB Control/Status Register 1

CSR Register Name: BPB_CSR1: BPB Control/Status Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 124	Offset End: 127
Bits	Access	Default	Label	Bit Description	
7:0	RW	0	BPB_FTO2	See BPB_FTOO.	

#### 14.2.61 LPP\_CTL: LPP Control Register

CSR Register Name: LPP_CTL: LPP Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 1C00	Offset End: 1C03
Bits	Access	Default	Label	Bit Description	
31	RO	0	LPPBUSY	<b>LPP Busy.</b> When set, indicates that there is data in the LPP output path, waiting to be driven onto the PTI pins or the External Bridge interface (whichever is applicable).	
30	RW/1C	0	BSSBACT	<b>External Bridge Path Active:</b> If set indicates that LPP is waiting on one or more credits from the external bridge. Once cleared then there is no data in flight between LPP and the external bridge. This bit can be cleared by writing 1 to it which will also reset the external bridge credits.	
29:26	RO	0	RSVD	Reserved	
25	RW	0	DEST	<b>LPP Destination.</b> 0: PTI 1: External Bridge Note : the default state of this bit depends on the LPP path in the SoC. If only PTI is supported the default is PTI. If only an external bridge is supported, the default is the external bridge. If both are supported, the default is PTI.	
24	RO	0	RSVD	Reserved	



CSR Register Name: LPP_CTL: LPP Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 1C00	Offset End: 1C03
Bits	Access	Default	Label	Bit Description	
23	RO	0	Reserved	Reserved	
22:20	RW	0	PATGENMOD	<b>Pattern Generation Mode:</b> This field selects one of the five MIPI-PTI training/calibration patterns to drive onto the PTI Port pins. See MIPI PTI Specification for details. 001: Pattern 1 010: Pattern 2 011: Pattern 3 100: Pattern 4 101: Pattern 5 110: Pattern 6 All others reserved for future use.	
19:18	RO	0	Reserved	Reserved	
17:16	RW	0	PTICLKDIV	<b>Clock Divider:</b> This field selects the scaled down version of the clock to use for the trace clock and send to GPIO pins: 00: 1x (Use Intel Trace Hub clock), 01: 2x (Use 1/2 Intel Trace Hub clock), 10: 4x (Use 1/4 Intel Trace Hub clock), 11: 8x (Use 1/8 Intel Trace Hub clock).	
15:10	RO	0	Reserved	Reserved	
9	RO	0	BSSBPRESENT	<b>External Bridge Present:</b> If set indicates that the External Bridge path of LPP is connected to an external (to the Intel Trace Hub) Bridge and this mode is valid.	
8	RO	0	PTIPRESENT	<b>PTI Present:</b> If set indicates that the PTI path of LPP is connected to the GPIO and PTI mode is valid.	
7:4	RW	1	PTIMODESEL	<b>PTI Mode Select:</b> This register selects the PTI port output Mode. The encoding for the field is as follows: 0001: 4-bit mode 0010: 8-bit mode 1000: 16-bit mode All others reserved for future use	
3:2	RO	0	Reserved	Reserved	
1	RW	1	PTIFREECLK	<b>PTI Free-Running Clock.</b> 0: The trace_clock acts as a strobe which only toggles when valid data is present on trace_data pins. 1: The trace_clock will act as a free-running clock. Note : If PTI Free-running clock mode is enabled, the PnFLUSH bit in the GTHOPT register must be set to avoid STP protocol errors at the PTI interface.	
0	RW	1	LPPEN	<b>LPP Enable:</b> 0: LPP output path/port is disabled and clock-gated. 1: LPP output path/port is enabled.	

## 14.3 STH Registers

### 14.3.1 STH Registers Summary

STH Registers			
Offset Start	Offset End	Symbol	Register Name/Function
04000	04003	STHCAPO	STH Capability 0
04004	04007	STHCAPI	STH Capability 1
04008	0400B	TRIG	Create Trigger Packet
0400C	0400F	TRIG_TS	Create Trigger Packet with Time Stamp
04010	04013	XSYNC	Create Cross-Synchronization Packet
04014	04017	XSYNC_TS	Create Cross-Synchronization Packet w Time Stamp
04018	0401B	GERR	Create General Error Packet
0401C	0401F	TRIGTSPP	TRIG_TS packet payload
04020	04023	BDC	Backpressure Duration Counter
04024	04027	DPLC	Data Packet Lost Counter
04028	0402B	EVOAMCH	Event 0 Address Match Register
0402C	0402F	EVOAMSK	Event 0 Address Mask Register
04030	04033	EV0DMCH	Event 0 Data Match Register
04038	0403B	EVOCTRL	Event 0 Match/Mask Control Register
0403C	0403F	EV1AMCH	Event 1 Address Match Register
04040	04043	EV1AMSK	Event 1 Address Mask Register
04044	04047	EV1DMCH	Event 1 Data Match Register
0404C	0404F	EV1CTRL	Event 1 Match/Mask Control Register
04050	04053	EV2AMCH	Event 2 Address Match Register
04054	04057	EV2AMSK	Event 2 Address Mask Register
04058	0405B	EV2DMCH	Event 2 Data Match Register
04060	04063	EV2CTRL	Event 2 Match/Mask Control Register
04064	04067	EV3AMCH	Event 3 Address Match Register
04068	0406B	EV3AMSK	Event 3 Address Mask Register
0406C	0406F	EV3DMCH	Event 3 Data Match Register
04074	04077	EV3CTRL	Event 3 Match/Mask Control Register
04078	0407B	STHMISC	STH MISC Control register 1
0407C	0407F	STHCAPI	STH Capability 2



### 14.3.2 STHCAPO: STH Capability 0

CSR Register Name: STHCAPO: STH Capability 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04000	Offset End: 04003
Bits	Access	Default	Label	Bit Description	
31:16	RO	3FF	STHMNUM	<b>STH Highest Software Master:</b> Indicates the highest numbered software master. Total number of masters is STHMNUM - STHMSTR	
15:0	RO	100	STHMSTR	<b>SW Master Start:</b> Indicates the starting master number for the Software Trace Hub as exposed by the PCI BAR SW_BAR (SW_UBAR + SW_LBAR). This is the same as the value of the hardware parameter SW_MSTR_STRT.	

### 14.3.3 STHCAP1: STH Capability 1

CSR Register Name: STHCAP1: STH Capability 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04004	Offset End: 04007
Bits	Access	Default	Label	Bit Description	
31:24	RW	FF	FW_MSTR	<b>Firmware Master Stop.</b> This field indicates the highest Master number that is allocated to firmware trace sources. The default (reset) value of FW_MSTR_STP is set by the synthesis parameter FW_MSTR_STP. Thereafter, it can be written by any software or firmware to adjust the number of masters allocated to firmware. This field is intended to be used in two ways: When the STH is a single PCI BAR, this register field can be changed according to whatever software model best suits the use case(s) for the implementation. When the STH is split into a fixed ACPI BAR for firmware, and a movable PCI BAR for software, then this register field should be only read, and not written. This is because the value in this register reflects the dividing line between the ACPI BAR and the PCI BAR. If the dividing line is moved, software will have an inaccurate view of the STH.	
23:16	RO	10	RTITCNT	<b>Number of Intel PT Sources.</b> Indicates the number of logical processors supported in this instantiation of the STH. 0h: Intel PT is not implemented and not supported in this instantiation of the Intel TH 1h: 1 logical processor is supported 2h: 2 logical processors 3h: 3 logical processors etc. 0Fh: 15 logical processors	
15:8	RO	0	RSVD1	STHCAPI Reserved	
7:0	RO	80	CHLCNT	<b>Software Channels Per Master:</b> Indicates the number of channels per master for STH.	



#### 14.3.4 TRIG: Create Trigger Packet

CSR Register Name: TRIG: Create Trigger Packet				
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 04008
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	Reserved
7:0	WO	0	TRIGDAT	Trigger Data/Identifier. These bits will be the payload of the TRIG packet

#### 14.3.5 TRIG\_TS: Create Trigger Packet with Time Stamp

CSR Register Name: TRIG_TS: Create Trigger Packet with Time Stamp				
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 0400C
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	TRIG_TS Resereved
7:0	WO	0	TRIGDAT	Trigger Data/Identifier. These bits will be the payload of the TRIG_TS packet

#### 14.3.6 XSYNC: Create Cross-Synchronization Packet

CSR Register Name: XSYNC: Create Cross-Synchronization Packet				
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 04010
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	XSYNC Reserved
7:0	WO	0	XSYNC DAT	Cross-Synchronization Data/Identifier. These bits will be the payload of the XSYNC packet

#### 14.3.7 XSYNC\_TS: Create Cross-Synchronization Packet w Time Stamp

CSR Register Name: XSYNC_TS: Create Cross-Synchronization Packet w Time Stamp				
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 04014
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	XSYNC_TS Reserved
7:0	WO	0	XSYNC DAT	Cross-Synchronization Data/Identifier. These bits will be the payload of the XSYNC_TS packet



### 14.3.8 GERR: Create General Error Packet

CSR Register Name: GERR: Create General Error Packet					
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 04018	Offset End: 0401B
Bits	Access	Default	Label	Bit Description	
31:8	RO	0	RSVD	GERR Reserved	
7:0	WO	0	GERRDAT	<b>Create General Error Packet, Error Data/Identifier.</b> These bits will be the payload of the GERR packet	

### 14.3.9 TRIGTSPP: TRIG\_TS packet payload

CSR Register Name: TRIGTSPP: TRIG_TS packet payload					
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 0401C	Offset End: 0401F
Bits	Access	Default	Label	Bit Description	
31:9	RO	0	RSVD	TRIGTSPP Reserved	
8	RW	0	TRIGEN	<b>Trigger packet Enable.</b> This bit determines if a TRIG_TS packet is generated when the trigger input from the CTS is asserted 0: do not create a TRIG packet 1: create a TRIG packet with payload as specified in TRIGVAL.	
7:0	RW	0	TRIGVAL	<b>TRIG Value:</b> Specifies the payload in the TRIG_TS packet when the STH detects the trigger input from the CTS is asserted.	

### 14.3.10 BDC: Backpressure Duration Counter

CSR Register Name: BDC: Backpressure Duration Counter					
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 04020	Offset End: 04023
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	BDC	<b>Backpressure Duration Count value.</b> Indicates the number of clock cycles that STH is allowed to backpressure the host interface. See the Software Trace Hub chapter for details. Note that the default value indicates that STH will start dropping as soon as it cannot process any more data.	

### 14.3.11 DPLC: Data Packet Lost Counter

CSR Register Name: DPLC: Data Packet Lost Counter					
Bar: CSR_MTB_BAR		Reset: npk_RST_B		Offset Start: 04024	Offset End: 04027
Bits	Access	Default	Label	Bit Description	
31:17	RO	0	RSVD	DPLC Reserved	

<b>CSR Register Name:</b> DPLC: Data Packet Lost Counter					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 04024	<b>Offset End:</b> 04027
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
16	RW/1C	0	DPLCO	<b>Data Packets Lost Counter Overflow:</b> this is a sticky bit that indicates that the DPLC value has wrapped around at least once. This will be sent as the most significant bit of the number of packets lost after existing Drop Mode	
15:0	RO	0	DPLCV	<b>Data Packets Lost Counter Value:</b> The number of packets (not including triggers initiated by CTS) that have been dropped during a backpressure release period. This register field is only valid while data is being dropped by the STH. Once the STH is lossless again, the DPLC value is sent as a USER packet in the trace stream, and this register field is invalid.	

#### 14.3.12 EVOAMCH: Event 0 Address Match Register

<b>CSR Register Name:</b> EVOAMCH: Event 0 Address Match Register					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 04028	<b>Offset End:</b> 0402B
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31:0	RW	0	ADDRMT	<b>Event 0 Address Match Register:</b> Address match. Specifies the match value for the address. This effectively maps this event detector to a specific master, channel, and packet type.	

#### 14.3.13 EVOAMSK: Event 0 Address Mask Register

<b>CSR Register Name:</b> EVOAMSK: Event 0 Address Mask Register					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 0402C	<b>Offset End:</b> 0402F
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31:0	RW	0	ADDRMK	<b>Event 0 Address Mask Register:</b> Address mask. Specifies the mask value for the address. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set	

#### 14.3.14 EVODMCH: Event 0 Data Match Register

CSR Register Name: EVODMCH: Event 0 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04030
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMT	<b>Event 0 Data Match Register:</b> Data match. Specifies the match value for the data.
31:0	RW	0	DATAMSK	<b>Event 0 Data Mask Register:</b> Data Mask. Specifies the mask value for the data. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.15 EVOCTRL: Event 0 Match/Mask Control Register

CSR Register Name: EVOCTRL: Event 0 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04038
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	EVOCTRL Reserved
15:8	RO	0	RSVD2	EVOCTRL Reserved
7:6	RW	0	BAR_MCH	<b>BAR Match bits.</b> Specifies the match value for the BAR (Base Address Register). 00: BAR 0, CSR_MTB_BAR 01: BAR 1, STMR BAR. 10: BAR 2, RTIT BAR 11: BAR 3, Firmware BAR (FW_BAR)
5:4	RW	0	BAR_MSK	<b>BAR Mask bits.</b> Specifies the mask value for the BAR (Base Address Register). 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match.
3	RW	0	INVMATCH	<b>Invert Match:</b> When set, makes the trigger match function either NAND or NOR, depending upon the state of bit 2. 0: The match function type is as specified by bit 2. 1: The match function type is inverted from the specification of bit 2.
2	RW	0	ANDMATCH	<b>AND Match:</b> Sets the type of the trigger match function. 0: The match function is OR. When any bit enabled by its mask bit matches the value of its match bit the Match output is asserted. 1: The match function is AND. All bits enabled by their mask bits must match the value of their match bits for the Match output to be asserted.

CSR Register Name: EVOCTRL: Event 0 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04038
Bits	Access	Default	Label	Bit Description
1	RO	0	RSVD3	EVOCTRL Reserved
0	RW	0	ENEVTO	<b>Enable Event 0.</b> 0: do not generate Event 0 signal to CTS when Event 0 is detected 1: generate Event 0 signal to CTS when Event 0 is detected.

#### 14.3.16 EV1AMCH: Event 1 Address Match Register

CSR Register Name: EV1AMCH: Event 1 Address Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0403C
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDRMT	<b>Event 1 Address Match Register:</b> Address match. Specifies the match value for the address. This effectively maps this event detector to a specific master, channel, and packet type.

#### 14.3.17 EV1AMSK: Event 1 Address Mask Register

CSR Register Name: EV1AMSK: Event 1 Address Mask Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04040
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDRMK	<b>Event 1 Address Mask Register:</b> Address mask. Specifies the mask value for the address. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.18 EV1DMCH: Event 1 Data Match Register

CSR Register Name: EV1DMCH: Event 1 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04044
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMT	<b>Event 1 Data Match Register:</b> Data match. Specifies the match value for the data.



CSR Register Name: EV1DMCH: Event 1 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04044
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMSK	<b>Event 1 Data Mask Register:</b> Data Mask. Specifies the mask value for the data. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.19 EV1CTRL: Event 1 Match/Mask Control Register

CSR Register Name: EV1CTRL: Event 1 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0404C
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	EV1CTRL Reserved
15:8	RO	0	RSVD2	EV1CTRL Reserved
7:6	RW	0	BAR_MCH	<b>Event 1 Match/Mask Control Register:</b> BAR Match bits. Specifies the match value for the BAR (Base Address Register). 00: BAR 0, CSR_MTB_BAR 01: BAR 1, STMR BAR. 10: BAR 2, RTIT BAR 11: BAR 3, Firmware BAR (FW_BAR)
5:4	RW	0	BAR_MSK	<b>BAR Mask bits.</b> Specifies the mask value for the BAR (Base Address Register). 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match.
3	RW	0	INVMATCH	<b>Invert Match:</b> When set, makes the trigger match function either NAND or NOR, depending upon the state of bit 2. 0: The match function type is as specified by bit 2. 1: The match function type is inverted from the specification of bit 2.
2	RW	0	ANDMATCH	<b>AND Match:</b> Sets the type of the trigger match function. 0: The match function is OR. When any bit enabled by its mask bit matches the value of its match bit the Match output is asserted. 1: The match function is AND. All bits enabled by their mask bits must match the value of their match bits for the Match output to be asserted.
1	RO	0	RSVD3	EV1CTRL Reserved



CSR Register Name: EV1CTRL: Event 1 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0404C
Bits	Access	Default	Label	Bit Description
0	RW	0	ENEVT1	<b>Enable Event 1.</b> 0: do not generate Event 1 signal to CTS when Event 1 is detected 1: generate Event 1 signal to CTS when Event 1 is detected.

#### 14.3.20 EV2AMCH: Event 2 Address Match Register

CSR Register Name: EV2AMCH: Event 2 Address Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04050
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDRMT	<b>Event 2 Address Match Register:</b> Address match. Specifies the match value for the address. This effectively maps this event detector to a specific master, channel, and packet type.

#### 14.3.21 EV2AMSK: Event 2 Address Mask Register

CSR Register Name: EV2AMSK: Event 2 Address Mask Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04054
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDRMK	<b>Event 2 Address Mask Register:</b> Address mask. Specifies the mask value for the address. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.22 EV2DMCH: Event 2 Data Match Register

CSR Register Name: EV2DMCH: Event 2 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04058
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMT	<b>Event 2 Data Match Register:</b> Data match. Specifies the match value for the data.



CSR Register Name: EV2DMCH: Event 2 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04058
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMSK	<b>Event 2 Data Mask Register:</b> Data Mask. Specifies the mask value for the data. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.23 EV2CTRL: Event 2 Match/Mask Control Register

CSR Register Name: EV2CTRL: Event 2 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04060
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	EV2CTRL Reserved
15:8	RO	0	RSVD2	EV2CTRL Reserved
7:6	RW	0	BAR_MCH	<b>Event 2 Match/Mask Control Register:</b> BAR Match bits. Specifies the match value for the BAR (Base Address Register). 00: BAR 0, CSR_MTB_BAR 01: BAR 1, STMR BAR. 10: BAR 2, RTIT BAR 11: BAR 3, Firmware BAR (FW_BAR)
5:4	RW	0	BAR_MSK	<b>BAR Mask bits.</b> Specifies the mask value for the BAR (Base Address Register). 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match.
3	RW	0	INVMATCH	<b>Invert Match:</b> When set, makes the trigger match function either NAND or NOR, depending upon the state of bit 2. 0: The match function type is as specified by bit 2. 1: The match function type is inverted from the specification of bit 2.
2	RW	0	ANDMATCH	<b>AND Match:</b> Sets the type of the trigger match function. 0: The match function is OR. When any bit enabled by its mask bit matches the value of its match bit the Match output is asserted. 1: The match function is AND. All bits enabled by their mask bits must match the value of their match bits for the Match output to be asserted.
1	RO	0	RSVD3	EV2CTRL Reserved



CSR Register Name: EV2CTRL: Event 2 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04060
Bits	Access	Default	Label	Bit Description
0	RW	0	ENEVT2	<b>Enable Event 2.</b> 0: do not generate Event 2 signal to CTS when Event 2 is detected 1: generate Event 2 signal to CTS when Event 2 is detected.

#### 14.3.24 EV3AMCH: Event 3 Address Match Register

CSR Register Name: EV3AMCH: Event 3 Address Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04064
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDRMT	<b>Event 3 Address Match Register:</b> Address match. Specifies the match value for the address. This effectively maps this event detector to a specific master, channel, and packet type.

#### 14.3.25 EV3AMSK: Event 3 Address Mask Register

CSR Register Name: EV3AMSK: Event 3 Address Mask Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04068
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDRMK	<b>Event 3 Address Mask Register:</b> Address mask. Specifies the mask value for the address. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.26 EV3DMCH: Event 3 Data Match Register

CSR Register Name: EV3DMCH: Event 3 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0406C
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMT	<b>Event 3 Data Match Register:</b> Data match. Specifies the match value for the data.



CSR Register Name: EV3DMCH: Event 3 Data Match Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0406C
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATAMSK	<b>Event 3 Data Mask Register:</b> Data Mask. Specifies the mask value for the data. 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match. Note: To trigger an event at least one bit of this register has to be set

#### 14.3.27 EV3CTRL: Event 3 Match/Mask Control Register

CSR Register Name: EV3CTRL: Event 3 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04074
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	EV3CTRL Reserved
15:8	RO	0	RSVD2	EV3CTRL Reserved
7:6	RW	0	BAR_MCH	<b>Event 3 Match/Mask Control Register:</b> BAR Match bits. Specifies the match value for the BAR (Base Address Register). 00: BAR 0, CSR_MTB_BAR 01: BAR 1, STMR BAR. 10: BAR 2, RTIT BAR 11: BAR 3, Firmware BAR (FW_BAR)
5:4	RW	0	BAR_MSK	<b>BAR Mask bits.</b> Specifies the mask value for the BAR (Base Address Register). 0: Means the corresponding bit is not included in the match. 1: Means the corresponding bit is included in the match.
3	RW	0	INVMATCH	<b>Invert Match:</b> When set, makes the trigger match function either NAND or NOR, depending upon the state of bit 2. 0: The match function type is as specified by bit 2. 1: The match function type is inverted from the specification of bit 2.
2	RW	0	ANDMATCH	<b>AND Match:</b> Sets the type of the trigger match function. 0: The match function is OR. When any bit enabled by its mask bit matches the value of its match bit the Match output is asserted. 1: The match function is AND. All bits enabled by their mask bits must match the value of their match bits for the Match output to be asserted.
1	RO	0	RSVD3	EV3CTRL Reserved

CSR Register Name: EV3CTRL: Event 3 Match/Mask Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04074
Bits	Access	Default	Label	Bit Description
0	RW	0	ENEVT3	<b>Enable Event 3.</b> 0: do not generate Event 3 signal to CTS when Event 3 is detected 1: generate Event 3 signal to CTS when Event 3 is detected.

#### 14.3.28 STHMISC: STH MISC Control register 1

CSR Register Name: STHMISC: STH MISC Control register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 04078
Bits	Access	Default	Label	Bit Description
31:2	RO	0	RSVD	STHMISC Reserved
1	RW	0	BDCDIS	<b>Backpressure Duration Counter Disable.</b> This bit should be set whenever Intel Processor Trace data is being sent to the STH. When set, this bit disables the BDC, and STH will not enter Drop Mode due to BDC expiry. This is required so that STH does not drop any Intel PT data; rather, Intel PT data can only be dropped at/by the CPU.
0	RW	0	RSVD	Reserved

#### 14.3.29 STHCAP2: STH Capability 2

CSR Register Name: STHCAP2: STH Capability 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0407C
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	STHCAP2 Reserved
15:0	RO	10	FW_MSTR_STRT	<b>Firmware Master Start.</b> Specifies the starting Master number for the firmware ACPI BAR If the firmware ACPI BAR is not present, this field will read all zeros.



## 14.4 TSCU Registers

### 14.4.1 TSCU Registers Summary

TSCU Registers			
Offset Start	Offset End	Symbol	Register Name/Function
2000	2003	TSUCTRL	TSCU Control Register
2004	2007	TSCUSTAT	TSCU Status Register
2010	2013	NPKTSCSSL	NPKTSC Snapshot Lower Register
2014	2017	NPKTSCSSU	NPKTSC Snapshot Upper Register

### 14.4.2 TSUCTRL: TSCU Control Register

CSR Register Name: TSUCTRL: TSCU Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 2000
Bits	Access	Default	Label	Bit Description
31:8	RO	0	Reserved	Reserved
7:4	RW	1	MaxAllowedDelay	<b>MaxAllowedDelay.</b> Specifies the maximum allowed round trip time of the non-posted LocalStamp message when the sync command is bounded. The agent can choose one of three bound values pre-defined by the corresponding ARU register. The agent must choose a value which is longer than the minimum round trip delay of a non-posted message from the ARU to the target agent. Encoding: 0000 => Bound Range Low 0001 => Bound Range 2 0010 => Bound Range Max 0011-1111 => Reserved
3	RW	0	Rsvd	Reserved
2	RW/AC	0	SnapShotTSC	<b>Snapshot Timestamp Counter.</b> Writing a 1 causes the current value of both CTC to be written to the snapshot registers so that it can be safely read by software. This bit is automatically cleared one North Peak clock after being written to remove the need for a separate write to clear it before getting subsequent snapshot updates. Writing zero has no effect. Reads will always return 0 for this field.
1	RO	0	Rsvd	Reserved
0	RW/V	1	CTCResync	<b>CTC Resync.</b> When set, forces the CTC control state machine to go to the INIT state. When cleared, allows the CTC control state machine to begin the CTC initialization process if the ctc_enable signal is high and if the XCLK is running.

### 14.4.3 TSCUSTAT: TSCU Status Register

CSR Register Name: TSCUSTAT: TSCU Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 2004	Offset End: 2007
Bits	Access	Default	Label	Bit Description	
31:13	RO	0	Reserved	Reserved	
12:11	RO	0	LTA_WIDTH	LTA Fast count width :  00: 7bits 01: 8bits 10: 6bits	
10:8	RO	0	CTCCSMstat	<b>CTC Control State Machine Status.</b> The current state of the CTC control state machine. 000: INIT 001: IRDY 010: WAIT 011: ARMED 100: SYNC 101: ERROR 110: N/A 111: N/A	
7	RO	0	XCLKstat	<b>XCLK Status.</b> Indicates the status of the XCLK based on the number of NPK clocks since the last XCLK toggle change assertion and the value of the NPK offset counter. 0: XCLK is inactive. 1: XCLK is active.	
6	RW/1C	0	NPKTSCOvrFlw	<b>NPKTSC Overflow.</b> This bit indicates that an overflow of the entire NPKTSC has occurred. 0: no overflow detected. 1: an overflow has occurred.	
5	RW/1C	0	CTCOvrFlw	<b>CTC Overflow.</b> Indicates that an overflow of the entire CTC has occurred. 0: no overflow detected. 1: an overflow has occurred.	
4:3	RO	0	Rsvd2	Reserved	
2	RO/V	0	CTCErrorStat	<b>CTC Error Status.</b> Indicates the current state of the CTCvalid signal. 0: CTCerror signal is not asserted. 1: CTCerror signal is asserted indicating an error occurred while performing CTC initialization.	
1	RO/V	0	CTCSyncingStat	<b>CTC Syncing Status.</b> Indicates the current state of the CTCsyncing signal. 0: CTCsyncing is not asserted. CTC initialization is not in progress. 1: CTCsyncing is asserted. CTC initialization is in progress.	
0	RO/V	0	CTCValidStat	<b>CTC Valid Status.</b> Indicates the current state of the CTCvalid signal. 0: CTC is not in sync with the ART. 1: CTC initialization was successful and CTC is in sync with the ART.	

#### 14.4.4 NPKTSCSSL: NPKTSC Snapshot Lower Register

CSR Register Name: NPKTSCSSL: NPKTSC Snapshot Lower Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b	Offset Start: 2010	Offset End: 2013
Bits	Access	Default	Label	Bit Description
31:0	RW	0	NPKTSCSnapS hotL	<b>NPKTSC Snapshot Lower.</b> Lower DW of the NPKTSC snapshot value

#### 14.4.5 NPKTSCSSU: NPKTSC Snapshot Upper Register

CSR Register Name: NPKTSCSSU: NPKTSC Snapshot Upper Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b	Offset Start: 2014	Offset End: 2017
Bits	Access	Default	Label	Bit Description
31:0	RW	0	NPKTSCSnapS hotU	<b>NPKTSC Snapshot Upper.</b> Upper DW of the NPKTSC snapshot value

## 14.5 MSU Registers

### 14.5.1 MSU Registers Summary

MSU Registers			
Offset Start	Offset End	Symbol	Register Name/Function
A0000	A0003	MSUPARAMS	MSU Parameter Register
A0004	A0007	MINTCTL	MSU Interrupt Control Register
A0008	A000B	MSUSTATUS	MSU Status Register
A0010	A0013	MSUORDT	MSU OBM Residual Data Timer Register
A00FC	A00FF	MSUSPARE	MSU Spare Register
A0100	A0103	MSCOCTL	MSCn Control Register
A0104	A0107	MSCCOSTS	MSCn Status Register
A0108	A010B	MSCOBAR	MSCn Base Address Register
A010C	A010F	MSCODESTSZ	MSCn Memory Buffer Size
A0110	A0113	MSCOMWP	MSC0 Lower Memory Write Pointer Register
A0114	A0117	MSCOTBRP	MSC0 Trace Buffer Read Pointer
A0118	A011B	MSCOTBWP	MSC0 Trace Buffer Write Pointer
A011C	A011F	MSCONWSA	MSC0 Next Window Starting Address
A0120	A0123	MSCCOMMCR	MSCn MMIO Mode Control Register
A0128	A012B	MSCOOEWM	MSCn OBM Entry Watermark
A012C	A012F	MSCOOXWM	MSCn OBM Exit Watermark
A0130	A0133	MSCOFL	MSCn Fill Level
A0134	A0137	MSCOUBAR	MSC0 Upper Base Address Register
A0138	A013B	MSCOUMWP	MSC0 Upper Memory Write Pointer
A013C	A013F	MSCONWUSA	MSC0 Next Window Upper Starting Address
A01FC	A01FF	MSCOSPARE	MSCn Spare Register
A0200	A0203	MSC1CTL	MSCn Control Register
A0204	A0207	MSC1STS	MSCn Status Register
A0208	A020B	MSC1BAR	MSCn Base Address Register
A020C	A020F	MSC1DESTSZ	MSCn Memory Buffer Size
A0210	A0213	MSC1MWP	MSC1 Lower Memory Write Pointer Register
A0214	A0217	MSC1TBRP	MSCn Trace Buffer Read Pointer
A0218	A021B	MSC1TBWP	MSCn Trace Buffer Write Pointer
A021C	A021F	MSC1NWSA	MSCn Next Window Starting Address



MSU Registers			
Offset Start	Offset End	Symbol	Register Name/Function
A0220	A0223	MSC1MMCR	MSCn MMI 1 Mode Control Register
A0228	A022B	MSC1OEWM	MSCn OBM Entry Watermark
A022C	A022F	MSC1OXWM	MSCn OBM Exit Watermark
A0230	A0233	MSC1FL	MSCn Fill Level
A0234	A0237	MSC1UBAR	MSC1 Upper Base Address Register
A0238	A023B	MSC1UMWP	MSC1 Upper Memory Write Pointer
A023C	A023F	MSC1NWUSA	MSC0 Next Window Upper Starting Address
A02FC	A02FF	MSC1SPARE	MSCn Spare Register

#### 14.5.2 MSUPARAMS: MSU Parameter Register

CSR Register Name: MSUPARAMS: MSU Parameter Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0000
Bits	Access	Default	Label	Bit Description
31:19	RO	0	RSVD	MSUPARAMS Reserved
18:16	RO	2	MTBWID	<b>MTBWIDTH</b> is the MSC Trace Buffer width parameter for the two MTBs. This is also equivalent to the IOSF primary interface data bus width.
15:0	RO	100	MTBDEP	<b>MSC Trace Buffer Depth</b> . MTBDEP is the MSC Trace Buffer Depth Parameter for the two MTBs (both are of identical size and depth).

#### 14.5.3 MINTCTL: MSU Interrupt Control Register

CSR Register Name: MINTCTL: MSU Interrupt Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0004
Bits	Access	Default	Label	Bit Description
26	RW	0	M1OXIE	<b>MSC1 OBM Exit Interrupt Enable</b> : Setting this bit will enable issuing an interrupt when MSC1 detects an OBM exit event.
25	RW	0	M1OEIE	<b>MSC1 OBM Entry Interrupt Enable</b> : Setting this bit will enable issuing an interrupt when MSC1 detects an OBM entry event.
24	RW	0	M1BLIE	<b>MSC1 Block Last Interrupt Enable</b> : Setting this bit will enable issuing an interrupt when the Block Last event is detected by MSC1.
18	RW	0	MOOXIE	<b>MSC0 OBM Exit Interrupt Enable</b> : Setting this bit will enable issuing an interrupt when MSC0 detects an OBM exit event.



CSR Register Name: MINTCTL: MSU Interrupt Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0004
Bits	Access	Default	Label	Bit Description
17	RW	0	MOOEIE	<b>MSC0 OBM Entry Interrupt Enable:</b> Setting this bit will enable issuing an interrupt when MSC0 detects an OBM entry event.
16	RW	0	MOBLIE	<b>MSC0 Block Last Interrupt Enable:</b> Setting this bit will enable issuing an interrupt when the Block Last event is detected by MSC0.
0	RW	0	MICDE	<b>MSU Interrupt Capture Done Enable:</b> Setting this bit will enable issuing an interrupt once MSU capture has completed.

#### 14.5.4 MSUSTATUS: MSU Status Register

CSR Register Name: MSUSTATUS: MSU Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0008
Bits	Access	Default	Label	Bit Description
26	RW/1C	0	MSC1OBMX	<b>MSC1 OBM Exit:</b> This bit will be set once the MTB1 fill level crosses above the OBMExitWM from below.
25	RW/1C	0	MSC1OBME	<b>MSC1 OBM Entry:</b> This bit will be set once the MTB1 fill level crosses below OBMEEntryWM from above.
24	RW/1C	0	MSC1BLAST	<b>MSC1 Block Last:</b> When running in a linked-list mode, this bit will be set as soon as MSC1 fetches the header of the next block in a window and detects that the SW tag bit for Block Last is set. This bit will be set independent of whether an interrupt has been generated or not. This bit has no meaning in any non-linked list operational mode.
18	RW/1C	0	MSCOOBMX	<b>MSC0 OBM Exit:</b> This bit will be set once the MTB0 fill level crosses above the OBMExitWM from below.
17	RW/1C	0	MSCOOBME	<b>MSC0 OBM Entry:</b> This bit will be set once the MTB0 fill level crosses below OBMEEntryWM from above.
16	RW/1C	0	MSCOBLAST	<b>MSC0 Block Last:</b> When running in a linked-list mode, this bit will be set as soon as MSC0 fetches the header of the next block in a window and detects that the SW tag bit for Block Last is set. This bit will be set independent of whether an interrupt has been generated or not. This bit has no meaning in any non-linked list operational mode.
0	RW/1C	0	MSU_INT	<b>MSU Interrupt Capture Done output status.</b> Provides the user with readable status indicating that the capture has completed. Once set this status will be held high true until reset or both MSCnEN configuration bits are low.



#### 14.5.5 MSUORDT: MSU OBM Residual Data Timer Register

CSR Register Name: MSUORDT: MSU OBM Residual Data Timer Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0010	Offset End: A0013
Bits	Access	Default	Label	Bit Description	
31:0	RW	FF	MSUORDT	When CaptureDone is asserted to MSU in OBM modes, each MSC will wait OBMRDT clock cycles for system memory path to open before trying to access system memory.	

#### 14.5.6 MSUSPARE: MSU Spare Register

CSR Register Name: MSUSPARE: MSU Spare Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A00FC	Offset End: A00FF
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	RSVD	MSUSPARE Reserved	

#### 14.5.7 MSCOCTL: MSCn Control Register

CSR Register Name: MSCOCTL: MSCn Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0100	Offset End: A0103
Bits	Access	Default	Label	Bit Description	
31:13	RO	0	RSVDO	MSCOCTL Reserved	
12	RO	0	RSVD	Reserved	
11	RO	0	RSVD	MSCnCTL Reserved	
10:8	RW	2	MSC_LEN	<b>MSCn outbound write burst length.</b> The MSC by default will send only a full cache line (64 bytes) when writing to memory. MSCn_LEN is represented as the number of data cycles that is transferred. MSCn_LEN is used for both single buffer and multi block mode. If the primary bus width is 64 bits, the reset value for MSCn_LEN = 3b011, for a transfer length of $8 * 8$ bytes per transfer = 64 bytes total. If the primary bus width is 128 bits, the reset value for MSCn_LEN = 3b010, for a transfer length of $4 * 16$ bytes per transfer = 64 bytes total. This outbound posted write burst length can be programmed to any supported value by the software for flexibility. Here is the encoding for MSCn_LEN: 3b000 : Length = 1 3b001 : Length = 2 3b010 : Length = 4 default if MD_WIDTH = 128 3b011 : Length = 8 default if MD_WIDTH = 64 3b100 : Length = 16	



CSR Register Name: MSCOCTL: MSCn Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0100	Offset End: A0103
Bits	Access	Default	Label	Bit Description	
7:4	RW	0	MSC_MODE	<b>MSCn Mode Control</b> 0x0: single block or CSR-driven mode. 0x1: multi-block or linked-list mode 0x2: DCI Trace Handler Mode 0x3: Internal Buffer Mode. 0x4: opportunistic single-block mode 0x5: opportunistic linked-list mode 0x6: MMIO Mode others: reserved See also the note in the MSU chapter regarding reading of data when switching from a normal mode to Internal Buffer mode.	
3	RO	0	RSVD2	MSCOCTL Reserved	
2	RW	0	MSC_RD_HDR_OVRD	<b>MSCn Read Header Override.</b> In multi window mode, before the MSU begins writing data to the current memory block it performs a SW header read. The read response for this can take some time due to latency in the fabric. For performance optimization, the MSU presumpitively starts to write trace data to this block without knowing the Block Size which is contained in the SW header. Since all blocks are a minimum of 4 Kbytes, this is a fairly safe thing to do as long as it stops at this 4 Kbyte boundary if it hasn't yet received the SW header read response. The MSU must receive the SW header read response that contains the Block Size set to a value above 4Kbytes before it can proceed past this 4 Kbyte boundary. When RD_HDR_OVRD = 0, each MSC will operate as described above, which is considered the normal operating mode. When RD_HDR_OVRD = 1, each MSC will wait for the SW header read response containing this Block Size before it writes any data to this block. This will reduce the performance of the MSC operation.	
1	RW	0	MSC_WRAPEN	<b>Memory Wrap Enabled, MSCn.</b> 0: no wrap 1: wrap enabled Note: The usage of the memory wrap enable is dependent upon the operational mode of the MSC. In single block mode wrapping within the single memory block is enabled and/or disabled with this CSR. In multi-block mode wrapping is always enabled, so WRAPENn is not used. In DCI Trace Handler mode this CSR is unused, as the MSU will wrap within its trace buffer, but will not be writing the trace data to memory. In Internal Buffer mode wrapping within the trace buffer is enabled and/or disabled with this CSR.	
0	RW	0	MSC_EN	<b>MSCn Enable.</b> Enables MSCn operation when set.	



#### 14.5.8 MSCOSTS: MSCn Status Register

CSR Register Name: MSCOSTS: MSCn Status Register				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: A0104	Offset End: A0107
Bits	Access	Default	Label	Bit Description
31:14	RO	0	RSVD	Reserved
13	RO	1	MSCOTBE	MSC0 Trace Buffer Empty
12	RO	0	MSCOTBF	MSC0 Trace Buffer Full
11:9	RO	0	RSVD1	Reserved
8	RO	0	RDPEND	Read Pending: This bit will get set once the MSC issues a read to system memory to fetch the block header and cleared once the MSU receives the completion or an error
7:4	RO/V	0	MSC_STATE	Reserved for future use
3	RW/1C/V	0	ERROR_STATUS	<b>MSU Outbound Error Status.</b> If the MSU receives an error type response on an outbound read, then it will set this error status bit and halt its operation.
2	RO/V	0	MSC_PLE	<b>MSCn PipeLine Empty.</b> When set, indicates that there is no data in the pipeline intended for MSCn or in the MSC Trace Buffer for this MSCn. When software is moving to the retrieval stage, it can check this bit to ensure no data is pending before it begins processing the data.
1	RO/V	0	WRAPSTATO	<b>Memory Wrap Status, MSCn.</b> 0: memory wrap did not occur. That is, the write pointer never wrapped around to zero. Trace data should be reconstructed starting from the start of the memory buffer or trace buffer. 1: memory wrap occurred. Note: This status is dependent upon the operational mode of the MSC. Refer to the table in the Operational Details section for more information.
0	RO	0	RSVDO	MSCOSTS Reserved



#### 14.5.9 MSCOBAR: MSCn Base Address Register

CSR Register Name: MSCOBAR: MSCn Base Address Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0108	Offset End: A010B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	MSCBASE	<b>MSCO Base Address.</b> Specifies the starting address of the trace buffer to which the MSCn will store its data. The value programmed in MSCnBASE is shifted left by 12 bits to create the physical address [43:12]. Due to this shift, the starting address is restricted to start on a 4-kbyte boundary. Note: MSCnBASE is used in both single block mode and in multi block mode to point to the starting address of memory. In multi block mode, this MSCn Base Address will point to the first block in memory and the MSCn will interpret the remaining block and window addresses from the header information in memory.	

#### 14.5.10 MSCODESTSZ: MSCn Memory Buffer Size

CSR Register Name: MSCODESTSZ: MSCn Memory Buffer Size					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A010C	Offset End: A010F
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	MSCSIZE	<b>Size of memory buffer for MSCO.</b> Specifies the size of the system memory buffer for MSCn in terms of 4kB pages bits 43:12. By shifting the value in MSCnSIZE[31:0] left 12 bits, the result is MSCnSIZE multiplied by 4 kBytes. A value of 0h is undefined and should not be used. Only applicable when operating in CSR-driven mode to indicate the memory buffer size of the single memory block.	

#### 14.5.11 MSCOMWP: MSCO Lower Memory Write Pointer Register

CSR Register Name: MSCOMWP: MSCO Lower Memory Write Pointer Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0110	Offset End: A0113
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	MSCOLMWP	MSCO Lower Memory Write Pointer Register	

#### 14.5.12 MSCOTBRP: MSCO Trace Buffer Read Pointer

CSR Register Name: MSCOTBRP: MSCO Trace Buffer Read Pointer					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0114	Offset End: A0117
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	MSC_TBRP	<b>MSCn Trace Buffer Read Pointer.</b> Gives the current value of the Trace Buffer n read pointer for MSCn.	



#### 14.5.13 MSCOTBWP: MSCO Trace Buffer Write Pointer

CSR Register Name: MSCOTBWP: MSCO Trace Buffer Write Pointer				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0118
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSC_TBWP	<b>MSCn Trace Buffer Write Pointer.</b> Gives the current value of the Trace Buffer 0 write pointer for MSCn.

#### 14.5.14 MSCONWSA: MSCO Next Window Starting Address

CSR Register Name: MSCONWSA: MSCO Next Window Starting Address				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A011C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSC_NWSA	<b>MSCn Next Window Starting Address.</b> Gives the current value of the Next Window Starting Address as programmed by the software into the current blocks software header. Note: The value of this register is preserved only while MSCnEN = 1. If MSCnEN is cleared to zero, the value of NWSA will be erased.

#### 14.5.15 MSCOMMCR: MSCn MMIO Mode Control Register

CSR Register Name: MSCOMMCR: MSCn MMIO Mode Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0120
Bits	Access	Default	Label	Bit Description
31:12	RO	0	Reserved	Reserved
11:3	RW	0	ADDR	<b>Specifies the lower 9 bits of address for MMIO Mode operation.</b> The value of MSCnBAR is shifted left by 12, then these bits are concatenated to arrive at the complete address 63:3 value. Note: This register is only used in MMIO mode and is not used otherwise
2:0	RO	0	RSVD	Reserved

#### 14.5.16 MSCOOEWM: MSCn OBM Entry Watermark

CSR Register Name: MSCOOEWM: MSCn OBM Entry Watermark				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0128
Bits	Access	Default	Label	Bit Description
31:0	RW	0	MSCOOEWM	<b>MSCO OBM Entry Watermark.</b> Once the trace data fill level is less than or equal to this watermark value (MSCnFL <= MSCnOEWM), the MSC will suspend all its system memory traffic until an exit condition is detected



#### 14.5.17 MSCOOXWM: MSCn OBM Exit Watermark

CSR Register Name: MSCOOXWM: MSCn OBM Exit Watermark				
Bar: CSR_MTB_BAR		Reset: npk_rst_b	Offset Start: A012C	Offset End: A012F
Bits	Access	Default	Label	Bit Description
31:0	RW	FFFFFFFF	MSCOOXWM	<b>MSCO OBM Exit Watermark.</b> Once trace data fill level is larger than this watermark value (MSCnFL > MSCnOXWM), the MSC will initiate traffic to system memory independent of its state

#### 14.5.18 MSCOFL: MSCn Fill Level

CSR Register Name: MSCOFL: MSCn Fill Level				
Bar: CSR_MTB_BAR		Reset: npk_rst_b	Offset Start: A0130	Offset End: A0133
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSCOFL	<b>MSCO Fill Level.</b> The number of valid rows in the MTB.

#### 14.5.19 MSCOUBAR: MSCO Upper Base Address Register

CSR Register Name: MSCOUBAR: MSCO Upper Base Address Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b	Offset Start: A0134	Offset End: A0137
Bits	Access	Default	Label	Bit Description
31:20	RO	0	Reserved	Reserved
19:0	RW	0	UADDR	<b>Upper Address.</b> Specifies the upper 20 bits (bits 63:44) of the starting address of the trace buffer to which the MSCn will store its data. The value of this register is used in single-block modes and MMIO mode. It is not used in any other mode  If wrapping is enabled, and WRAPSTAT0 is high (1), then the trace buffer wrapped. In this case, the beginning of the stored trace is at MSCnUMWP + MSCnLMWP and end of the stored trace is at (MSCnUMWP + MSCnLMWP ) -1. If wrapping is not enabled, or WRAPSTAT0 is low, then the start of the trace buffer is at MSCnBAR and the end is at (MSCnUMWP + MSCnLMWP ) -1. MSCnMWP is not useful for multi-block mode. The information required to reconstruct the trace is stored into the memory headers. MSCnUMWP is loaded with the contents of the upper 32 bits of (MSCnUBAR + MSCnLBAR) << 12 when MSCnEN = 0. For this reason, at the end of the trace capture the contents of MSCnUMWP should be read prior to clearing the MSCnEN if this information is required.



#### 14.5.20 MSCOUMWP: MSCO Upper Memory Write Pointer

CSR Register Name: MSCOUMWP: MSCO Upper Memory Write Pointer				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0138
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSCOUMWP	<b>MSCO Upper Memory Write Pointer.</b> The upper 32-bits of the current value of the memory write pointer for MSCn.

#### 14.5.21 MSCONWUSA: MSCO Next Window Upper Starting Address

CSR Register Name: MSCONWUSA: MSCO Next Window Upper Starting Address				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A013C
Bits	Access	Default	Label	Bit Description
31:20	RO	0	Reserved	Reserved for future use
19:0	RO	0	MSCnNWUSA	<b>MSCO Next Window Starting Address.</b> Gives the current value of the upper 20 bits of the Next Window Starting Address as programmed by the software into the current block's software header. Note: This is applicable for multi-block mode only. Note: The value of this register is preserved only while MSCnEN = 1. If MSCnEN is cleared to zero, the value of NWSA will be erased.

#### 14.5.22 MSCOSPARE: MSCn Spare Register

CSR Register Name: MSCOSPARE: MSCn Spare Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A01FC
Bits	Access	Default	Label	Bit Description
31:0	RO	0	RSVD	MSCOSPARE Reserved

#### 14.5.23 MSC1CTL: MSCn Control Register

CSR Register Name: MSC1CTL: MSCn Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0200
Bits	Access	Default	Label	Bit Description
31:13	RO	0	RSVD	MSC1CTL Reserved
12	RO	0	RSVD	Reserved
11	RO	0	RSVDO	MSCnCTL Reserved

CSR Register Name: MSC1CTL: MSCn Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0200	Offset End: A0203
Bits	Access	Default	Label	Bit Description	
10:8	RW	2	MSC_LEN	<b>MSCn outbound write burst length.</b> The MSC by default will send only a full cache line (64 bytes) when writing to memory. MSCn_LEN is represented as the number of data cycles that is transferred. MSCn_LEN is used for both single buffer and multi block mode. If the primary bus width is 64 bits, the reset value for MSCn_LEN = 3b011, for a transfer length of 8 * 8 bytes per transfer = 64 bytes total. If the primary bus width is 128 bits, the reset value for MSCn_LEN = 3b010, for a transfer length of 4 * 16 bytes per transfer = 64 bytes total. This outbound posted write burst length can be programmed to any supported value by the software for flexibility. Here is the encoding for MSCn_LEN: 3b000 : Length = 1 3b001 : Length = 2 3b010 : Length = 4 default if MD_WIDTH = 128 3b011 : Length = 8 default if MD_WIDTH = 64 3b100 : Length = 16	
7:4	RW	0	MSC_MODE	<b>MSCn Mode Control</b> <b>0x0:</b> single block or CSR-driven mode. <b>0x1:</b> multi-block or linked-list mode <b>0x2:</b> DCI Trace Handler Mode <b>0x3:</b> Internal Buffer Mode. <b>0x4:</b> opportunistic single-block mode <b>0x5:</b> opportunistic linked-list mode <b>0x6:</b> MMIO Mode others: reserved See also the note in the MSU chapter regarding reading of data when switching from a normal mode to Internal Buffer mode.	
3	RO	0	RSVD1	MSC1CTL Reserved	
2	RW	0	MSC_RD_HDR_OVRD	<b>MSCn Read Header Override.</b> In multi window mode, before the MSU begins writing data to the current memory block it performs a SW header read. The read response for this can take some time due to latency in the fabric. For performance optimization, the MSU presumptively starts to write trace data to this block without knowing the Block Size which is contained in the SW header. Since all blocks are a minimum of 4 Kbytes, this is a fairly safe thing to do as long as it stops at this 4 Kbyte boundary if it hasn't yet received the SW header read response. The MSU must receive the SW header read response that contains the Block Size set to a value above 4Kbytes before it can proceed past this 4 Kbyte boundary. When RD_HDR_OVRD = 0, each MSC will operate as described above, which is considered the normal operating mode. When RD_HDR_OVRD = 1, each MSC will wait for the SW header read response containing this Block Size before it writes any data to this block. This will reduce the performance of the MSC operation.	



CSR Register Name: MSC1CTL: MSCn Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0200	Offset End: A0203
Bits	Access	Default	Label	Bit Description	
1	RW	0	MSC_WRAPEN	<b>Memory Wrap Enabled, MSCn.</b> 0: no wrap 1: wrap enabled Note: The usage of the memory wrap enable is dependent upon the operational mode of the MSC. In single block mode wrapping within the single memory block is enabled and/or disabled with this CSR. In multi-block mode wrapping is always enabled, so WRAPEN is not used. In DCI Trace Handler mode this CSR is unused, as the MSU will wrap within its trace buffer, but will not be writing the trace data to memory. In Internal Buffer mode wrapping within the trace buffer is enabled and/or disabled with this CSR.	
0	RW	0	MSC_EN	<b>MSCn Enable.</b> Enables MSCn operation when set.	

#### 14.5.24 MSC1STS: MSCn Status Register

CSR Register Name: MSC1STS: MSCn Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0204	Offset End: A0207
Bits	Access	Default	Label	Bit Description	
31:14	RO	0	RSVD	Reserved	
13	RO	1	MSC1TBE	MSCn Trace Buffer Empty	
12	RO	0	MSC1TBF	MSCn Trace Buffer Full	
11:9	RO	0	RSVD1	Reserved	
8	RO	0	RDPEND	Read Pending: This bit will get set once the MSC issues a read to system memory to fetch the block header and cleared once the MSU receives the completion or an error	
7:4	RO/V	0	MSC_STATE	Reserved for future use	
3	RW/1C/V	0	ERROR_STATUS	<b>MSU Outbound Error Status.</b> If the MSU receives an error type response on an outbound read, then it will set this error status bit and halt its operation.	
2	RO/V	0	MSC_PLE	<b>MSCn PipeLine Empty.</b> When set, indicates that there is no data in the pipeline intended for MSCn or in the MSC Trace Buffer for this MSCn. When software is moving to the retrieval stage, it can check this bit to ensure no data is pending before it begins processing the data.	



CSR Register Name: MSC1STS: MSCn Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0204	Offset End: A0207
Bits	Access	Default	Label	Bit Description	
1	RO/V	0	WRAPSTAT1	<b>Memory Wrap Status, MSCn.</b> 0: memory wrap did not occur. That is, the write pointer never wrapped around to zero. Trace data should be reconstructed starting from the start of the memory buffer or trace buffer. 1: memory wrap occurred. Note: This status is dependent upon the operational mode of the MSC. Refer to the table in the Operational Details section for more information.	
0	RO	0	RSVDO	MSC1STS Reserved	

#### 14.5.25 MSC1BAR: MSCn Base Address Register

CSR Register Name: MSC1BAR: MSCn Base Address Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0208	Offset End: A020B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	MSCBASE	<b>MSCn Base Address.</b> Specifies the starting address of the trace buffer to which the MSCn will store its data. The value programmed in MSCnBASE is shifted left by 12 bits to create the physical address [43:12]. Due to this shift, the starting address is restricted to start on a 4-kbyte boundary. Note: MSCnBASE is used in both single block mode and in multi block mode to point to the starting address of memory. In multi block mode, this MSCn Base Address will point to the first block in memory and the MSCn will interpret the remaining block and window addresses from the header information in memory.	

#### 14.5.26 MSC1DETSZ: MSCn Memory Buffer Size

CSR Register Name: MSC1DETSZ: MSCn Memory Buffer Size					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A020C	Offset End: A020F
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	MSCSIZE	<b>Size of memory buffer for MSCn.</b> Specifies the size of the system memory buffer for MSCn in terms of 4kB pages bits 43:12. By shifting the value in MSCnSIZE[31:0] left 12 bits, the result is MSCnSIZE multiplied by 4 kBytes. A value of 0h is undefined and should not be used. Only applicable when operating in CSR-driven mode to indicate the memory buffer size of the single memory block.	



#### 14.5.27 MSC1MWP: MSC1 Lower Memory Write Pointer Register

CSR Register Name: MSC1MWP: MSC1 Lower Memory Write Pointer Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0210
Bits	Access	Default	Label	Bit Description
31:0	RW	0	MSC1LMWP	MSC1 Lower Memory Write Pointer Register

#### 14.5.28 MSC1TBRP: MSCn Trace Buffer Read Pointer

CSR Register Name: MSC1TBRP: MSCn Trace Buffer Read Pointer				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0214
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSC_TBRP	<b>MSCn Trace Buffer Read Pointer.</b> Gives the current value of the Trace Buffer n read pointer for MSCn.

#### 14.5.29 MSC1TBWP: MSCn Trace Buffer Write Pointer

CSR Register Name: MSC1TBWP: MSCn Trace Buffer Write Pointer				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0218
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSC_TBWP	<b>MSCn Trace Buffer Write Pointer.</b> Gives the current value of the Trace Buffer 0 write pointer for MSCn.

#### 14.5.30 MSC1NWSA: MSCn Next Window Starting Address

CSR Register Name: MSC1NWSA: MSCn Next Window Starting Address				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A021C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	MSC_NWSA	<b>MSCn Next Window Starting Address.</b> Gives the current value of the Next Window Starting Address as programmed by the software into the current blocks software header. Note: The value of this register is preserved only while MSCnEN = 1. If MSCnEN is cleared to zero, the value of NWSA will be erased.



#### 14.5.31 MSC1MMCR: MSCn MMI 1 Mode Control Register

CSR Register Name: MSC1MMCR: MSCn MMI 1 Mode Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0220	Offset End: A0223
Bits	Access	Default	Label	Bit Description	
31:12	RO	0	Reserved	Reserved	
11:3	RW	0	ADDR	<b>MSCn MMIO Address.</b> Specifies the lower 9 bits of address for MMIO Mode operation. The value of MSCnBAR is shifted left by 12, then these bits are concatenated to arrive at the complete address 63:3 value. Note: This register is only used in MMIO mode and is not used otherwise.	
2:0	RO	0	Reserved	Reserved	

#### 14.5.32 MSC1OEWM: MSCn OBM Entry Watermark

CSR Register Name: MSC1OEWM: MSCn OBM Entry Watermark					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0228	Offset End: A022B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	MSC1OEWM	<b>MSC1 OBM Entry Watermark.</b> Once the trace data fill level is less than or equal to this watermark value (MSCnFL <= MSCnOEWM), the MSC will suspend all its system memory traffic until an exit condition is detected	

#### 14.5.33 MSC1OXWM: MSCn OBM Exit Watermark

CSR Register Name: MSC1OXWM: MSCn OBM Exit Watermark					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A022C	Offset End: A022F
Bits	Access	Default	Label	Bit Description	
31:0	RW	FFFFFFFF	MSC1OXWM	<b>MSC1 OBM Exit Watermark.</b> Once trace data fill level is larger than this watermark value (MSCnFL > MSCnOXWM), the MSC will initiate traffic to system memory independent of its state	

#### 14.5.34 MSC1FL: MSCn Fill Level

CSR Register Name: MSC1FL: MSCn Fill Level					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0230	Offset End: A0233
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	MSC1FL	<b>MSC1 Fill Level.</b> The number of valid rows in the MTB for MSC1.	

#### 14.5.35 MSC1UBAR: MSC1 Upper Base Address Register

CSR Register Name: MSC1UBAR: MSC1 Upper Base Address Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0234	Offset End: A0237
Bits	Access	Default	Label	Bit Description	
31:20	RO	0	Reserved	Reserved	
19:0	RW	0	UADDR	<p><b>MSC1 BAR Upper Address.</b> Specifies the upper 20 bits (bits 63:44) of the starting address of the trace buffer to which the MSCn will store its data. The value of this register is used in single-block modes and MMIO mode. It is not used in any other mode.</p> <p>If wrapping is enabled, and WRAPSTAT0 is high (1), then the trace buffer wrapped. In this case, the beginning of the stored trace is at MSCnUMWP + MSCnLMWP and end of the stored trace is at (MSCnUMWP + MSCnLMWP ) -1.</p> <p>If wrapping is not enabled, or WRAPSTAT0 is low, then the start of the trace buffer is at MSCnBAR and the end is at (MSCnUMWP + MSCnLMWP ) -1.</p> <p>MSCnMWP is not useful for linked-list mode. The information required to reconstruct the trace is stored into the memory headers.</p> <p>MSCnUMWP is loaded with the contents of the upper 32 bits of (MSCnUBAR + MSCnLBAR) &lt;&lt; 12 when MSCnEN = 0. For this reason, at the end of the trace capture the contents of MSCnUMWP should be read prior to clearing the MSCnEN if this information is required.</p> <p>Note: This pointer will always contain the address of the next MWr to system memory.</p>	

#### 14.5.36 MSC1UMWP: MSC1 Upper Memory Write Pointer

CSR Register Name: MSC1UMWP: MSC1 Upper Memory Write Pointer					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0238	Offset End: A023B
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	MSC1UMWP	<p><b>MSC1 Upper Memory Write Pointer.</b> The upper 32-bits of the current value of the memory write pointer for MSCn. In single block mode, software will use this information along with the wrap status to interpret the trace data stored in memory. At the end of the capture, MSCnUMWP + MSCnLMWP points to the next location in memory that data would be written to. This means that (MSCnUMWP + MSCnLMWP ) -1 is the last byte of trace data that was successfully written.</p> <p>If wrapping is enabled, and WRAPSTAT0 is high (1), then the trace buffer wrapped. In this case, the beginning of the stored trace is at MSCnUMWP + MSCnLMWP and end of the stored trace is at (MSCnUMWP + MSCnLMWP ) -1.</p> <p>If wrapping is not enabled, or WRAPSTAT0 is low, then the start of the trace buffer is at MSCnBAR and the end is at</p>	



CSR Register Name: MSC1UMWP: MSC1 Upper Memory Write Pointer				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A0238
Bits	Access	Default	Label	Bit Description
				<p>(MSCnUMWP + MSCnLMWP) -1.</p> <p>MSCnMWP is not useful for linked-list mode. The information required to reconstruct the trace is stored into the memory headers.</p> <p>MSCnUMWP is loaded with the contents of the upper 32 bits of (MSCnUBAR + MSCnLBAR) &lt;&lt; 12 when MSCnEN = 0. For this reason, at the end of the trace capture the contents of MSCnUMWP should be read prior to clearing the MSCnEN if this information is required.</p> <p>Note: This pointer will always contain the address of the next MWr to system memory.</p> <p>If wrapping is enabled, and WRAPSTAT0 is high (1), then the trace buffer wrapped. In this case, the beginning of the stored trace is at MSCnUMWP + MSCnLMWP and end of the stored trace is at (MSCnUMWP + MSCnLMWP) -1.</p> <p>If wrapping is not enabled, or WRAPSTAT0 is low, then the start of the trace buffer is at MSCnBAR and the end is at (MSCnUMWP + MSCnLMWP) -1.</p> <p>MSCnMWP is not useful for linked-list mode. The information required to reconstruct the trace is stored into the memory headers.</p> <p>MSCnUMWP is loaded with the contents of the upper 32 bits of (MSCnUBAR + MSCnLBAR) &lt;&lt; 12 when MSCnEN = 0. For this reason, at the end of the trace capture the contents of MSCnUMWP should be read prior to clearing the MSCnEN if this information is required.</p> <p>Note: This pointer will always contain the address of the next MWr to system memory.</p>

#### 14.5.37 MSC1NWUSA: MSCO Next Window Upper Starting Address

CSR Register Name: MSC1NWUSA: MSCO Next Window Upper Starting Address				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A023C
Bits	Access	Default	Label	Bit Description
31:20	RO	0	Reserved	Reserved for future use
19:0	RO	0	MSC1NWUSA	<p><b>MSC1 Next Window Starting Address.</b> Gives the current value of the upper 20 bits of the Next Window Starting Address as programmed by the software into the current block's software header.</p> <p>Note: This is applicable for multi-window mode only.</p> <p>Note: The value of this register is preserved only while MSCnEN = 1. If MSCnEN is cleared to zero, the value of NWSA will be erased.</p>



#### 14.5.38 MSC1SPARE: MSCn Spare Register

CSR Register Name: MSC1SPARE: MSCn Spare Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A02FC	Offset End: A02FF
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	RSVD	MSC1SPARE Reserved	

## 14.6 DCI Trace Handler Registers

### 14.6.1 DCI Trace Handler Registers Summary

DCI Trace Handler Registers			
Offset Start	Offset End	Symbol	Register Name/Function
A1000	A1003	STREAMCFG1	NPKH Streaming Configuration
A1004	A1007	STREAMCFG2	Streaming Configuration 2
A1008	A100B	DBCSTSCMD	DBC.Trace Status Command
A100C	A100F	DBCSTS DATA	DBC Status Data
A1010	A1013	EXISTSCMD	EXI Bridge Status Command
A1014	A1017	EXISTSDATA	Exi Status Data
A1018	A101B	TIMEOUT	NPKH Time out register
A101C	A101F	BRIDGE_BASE_ADDR_HI	EXI Bridge Base Address Hi register
A1020	A1023	BRIDGE_BASE_ADDR_LO	EXI Bridge Base Address Lo register
A1024	A1027	DBC_BASE_ADDR_HI	DBC Bridge Base Address Hi register
A1028	A102B	DBC_BASE_ADDR_LO	DBC Base Address Lo register
A102C	A102F	NPKHSTS	NPKH Status register
A1030	A1033	NPKH_RETRY	NPKH Retry Register
A1034	A1037	RSVD2	NPKH Reserved Register 2

### 14.6.2 STREAMCFG1: NPKH Streaming Configuration

CSR Register Name: STREAMCFG1: NPKH Streaming Configuration				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: A1000	Offset End: A1003	
Bits	Access	Default	Label	Bit Description
31	RW/AC	0	RESET	<b>DCI Trace handler Soft Reset.</b> Writing 1 to this will reset DCI Trace Handler state machine and credits. This bit is self-clearing, and will always read as zero.
30	RO	0	RSVD1	Reserved for future use.
29	RO	0	RSVDO	Reserved for future use.
28	RW	0	ENABLE	<b>DCI Trace Handler Enable.</b> 0: DCI TH is disabled 1: DCI TH is enabled, and will stream data to the destination.
27	RO	0	FLUSHDONE	<b>Flush Done.</b> This bit is set when the flush is completed.
26	RW	0	FLUSH	<b>Flush DCI TH data.</b> Writing 1 to this bit will cause the DCI TH to flush trace data to current streaming destination. This bit is not self-clearing.



CSR Register Name: STREAMCFG1: NPKH Streaming Configuration				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1000
Bits	Access	Default	Label	Bit Description
25:14	RW	0	EXIOFFSET	<b>DCI Bridge Offset.</b> Specifies the offset from the DCI Bridge BAR EXIBASEHI + EXIBASELO that the DCI TH will use when writing data to the DCI Bridge
13:2	RW	0	RSVD	Reserved for future use
1	RO	0	STRMMODE	<b>Current streaming mode.</b> Reflects the current streaming mode. 1: BSSB Mode 0: DBC.Trace Mode.
0	RW	0	SETSTRMMODE	<b>Streaming mode:</b> 0: DBC.Trace mode. 1: BSSB Mode Changing this bit would cause the DCI TH to change streaming mode after all pending transactions in current streaming mode are completed.

#### 14.6.3 STREAMCFG2: Streaming Configuration 2

CSR Register Name: STREAMCFG2: Streaming Configuration 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1004
Bits	Access	Default	Label	Bit Description
31:24	RO	0	RSVD	STREAMCFG2 Reserved
23:12	RW	0	DBCOFFSET	<b>DbC.Trace DMA Request Offset.</b> Specifies the offset from the DbC BAR (DBCBASEHI + DBCBASELO) that the DCI TH will use when sending DMA requests to the DbC.Trace.
11:0	RW	0	EXISTSOFFSET	<b>DCI Status Register Offset.</b> Specifies the offset of the DCI streaming Status register that the DCI Trace handler uses for sending status updates. This register and feature is currently not used.

#### 14.6.4 DBCSTSCMD: DBC.Trace Status Command

CSR Register Name: DBCSTSCMD: DBC.Trace Status Command				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1008
Bits	Access	Default	Label	Bit Description
31:4	RO	0	RSVD	DBCSTSCMD Reserved
3:0	RW	0	CMD	Status return command : 0000: none 0001: Credit return 1111: Error report



#### 14.6.5 DBCSTSDATA: DBC Status Data

CSR Register Name: DBCSTSDATA: DBC Status Data				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A100C
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATA	Data value associated with status cmd.

#### 14.6.6 EXISTSCMD: EXI Bridge Status Command

CSR Register Name: EXISTSCMD: EXI Bridge Status Command				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1010
Bits	Access	Default	Label	Bit Description
31:4	RO	0	RSVD	EXISTSCMD Reserved
3:0	RW	0	CMD	Status return command : 0000: None 0001: Credit Return 1111: Error report others: reserved

#### 14.6.7 EXISTSDATA: Exi Status Data

CSR Register Name: EXISTSDATA: Exi Status Data				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1014
Bits	Access	Default	Label	Bit Description
31:0	RW	0	DATA	Data value associated with status cmd.

#### 14.6.8 TIMEOUT: NPKH Time out register

CSR Register Name: TIMEOUT: NPKH Time out register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1018
Bits	Access	Default	Label	Bit Description
31:0	RW	0	TIMEOUT	DCI Trace Handler Time out register. Time out value for DCI Trace handler 00000000h : Time out disabled all other values : Time out value.



#### 14.6.9 BRIDGE\_BASE\_ADDR\_HI: EXI Bridge Base Address Hi register

CSR Register Name: BRIDGE_BASE_ADDR_HI: EXI Bridge Base Address Hi register				
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: A101C
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDR_HI	<b>DCI Bridge Base Address.</b> Specifies the high-order DWord of the DCI Bridge base address bits 63:32

#### 14.6.10 BRIDGE\_BASE\_ADDR\_LO: EXI Bridge Base Address Lo register

CSR Register Name: BRIDGE_BASE_ADDR_LO: EXI Bridge Base Address Lo register				
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: A1020
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDR_LO	<b>DCI Bridge Base Address.</b> Specifies the low-order DWord of the DCI Bridge base address bits 31:0

#### 14.6.11 DBC\_BASE\_ADDR\_HI: DBC Bridge Base Address Hi register

CSR Register Name: DBC_BASE_ADDR_HI: DBC Bridge Base Address Hi register				
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: A1024
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDR_HI	<b>DbC.Trace Base Address.</b> Specifies the high-order DWord of the DbC.Trace base address bits 63:32

#### 14.6.12 DBC\_BASE\_ADDR\_LO: DBC Base Address Lo register

CSR Register Name: DBC_BASE_ADDR_LO: DBC Base Address Lo register				
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: A1028
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDR_LO	<b>DbC.Trace Base Address.</b> Specifies the low-order DWord of the DbC.Trace base address bits 31:0

#### 14.6.13 NPKHSTS: NPKH Status register

CSR Register Name: NPKHSTS: NPKH Status register				
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: A102C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	TBD	Reserved



#### 14.6.14 NPKH\_RETRY: NPKH Retry Register

CSR Register Name: NPKH_RETRY: NPKH Retry Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: A1030	Offset End: A1033
Bits	Access	Default	Label	Bit Description	
5:0	RW	0	MAXRETRIES	<b>DCI Trace Handler Retry Register.</b> Maximum number of retries. When non-zero, specifies the maximum number of times the DCI Trace Handler will reset itself and retry its data	



## 14.7 CTS Registers

### 14.7.1 CTS Registers Summary

CTS Registers			
Offset Start	Offset End	Symbol	Register Name/Function
03000	03003	CLAUSE_EVENT_MODE_C0S0	Clause Event Mode Register C0S0
03004	03007	CLAUSE_EVENT_MODE_C1S0	Clause Event Mode Register C1S0
03008	0300B	CLAUSE_EVENT_MODE_C2S0	Clause Event Mode Register C2S0
0300C	0300F	CLAUSE_EVENT_MODE_C3S0	Clause Event Mode Register C3S0
03018	0301B	CLAUSE_EVENT_MODE_C0S1	Clause Event Mode Register C0S1
0301C	0301F	CLAUSE_EVENT_MODE_C1S1	Clause Event Mode Register C1S1
03020	03023	CLAUSE_EVENT_MODE_C2S1	Clause Event Mode Register C2S1
03024	03027	CLAUSE_EVENT_MODE_C3S1	Clause Event Mode Register C3S1
03030	03033	CLAUSE_EVENT_MODE_C0S2	Clause Event Mode Register C0S2
03034	03037	CLAUSE_EVENT_MODE_C1S2	Clause Event Mode Register C1S2
03038	0303B	CLAUSE_EVENT_MODE_C2S2	Clause Event Mode Register C2S2
0303C	0303F	CLAUSE_EVENT_MODE_C3S2	Clause Event Mode Register C3S2
03048	0304B	CLAUSE_EVENT_MODE_C0S3	Clause Event Mode Register C0S3
0304C	0304F	CLAUSE_EVENT_MODE_C1S3	Clause Event Mode Register C1S3
03050	03053	CLAUSE_EVENT_MODE_C2S3	Clause Event Mode Register C2S3
03054	03057	CLAUSE_EVENT_MODE_C3S3	Clause Event Mode Register C3S3
03060	03063	CLAUSE_EVENT_MODE_C0S4	Clause Event Mode Register C0S4
03064	03067	CLAUSE_EVENT_MODE_C1S4	Clause Event Mode Register C1S4
03068	0306B	CLAUSE_EVENT_MODE_C2S4	Clause Event Mode Register C2S4



CTS Registers			
Offset Start	Offset End	Symbol	Register Name/Function
0306C	0306F	CLAUSE_EVENT_MODE_C3S4	Clause Event Mode Register C3S4
030C0	030C3	CLAUSE_EVENT_ENABLE_C0S0	Clause Event Enable Register C0S0
030C4	030C7	CLAUSE_EVENT_ENABLE_C1S0	Clause Event Enable Register C1S0
030C8	030CB	CLAUSE_EVENT_ENABLE_C2S0	Clause Event Enable Register C2S0
030CC	030CF	CLAUSE_EVENT_ENABLE_C3S0	Clause Event Enable Register C3S0
030D8	030DB	CLAUSE_EVENT_ENABLE_C0S1	Clause Event Enable Register C0S1
030DC	030DF	CLAUSE_EVENT_ENABLE_C1S1	Clause Event Enable Register C1S1
030E0	030E3	CLAUSE_EVENT_ENABLE_C2S1	Clause Event Enable Register C2S1
030E4	030E7	CLAUSE_EVENT_ENABLE_C3S1	Clause Event Enable Register C3S1
030F0	030F3	CLAUSE_EVENT_ENABLE_C0S2	Clause Event Enable Register C0S2
030F4	030F7	CLAUSE_EVENT_ENABLE_C1S2	Clause Event Enable Register C1S2
030F8	030FB	CLAUSE_EVENT_ENABLE_C2S2	Clause Event Enable Register C2S2
030FC	030FF	CLAUSE_EVENT_ENABLE_C3S2	Clause Event Enable Register C3S2
03108	0310B	CLAUSE_EVENT_ENABLE_C0S3	Clause Event Enable Register C0S3
0310C	0310F	CLAUSE_EVENT_ENABLE_C1S3	Clause Event Enable Register C1S3
03110	03113	CLAUSE_EVENT_ENABLE_C2S3	Clause Event Enable Register C2S3
03114	03117	CLAUSE_EVENT_ENABLE_C3S3	Clause Event Enable Register C3S3
03120	03123	CLAUSE_EVENT_ENABLE_C0S4	Clause Event Enable Register C0S4
03124	03127	CLAUSE_EVENT_ENABLE_C1S4	Clause Event Enable Register C1S4
03128	0312B	CLAUSE_EVENT_ENABLE_C2S4	Clause Event Enable Register C2S4



CTS Registers			
Offset Start	Offset End	Symbol	Register Name/Function
0312C	0312F	CLAUSE_EVENT_ENABLE_C3S4	Clause Event Enable Register C3S4
03180	03183	CLAUSE_ACTION_CONTROL_C0S0	Clause Action Control Registers C0S0
03180	03183	CLAUSE_ACTION_CONTROL_C2S2	Clause Action Control Registers C2S2
03184	03187	CLAUSE_ACTION_CONTROL_C1S0	Clause Action Control Registers C1S0
03184	03187	CLAUSE_ACTION_CONTROL_C1S2	Clause Action Control Registers C1S2
03188	0318B	CLAUSE_ACTION_CONTROL_C2S0	Clause Action Control Registers C2S0
0318C	0318F	CLAUSE_ACTION_CONTROL_C3S0	Clause Action Control Registers C3S0
03198	0319B	CLAUSE_ACTION_CONTROL_C0S1	Clause Action Control Registers C0S1
0319C	0319F	CLAUSE_ACTION_CONTROL_C1S1	Clause Action Control Registers C1S1
031A0	031A3	CLAUSE_ACTION_CONTROL_C2S1	Clause Action Control Registers C2S1
031A4	031A7	CLAUSE_ACTION_CONTROL_C3S1	Clause Action Control Registers C3S1
031B0	031B3	CLAUSE_ACTION_CONTROL_C0S2	Clause Action Control Registers C0S2
031B4	031B7	CLAUSE_ACTION_CONTROL_C1S2	Clause Action Control Registers C1S2
031B8	031BB	CLAUSE_ACTION_CONTROL_C2S2	Clause Action Control Registers C2S2
031BC	031BF	CLAUSE_ACTION_CONTROL_C3S2	Clause Action Control Registers C3S2
031C8	031CB	CLAUSE_ACTION_CONTROL_C0S3	Clause Action Control Registers C0S3
031CC	031CF	CLAUSE_ACTION_CONTROL_C1S3	Clause Action Control Registers C1S3
031D0	031D3	CLAUSE_ACTION_CONTROL_C2S3	Clause Action Control Registers C2S3
031D4	031D7	CLAUSE_ACTION_CONTROL_C3S3	Clause Action Control Registers C3S3
031E0	031E3	CLAUSE_ACTION_CONTROL_C0S4	Clause Action Control Registers C0S4



CTS Registers			
Offset Start	Offset End	Symbol	Register Name/Function
031E4	031E7	CLAUSE_ACTION_CONTROL_C1S4	Clause Action Control Registers C1S4
031E8	031EB	CLAUSE_ACTION_CONTROL_C2S4	Clause Action Control Registers C2S4
031EC	031EF	CLAUSE_ACTION_CONTROL_C3S4	Clause Action Control Registers C3S4
03240	03243	SCT0_CONTROL	SCT0 Control Register
03244	03247	SCT1_CONTROL	SCT1 Control Register
03248	0324B	SCT2_CONTROL	SCT2 Control Register
03250	03253	LCT0_LOWER_CONTROL	LCT0 Lower Control Register
03254	03257	LCT0_UPPER_CONTROL	LCT0 Upper Control Register
03258	0325B	LCT1_LOWER_CONTROL	LCT1 Lower Control Register
0325C	0325F	LCT1_UPPER_CONTROL	LCT1 Upper Control Register
03270	03273	SCT0_CURRENT_STATUS	SCT0 Current Status Register
03274	03277	SCT1_CURRENT_STATUS	SCT1 Current Status Register
03278	0327B	SCT2_CURRENT_STATUS	SCT2 Current Status Register
03280	03283	LCT0_LOWER_CURRENT_STATUS	LCT0 Lower Current Status Register
03284	03287	LCT0_UPPER_CURRENT_STATUS	LCT0 Upper Current Status Register
03288	0328B	LCT1_LOWER_CURRENT_STATUS	LCT1 Lower Current Status Register
0328C	0328F	LCT1_UPPER_CURRENT_STATUS	LCT1 Upper Current Status Register
032A0	032A3	CTS_STATUS	CTS Status Register
032A4	032A7	CTS_CONTROL	CTS Control Register
032A8	032AB	LCT0_PEAK_TIMER_LOWER_STATUS	LCT0 Peak Timer Lower Status Register
032AC	032AF	LCT0_PEAK_TIMER_UPPER_STATUS	LCT0 Peak Timer Upper Status Register
032B0	032B3	LCT1_PEAK_TIMER_LOWER_STATUS	LCT1 Peak Timer Lower Status Register
032B4	032B7	LCT1_PEAK_TIMER_UPPER_STATUS	LCT1 Peak Timer Upper Status Register
032C8	032CB	PARAMETER_STATUS	Parameter Status Register
03300	03303	EXTENDED_CLAUSE_ACTION_CONTROL_CO50	Extended Clause Action Control Registers CO50



CTS Registers			
Offset Start	Offset End	Symbol	Register Name/Function
03304	03307	EXTENDED_CLAUSE_ACTION_CONTROL_C1S0	Extended Clause Action Control Registers C1S0
03308	0330B	EXTENDED_CLAUSE_ACTION_CONTROL_C2S0	Extended Clause Action Control Registers C2S0
0330C	0330F	EXTENDED_CLAUSE_ACTION_CONTROL_C3S0	Extended Clause Action Control Registers C3S0
03318	0331B	EXTENDED_CLAUSE_ACTION_CONTROL_C0S1	Extended Clause Action Control Registers C0S1
0331C	0331F	EXTENDED_CLAUSE_ACTION_CONTROL_C1S1	Extended Clause Action Control Registers C1S1
03320	03323	EXTENDED_CLAUSE_ACTION_CONTROL_C2S1	Extended Clause Action Control Registers C2S1
03324	03327	EXTENDED_CLAUSE_ACTION_CONTROL_C3S1	Extended Clause Action Control Registers C3S1
03330	03333	EXTENDED_CLAUSE_ACTION_CONTROL_C0S2	Extended Clause Action Control Registers C0S2
03334	03337	EXTENDED_CLAUSE_ACTION_CONTROL_C1S2	Extended Clause Action Control Registers C1S2
03338	0333B	EXTENDED_CLAUSE_ACTION_CONTROL_C2S2	Extended Clause Action Control Registers C2S2
0333C	0333F	EXTENDED_CLAUSE_ACTION_CONTROL_C3S2	Extended Clause Action Control Registers C3S2
03348	0334B	EXTENDED_CLAUSE_ACTION_CONTROL_C0S3	Extended Clause Action Control Registers C0S3
0334C	0334F	EXTENDED_CLAUSE_ACTION_CONTROL_C1S3	Extended Clause Action Control Registers C1S3
03350	03353	EXTENDED_CLAUSE_ACTION_CONTROL_C2S3	Extended Clause Action Control Registers C2S3
03354	03357	EXTENDED_CLAUSE_ACTION_CONTROL_C3S3	Extended Clause Action Control Registers C3S3
03360	03363	EXTENDED_CLAUSE_ACTION_CONTROL_C0S4	Extended Clause Action Control Registers C0S4
03364	03367	EXTENDED_CLAUSE_ACTION_CONTROL_C1S4	Extended Clause Action Control Registers C1S4
03368	0336B	EXTENDED_CLAUSE_ACTION_CONTROL_C2S4	Extended Clause Action Control Registers C2S4
0336C	0336F	EXTENDED_CLAUSE_ACTION_CONTROL_C3S4	Extended Clause Action Control Registers C3S4



#### 14.7.2 CLAUSE\_EVENT\_MODE\_COSO: Clause Event Mode Register COSO

CSR Register Name: CLAUSE_EVENT_MODE_COSO: Clause Event Mode Register COSO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 03000	Offset End: 03003
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.3 CLAUSE\_EVENT\_MODE\_C1SO: Clause Event Mode Register C1SO

CSR Register Name: CLAUSE_EVENT_MODE_C1SO: Clause Event Mode Register C1SO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 03004	Offset End: 03007
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources



CSR Register Name: CLAUSE_EVENT_MODE_C1SO: Clause Event Mode Register C1SO				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03004
Bits	Access	Default	Label	Bit Description
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.4 CLAUSE\_EVENT\_MODE\_C2SO: Clause Event Mode Register C2SO

CSR Register Name: CLAUSE_EVENT_MODE_C2SO: Clause Event Mode Register C2SO				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03008
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched

<b>CSR Register Name: CLAUSE_EVENT_MODE_C2SO: Clause Event Mode Register C2SO</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03008</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.5 CLAUSE\_EVENT\_MODE\_C3SO: Clause Event Mode Register C3SO

<b>CSR Register Name: CLAUSE_EVENT_MODE_C3SO: Clause Event Mode Register C3SO</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 0300C</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched

<b>CSR Register Name: CLAUSE_EVENT_MODE_C3SO: Clause Event Mode Register C3SO</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 0300C</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.6 CLAUSE\_EVENT\_MODE\_COS1: Clause Event Mode Register COS1

<b>CSR Register Name: CLAUSE_EVENT_MODE_COS1: Clause Event Mode Register COS1</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03018</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0

CSR Register Name: CLAUSE_EVENT_MODE_COS1: Clause Event Mode Register COS1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03018
Bits	Access	Default	Label	Bit Description
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.7 CLAUSE\_EVENT\_MODE\_C1S1: Clause Event Mode Register C1S1

CSR Register Name: CLAUSE_EVENT_MODE_C1S1: Clause Event Mode Register C1S1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0301C
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0



#### 14.7.8 CLAUSE\_EVENT\_MODE\_C2S1: Clause Event Mode Register C2S1

CSR Register Name: CLAUSE_EVENT_MODE_C2S1: Clause Event Mode Register C2S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03020	Offset End: 03023	
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.9 CLAUSE\_EVENT\_MODE\_C3S1: Clause Event Mode Register C3S1

CSR Register Name: CLAUSE_EVENT_MODE_C3S1: Clause Event Mode Register C3S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03024	Offset End: 03027	
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources



CSR Register Name: CLAUSE_EVENT_MODE_C3S1: Clause Event Mode Register C3S1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03024
Bits	Access	Default	Label	Bit Description
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.10 CLAUSE\_EVENT\_MODE\_COS2: Clause Event Mode Register COS2

CSR Register Name: CLAUSE_EVENT_MODE_COS2: Clause Event Mode Register COS2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03030
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched

<b>CSR Register Name: CLAUSE_EVENT_MODE_COS2: Clause Event Mode Register COS2</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03030</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.11 CLAUSE\_EVENT\_MODE\_C1S2: Clause Event Mode Register C1S2

<b>CSR Register Name: CLAUSE_EVENT_MODE_C1S2: Clause Event Mode Register C1S2</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03034</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched

<b>CSR Register Name: CLAUSE_EVENT_MODE_C1S2: Clause Event Mode Register C1S2</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03034</b>	<b>Offset End: 03037</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0	
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0	

#### 14.7.12 CLAUSE\_EVENT\_MODE\_C2S2: Clause Event Mode Register C2S2

<b>CSR Register Name: CLAUSE_EVENT_MODE_C2S2: Clause Event Mode Register C2S2</b>					
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03038</b>	<b>Offset End: 0303B</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources	
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0	



CSR Register Name: CLAUSE_EVENT_MODE_C2S2: Clause Event Mode Register C2S2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03038
Bits	Access	Default	Label	Bit Description
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.13 CLAUSE\_EVENT\_MODE\_C3S2: Clause Event Mode Register C3S2

CSR Register Name: CLAUSE_EVENT_MODE_C3S2: Clause Event Mode Register C3S2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0303C
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0



#### 14.7.14 CLAUSE\_EVENT\_MODE\_COS3: Clause Event Mode Register COS3

CSR Register Name: CLAUSE_EVENT_MODE_COS3: Clause Event Mode Register COS3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03048	Offset End: 0304B	
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.15 CLAUSE\_EVENT\_MODE\_C1S3: Clause Event Mode Register C1S3

CSR Register Name: CLAUSE_EVENT_MODE_C1S3: Clause Event Mode Register C1S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 0304C	Offset End: 0304F	
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources



CSR Register Name: CLAUSE_EVENT_MODE_C1S3: Clause Event Mode Register C1S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0304C	Offset End: 0304F
Bits	Access	Default	Label	Bit Description	
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0	
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0	

#### 14.7.16 CLAUSE\_EVENT\_MODE\_C2S3: Clause Event Mode Register C2S3

CSR Register Name: CLAUSE_EVENT_MODE_C2S3: Clause Event Mode Register C2S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03050	Offset End: 03053
Bits	Access	Default	Label	Bit Description	
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources	
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched	

<b>CSR Register Name: CLAUSE_EVENT_MODE_C2S3: Clause Event Mode Register C2S3</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03050</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.17 CLAUSE\_EVENT\_MODE\_C3S3: Clause Event Mode Register C3S3

<b>CSR Register Name: CLAUSE_EVENT_MODE_C3S3: Clause Event Mode Register C3S3</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03054</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched

<b>CSR Register Name: CLAUSE_EVENT_MODE_C3S3: Clause Event Mode Register C3S3</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03054</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.18 CLAUSE\_EVENT\_MODE\_COS4: Clause Event Mode Register COS4

<b>CSR Register Name: CLAUSE_EVENT_MODE_COS4: Clause Event Mode Register COS4</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03060</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0



CSR Register Name: CLAUSE_EVENT_MODE_COS4: Clause Event Mode Register COS4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03060
Bits	Access	Default	Label	Bit Description
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.19 CLAUSE\_EVENT\_MODE\_C1S4: Clause Event Mode Register C1S4

CSR Register Name: CLAUSE_EVENT_MODE_C1S4: Clause Event Mode Register C1S4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03064
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0



## 14.7.20 CLAUSE\_EVENT\_MODE\_C2S4: Clause Event Mode Register C2S4

CSR Register Name: CLAUSE_EVENT_MODE_C2S4: Clause Event Mode Register C2S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 03068	Offset End: 0306B
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0

#### 14.7.21 CLAUSE\_EVENT\_MODE\_C3S4: Clause Event Mode Register C3S4

CSR Register Name: CLAUSE_EVENT_MODE_C3S4: Clause Event Mode Register C3S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 0306C	Offset End: 0306F	
Bits	Access	Default	Label	Bit Description
28	RW	0	OP_TYPE	Boolean Operation Type control for this clause 1: OR together all enabled match sources 0: AND together all enabled match sources
25	RW	0	LCT1_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
24	RW	0	LCT0_MATCH_MODE	Match Mode control for the Large 45-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
22	RW	0	SCT2_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 2 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
21	RW	0	SCT1_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 1 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
20	RW	0	SCT0_MATCH_MODE	Match Mode control for the Small 20-bit Counter/Timer 0 for this clause 1: match on counter/timer match 0: match on counter/timer not matched
19:4	RW	0	EVENT_IN_MODE	Mode control for each Event_In[15:0] input for this clause 1: match on a logical 1 0: match on a logical 0
3:0	RW	0	SIGNAL_IN_MODE	Mode control for each Signal_In[3:0] input for this clause 1: match on a logical 1 0: match on a logical 0



### 14.7.22 CLAUSE\_EVENT\_ENABLE\_COSO: Clause Event Enable Register COSO

CSR Register Name: CLAUSE_EVENT_ENABLE_COSO: Clause Event Enable Register COSO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030C0	Offset End: 030C3	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.23 CLAUSE\_EVENT\_ENABLE\_C1SO: Clause Event Enable Register C1SO

CSR Register Name: CLAUSE_EVENT_ENABLE_C1SO: Clause Event Enable Register C1SO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030C4	Offset End: 030C7	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



#### 14.7.24 CLAUSE\_EVENT\_ENABLE\_C2SO: Clause Event Enable Register C2SO

CSR Register Name: CLAUSE_EVENT_ENABLE_C2SO: Clause Event Enable Register C2SO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030C8	Offset End: 030CB	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.25 CLAUSE\_EVENT\_ENABLE\_C3SO: Clause Event Enable Register C3SO

CSR Register Name: CLAUSE_EVENT_ENABLE_C3SO: Clause Event Enable Register C3SO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 030CC	Offset End: 030CF
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



#### 14.7.26 CLAUSE\_EVENT\_ENABLE\_COS1: Clause Event Enable Register COS1

CSR Register Name: CLAUSE_EVENT_ENABLE_COS1: Clause Event Enable Register COS1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030D8	Offset End: 030DB	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.27 CLAUSE\_EVENT\_ENABLE\_C1S1: Clause Event Enable Register C1S1

CSR Register Name: CLAUSE_EVENT_ENABLE_C1S1: Clause Event Enable Register C1S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030DC	Offset End: 030DF	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.28 CLAUSE\_EVENT\_ENABLE\_C2S1: Clause Event Enable Register C2S1

CSR Register Name: CLAUSE_EVENT_ENABLE_C2S1: Clause Event Enable Register C2S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030E0	Offset End: 030E3	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



#### 14.7.29 CLAUSE\_EVENT\_ENABLE\_C3S1: Clause Event Enable Register C3S1

CSR Register Name: CLAUSE_EVENT_ENABLE_C3S1: Clause Event Enable Register C3S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030E4	Offset End: 030E7	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.30 CLAUSE\_EVENT\_ENABLE\_COS2: Clause Event Enable Register COS2

CSR Register Name: CLAUSE_EVENT_ENABLE_COS2: Clause Event Enable Register COS2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 030F0	Offset End: 030F3
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.31 CLAUSE\_EVENT\_ENABLE\_C1S2: Clause Event Enable Register C1S2

CSR Register Name: CLAUSE_EVENT_ENABLE_C1S2: Clause Event Enable Register C1S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 030F4	Offset End: 030F7	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.32 CLAUSE\_EVENT\_ENABLE\_C2S2: Clause Event Enable Register C2S2

CSR Register Name: CLAUSE_EVENT_ENABLE_C2S2: Clause Event Enable Register C2S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 030F8	Offset End: 030FB
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.33 CLAUSE\_EVENT\_ENABLE\_C3S2: Clause Event Enable Register C3S2

CSR Register Name: CLAUSE_EVENT_ENABLE_C3S2: Clause Event Enable Register C3S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 030FC	Offset End: 030FF
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



#### 14.7.34 CLAUSE\_EVENT\_ENABLE\_COS3: Clause Event Enable Register COS3

CSR Register Name: CLAUSE_EVENT_ENABLE_COS3: Clause Event Enable Register COS3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03108	Offset End: 0310B	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.35 CLAUSE\_EVENT\_ENABLE\_C1S3: Clause Event Enable Register C1S3

CSR Register Name: CLAUSE_EVENT_ENABLE_C1S3: Clause Event Enable Register C1S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 0310C	Offset End: 0310F	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.36 CLAUSE\_EVENT\_ENABLE\_C2S3: Clause Event Enable Register C2S3

CSR Register Name: CLAUSE_EVENT_ENABLE_C2S3: Clause Event Enable Register C2S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03110	Offset End: 03113	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.37 CLAUSE\_EVENT\_ENABLE\_C3S3: Clause Event Enable Register C3S3

CSR Register Name: CLAUSE_EVENT_ENABLE_C3S3: Clause Event Enable Register C3S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03114	Offset End: 03117	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



### 14.7.38 CLAUSE\_EVENT\_ENABLE\_COS4: Clause Event Enable Register COS4

CSR Register Name: CLAUSE_EVENT_ENABLE_COS4: Clause Event Enable Register COS4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03120	Offset End: 03123	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable

### 14.7.39 CLAUSE\_EVENT\_ENABLE\_C1S4: Clause Event Enable Register C1S4

CSR Register Name: CLAUSE_EVENT_ENABLE_C1S4: Clause Event Enable Register C1S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03124	Offset End: 03127	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



#### 14.7.40 CLAUSE\_EVENT\_ENABLE\_C2S4: Clause Event Enable Register C2S4

CSR Register Name: CLAUSE_EVENT_ENABLE_C2S4: Clause Event Enable Register C2S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03128	Offset End: 0312B	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 2 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable

#### 14.7.41 CLAUSE\_EVENT\_ENABLE\_C3S4: Clause Event Enable Register C3S4

CSR Register Name: CLAUSE_EVENT_ENABLE_C3S4: Clause Event Enable Register C3S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 0312C	Offset End: 0312F	
Bits	Access	Default	Label	Bit Description
31	RW	0	IF_ANYTHING	<b>When the 'If Anything' clause is enabled for a clause it allows the user to set any actions independent of the event inputs.</b> Setting the 'If Anything' Clause Event Enable bit will force a hit on this event enable. This guarantees an active event to a clause. The 'If Anything' clause is independent of the OP_Type for the clause. The Clause Event Mode for the 'If Anything' event is not required and not implemented.
25	RW	0	LCT1_SELECT	Selects the Large 45-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
24	RW	0	LCT0_SELECT	Selects the Large 45-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
22	RW	0	SCT2_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
21	RW	0	SCT1_SELECT	Selects the Small 20-bit Counter/Timer 1 as match event for this clause 1: enable; 0: disable
20	RW	0	SCT0_SELECT	Selects the Small 20-bit Counter/Timer 0 as match event for this clause 1: enable; 0: disable
19:4	RW	0	EVENT_IN_SELECT	Selects each Event_In[15:0] as match events for this clause 1: enable; 0: disable
3:0	RW	0	SIGNAL_IN_SELECT	Selects each Signal_In[3:0] as match events for this clause 1: enable; 0: disable



#### 14.7.42 CLAUSE\_ACTION\_CONTROL\_COSO: Clause Action Control Registers COSO

CSR Register Name: CLAUSE_ACTION_CONTROL_COSO: Clause Action Control Registers COSO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 03180	Offset End: 03183
Bits	Access	Default	Label	Bit Description
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
11	RW	0	LCT0_START_I NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter



CSR Register Name: CLAUSE_ACTION_CONTROL_COSO: Clause Action Control Registers COSO					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03180	Offset End: 03183
Bits	Access	Default	Label	Bit Description	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.43 CLAUSE\_ACTION\_CONTROL\_C2S2: Clause Action Control Registers C2S2

CSR Register Name: CLAUSE_ACTION_CONTROL_C2S2: Clause Action Control Registers C2S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03180	Offset End: 03183
Bits	Access	Default	Label	Bit Description	
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.	



#### 14.7.44 CLAUSE\_ACTION\_CONTROL\_C1S0: Clause Action Control Registers C1S0

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S0: Clause Action Control Registers C1S0				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03184	Offset End: 03187	
Bits	Access	Default	Label	Bit Description
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.



#### 14.7.45 CLAUSE\_ACTION\_CONTROL\_C1S2: Clause Action Control Registers C1S2

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S2: Clause Action Control Registers C1S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 03184	Offset End: 03187
Bits	Access	Default	Label	Bit Description
9	RW	0	SCT2_START_I NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter

#### 14.7.46 CLAUSE\_ACTION\_CONTROL\_C2S0: Clause Action Control Registers C2S0

CSR Register Name: CLAUSE_ACTION_CONTROL_C2S0: Clause Action Control Registers C2S0				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 03188	Offset End: 0318B
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.



CSR Register Name: CLAUSE_ACTION_CONTROL_C2SO: Clause Action Control Registers C2SO				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 03188	Offset End: 0318B	
Bits	Access	Default	Label	Bit Description
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter



CSR Register Name: CLAUSE_ACTION_CONTROL_C2SO: Clause Action Control Registers C2SO				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03188
Bits	Access	Default	Label	Bit Description
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.



#### 14.7.47 CLAUSE\_ACTION\_CONTROL\_C3S0: Clause Action Control Registers C3S0

CSR Register Name: CLAUSE_ACTION_CONTROL_C3S0: Clause Action Control Registers C3S0				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 0318C	Offset End: 0318F	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C3SO: Clause Action Control Registers C3SO					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 0318C	Offset End: 0318F
Bits	Access	Default	Label	Bit Description	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C3SO: Clause Action Control Registers C3SO					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0318C	Offset End: 0318F
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.48 CLAUSE\_ACTION\_CONTROL\_COS1: Clause Action Control Registers COS1

CSR Register Name: CLAUSE_ACTION_CONTROL_COS1: Clause Action Control Registers COS1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03198	Offset End: 0319B
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



<b>CSR Register Name: CLAUSE_ACTION_CONTROL_COS1: Clause Action Control Registers COS1</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03198</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter



CSR Register Name: CLAUSE_ACTION_CONTROL_COS1: Clause Action Control Registers COS1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03198
Bits	Access	Default	Label	Bit Description
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.



#### 14.7.49 CLAUSE\_ACTION\_CONTROL\_C1S1: Clause Action Control Registers C1S1

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S1: Clause Action Control Registers C1S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 0319C	Offset End: 0319F	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S1: Clause Action Control Registers C1S1					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 0319C	Offset End: 0319F
Bits	Access	Default	Label	Bit Description	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S1: Clause Action Control Registers C1S1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0319C	Offset End: 0319F
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.50 CLAUSE\_ACTION\_CONTROL\_C2S1: Clause Action Control Registers C2S1

CSR Register Name: CLAUSE_ACTION_CONTROL_C2S1: Clause Action Control Registers C2S1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031A0	Offset End: 031A3
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S1: Clause Action Control Registers C2S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031A0	Offset End: 031A3	
Bits	Access	Default	Label	Bit Description
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S1: Clause Action Control Registers C2S1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031A0	Offset End: 031A3
Bits	Access	Default	Label	Bit Description	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	



### 14.7.51 CLAUSE\_ACTION\_CONTROL\_C3S1: Clause Action Control Registers C3S1

CSR Register Name: CLAUSE_ACTION_CONTROL_C3S1: Clause Action Control Registers C3S1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031A4	Offset End: 031A7	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S1: Clause Action Control Registers C3S1					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 031A4	Offset End: 031A7
Bits	Access	Default	Label	Bit Description	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S1: Clause Action Control Registers C3S1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031A4	Offset End: 031A7
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.52 CLAUSE\_ACTION\_CONTROL\_COS2: Clause Action Control Registers COS2

CSR Register Name: CLAUSE_ACTION_CONTROL_COS2: Clause Action Control Registers COS2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B0	Offset End: 031B3
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



CSR Register Name: CLAUSE_ACTION_CONTROL_COS2: Clause Action Control Registers COS2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B0	Offset End: 031B3
Bits	Access	Default	Label	Bit Description	
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.	
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	



CSR Register Name: CLAUSE_ACTION_CONTROL_COS2: Clause Action Control Registers COS2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B0	Offset End: 031B3
Bits	Access	Default	Label	Bit Description	
11	RW	0	LCT0_START_I NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	



### 14.7.53 CLAUSE\_ACTION\_CONTROL\_C1S2: Clause Action Control Registers C1S2

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S2: Clause Action Control Registers C1S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031B4	Offset End: 031B7	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S2: Clause Action Control Registers C1S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B4	Offset End: 031B7
Bits	Access	Default	Label	Bit Description	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S2: Clause Action Control Registers C1S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B4	Offset End: 031B7
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.54 CLAUSE\_ACTION\_CONTROL\_C2S2: Clause Action Control Registers C2S2

CSR Register Name: CLAUSE_ACTION_CONTROL_C2S2: Clause Action Control Registers C2S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B8	Offset End: 031BB
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S2: Clause Action Control Registers C2S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B8	Offset End: 031BB
Bits	Access	Default	Label	Bit Description	
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
11	RW	0	LCT0_START_I NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S2: Clause Action Control Registers C2S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031B8	Offset End: 031BB
Bits	Access	Default	Label	Bit Description	
8	RW	0	SCT1_START_I NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAG E	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORA GE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OU T	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_ OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	



### 14.7.55 CLAUSE\_ACTION\_CONTROL\_C3S2: Clause Action Control Registers C3S2

CSR Register Name: CLAUSE_ACTION_CONTROL_C3S2: Clause Action Control Registers C3S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031BC	Offset End: 031BF	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S2: Clause Action Control Registers C3S2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031BC	Offset End: 031BF	
Bits	Access	Default	Label	Bit Description
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S2: Clause Action Control Registers C3S2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031BC	Offset End: 031BF
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.56 CLAUSE\_ACTION\_CONTROL\_COS3: Clause Action Control Registers COS3

CSR Register Name: CLAUSE_ACTION_CONTROL_COS3: Clause Action Control Registers COS3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031C8	Offset End: 031CB
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



CSR Register Name: CLAUSE_ACTION_CONTROL_COS3: Clause Action Control Registers COS3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031C8	Offset End: 031CB
Bits	Access	Default	Label	Bit Description	
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.	
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	



CSR Register Name: CLAUSE_ACTION_CONTROL_COS3: Clause Action Control Registers COS3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031C8
Bits	Access	Default	Label	Bit Description
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.



### 14.7.57 CLAUSE\_ACTION\_CONTROL\_C1S3: Clause Action Control Registers C1S3

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S3: Clause Action Control Registers C1S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031CC	Offset End: 031CF	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S3: Clause Action Control Registers C1S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031CC	Offset End: 031CF	
Bits	Access	Default	Label	Bit Description
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S3: Clause Action Control Registers C1S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031CC	Offset End: 031CF
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.58 CLAUSE\_ACTION\_CONTROL\_C2S3: Clause Action Control Registers C2S3

CSR Register Name: CLAUSE_ACTION_CONTROL_C2S3: Clause Action Control Registers C2S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031D0	Offset End: 031D3
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S3: Clause Action Control Registers C2S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031D0	Offset End: 031D3
Bits	Access	Default	Label	Bit Description	
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.	
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S3: Clause Action Control Registers C2S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031D0	Offset End: 031D3
Bits	Access	Default	Label	Bit Description	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	



### 14.7.59 CLAUSE\_ACTION\_CONTROL\_C3S3: Clause Action Control Registers C3S3

CSR Register Name: CLAUSE_ACTION_CONTROL_C3S3: Clause Action Control Registers C3S3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031D4	Offset End: 031D7	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S3: Clause Action Control Registers C3S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031D4	Offset End: 031D7
Bits	Access	Default	Label	Bit Description	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S3: Clause Action Control Registers C3S3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031D4	Offset End: 031D7
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.60 CLAUSE\_ACTION\_CONTROL\_COS4: Clause Action Control Registers COS4

CSR Register Name: CLAUSE_ACTION_CONTROL_COS4: Clause Action Control Registers COS4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E0	Offset End: 031E3
Bits	Access	Default	Label	Bit Description	
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11.</b> In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.	
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.	
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	



CSR Register Name: CLAUSE_ACTION_CONTROL_COS4: Clause Action Control Registers COS4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E0	Offset End: 031E3
Bits	Access	Default	Label	Bit Description	
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.	
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	



CSR Register Name: CLAUSE_ACTION_CONTROL_COS4: Clause Action Control Registers COS4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E0	Offset End: 031E3
Bits	Access	Default	Label	Bit Description	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	



### 14.7.61 CLAUSE\_ACTION\_CONTROL\_C1S4: Clause Action Control Registers C1S4

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S4: Clause Action Control Registers C1S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031E4	Offset End: 031E7	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.

CSR Register Name: CLAUSE_ACTION_CONTROL_C1S4: Clause Action Control Registers C1S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031E4	Offset End: 031E7	
Bits	Access	Default	Label	Bit Description
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.



CSR Register Name: CLAUSE_ACTION_CONTROL_C1S4: Clause Action Control Registers C1S4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E4
Bits	Access	Default	Label	Bit Description
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.
3:0	RW	0	SET_SIGNAL_OUT	If Signal_Out_Mode[1:0] = 00 pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.

#### 14.7.62 CLAUSE\_ACTION\_CONTROL\_C2S4: Clause Action Control Registers C2S4

CSR Register Name: CLAUSE_ACTION_CONTROL_C2S4: Clause Action Control Registers C2S4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E8
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	Set_Signal_Value is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	Go to state m as result of clause/state event match, where m is the contents of this register. The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	Clear Large 45-bit Counter/Timer1 on clause/state event match. When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S4: Clause Action Control Registers C2S4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E8	Offset End: 031EB
Bits	Access	Default	Label	Bit Description	
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.	
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.	
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.	
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.	
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.	
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	



CSR Register Name: CLAUSE_ACTION_CONTROL_C2S4: Clause Action Control Registers C2S4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031E8	Offset End: 031EB
Bits	Access	Default	Label	Bit Description	
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter	
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.	
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	



### 14.7.63 CLAUSE\_ACTION\_CONTROL\_C3S4: Clause Action Control Registers C3S4

CSR Register Name: CLAUSE_ACTION_CONTROL_C3S4: Clause Action Control Registers C3S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031EC	Offset End: 031EF	
Bits	Access	Default	Label	Bit Description
31	RW	0	SET_SIGNAL_VALUE	<b>Set_Signal_Value</b> is an action qualifier for the Set_Signal_Out[3:0] actions for each clause of each state if Signal_Out_Mode[1:0] is set to 11. In this mode, the driven level of the Signal_Out output wire will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. The Set_Signal_Value is only used in the signal out qualifier mode of operation.
30:27	RW	0	SET_STATE	<b>Go to state m as result of clause/state event match, where m is the contents of this register.</b> The default reset value for each clause will be the IDLE state. The user can set the next state to any state to build a desired sequencer flow.
24	RW	0	LCT1_CLEAR	<b>Clear Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT1_start_inc_inc is needed to restart the timer, but can be set simultaneously with LCT1_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
23	RW	0	LCT0_CLEAR	<b>Clear Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. LCT0_start_inc is needed to restart the timer, but can be set simultaneously with LCT0_clear. When in Counter mode, a 1 on this action will clear the counter. When in Peak Timer mode, a 1 on this action clears the timer, and halts its operation while simultaneously updating the Peak Timer Status register if the LCT exceeds the current contents of the Peak Timer Status register.
21	RW	0	SCT2_CLEAR	<b>Clear Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT2_start_inc is needed to restart the timer, but can be set simultaneously with SCT2_clear. When in Counter mode, a 1 on this action will clear the counter.
20	RW	0	SCT1_CLEAR	<b>Clear Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT1_start_inc is needed to restart the timer, but can be set simultaneously with SCT1_clear. When in Counter mode, a 1 on this action will clear the counter.



CSR Register Name: CLAUSE_ACTION_CONTROL_C3S4: Clause Action Control Registers C3S4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 031EC	Offset End: 031EF	
Bits	Access	Default	Label	Bit Description
19	RW	0	SCT0_CLEAR	<b>Clear Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action clears the timer and halts its operation. SCT0_start_inc is needed to restart the timer, but can be set simultaneously with SCT0_clear. When in Counter mode, a 1 on this action will clear the counter.
16	RW	0	LCT1_STOP	<b>Stop Large 45-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
15	RW	0	LCT0_STOP	<b>Stop Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will stop the timer at its current value. This enables starting and stopping without losing timer value. When in Counter mode, this action is undefined.
12	RW	0	LCT1_START_I_NC	<b>Enable Large 45-action Counter/Timer1 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
11	RW	0	LCT0_START_I_NC	<b>Enable Large 45-bit Counter/Timer0 on clause/state event match.</b> When in Timer or Peak Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
9	RW	0	SCT2_START_I_NC	<b>Enable Small 20-bit Counter/Timer2 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
8	RW	0	SCT1_START_I_NC	<b>Enable Small 20-bit Counter/Timer1 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
7	RW	0	SCT0_START_I_NC	<b>Enable Small 20-bit Counter/Timer0 on clause/state event match.</b> When in Timer mode, a 1 on this action will start the timer; When in Counter mode, a 1 on this action will increment the counter
6	RW	0	STOP_STORAGE	<b>Clear to logical 0 the Stor_Qual0 output.</b> In the Intel Trace Hub, Stor_Qual0 is connected to CaptureDone, so this action will clear CaptureDone. It would be highly unusual for this bit to be set in a trigger sequence in the Intel Trace Hub. Instead, clearing CaptureDone is normally done as part of the initial configuration sequence.
5	RW	0	START_STORAGE	<b>Set to logical 1 the Stor_Qual0 output, indicating to begin or resume trace storage.</b> In Intel TH, Stor_Qual0 is connected to CaptureDone, so this action will set CaptureDone.

CSR Register Name: CLAUSE_ACTION_CONTROL_C3S4: Clause Action Control Registers C3S4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 031EC	Offset End: 031EF
Bits	Access	Default	Label	Bit Description	
4	RW	0	SET_TRIG_OUT	<b>Pulse the Trig_Out output.</b> All counter/ timers will continue to operate until a transition to the IDLE state, which may or may not be coincident with set_trig_out. This allows for continued operation past the Trig_Out event to control things like Stor_Qual or counting of events or time beyond a trigger for example.	
3:0	RW	0	SET_SIGNAL_OUT	<b>If Signal_Out_Mode[1:0] = 00</b> pulsed mode, then this action will pulse the corresponding Signal_Out[3:0] output for one cycle as result of an event match. If Signal_Out_Mode[1:0] = 01 level mode, then this action will change invert the level of the corresponding Signal_Out[3:0] output as a result of an event match. If Signal_Out_Mode[1:0] = 11 signal out qualifier mode, then this action will set the corresponding Signal_Out[3:0] output to the value of Set_Signal_Out action qualifier.	

#### 14.7.64 SCTO\_CONTROL: SCTO Control Register

CSR Register Name: SCTO_CONTROL: SCTO Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03240	Offset End: 03243
Bits	Access	Default	Label	Bit Description	
26:23	RW	0	SCTO_EVENTS_EL	<b>New for CTS 2.0</b> is the dedicated event counter mode of operation for the small counter /timers. This new SCTO_EventSel3:0 field selects which Event_In[15:0] input to count if this SCT is configured in this new mode. F = Event_in15 .. 0: Event_in0 Note that the number of bits in this field will be reduced to the minimum number to support the parameter selected for Event_Inn:0 inputs. If for example 4 Event_In wires is selected, then only 2 EventSel configuration bits will be required. The remaining bit positions will be grounded to 0.	
22	RW	0	SCTO_MODE_1	<b>SCTO_Mode1 is combined with SCTO_Mode0 in bit position 20 to create new SCT modes available for CTS 2.0.</b> If SCTO_Mode1 is set to 0, then the SCT operation is backwards compatible to CTS 1.0 functionality. The mode definitions are now: 11: Reserved 10: dedicated event counter mode new for CTS 2.0 01: backward compatible to CTS 1.0 to operate as a timer 00: backward compatible to CTS 1.0 to operate as a counter	
21	RW	0	SCTO_TIMEBASE	<b>SCTO_Timebase select.</b> If this Counter/Timer is running in timer mode, this bit will configure its time-base. 0: increment continuously at LCLK frequency if enabled 1: increment only if Reference_Pulse input is high.	

CSR Register Name: SCT0_CONTROL: SCT0 Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03240
Bits	Access	Default	Label	Bit Description
20	RW	0	SCT0_MODE_0	<b>SCT0_Mode0 combined with SCT0_Mode1 in bit position 22 which was added for CTS 2.0.</b> The mode definitions are now: 11: Reserved 10: dedicated event counter mode new for CTS 2.0 01: backward compatible to CTS 1.0 to operate as a timer 00: backward compatible to CTS 1.0 to operate as a counter
19:0	RW	0	SCT0_MATCH_VALUE	<b>Small Counter/Timer 0 SCT0 Match Value.</b> The value programmed in this register will be compared against SCT0 every cycle to create a match event to the clause logic.

#### 14.7.65 SCT1\_CONTROL: SCT1 Control Register

CSR Register Name: SCT1_CONTROL: SCT1 Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03244
Bits	Access	Default	Label	Bit Description
26:23	RW	0	SCT1_EVENTS_EL	<b>New for CTS 2.0</b> is the dedicated event counter mode of operation for the small counter /timers. This new SCT1_EventSel3:0 field selects which Event_In[15:0] input to count if this SCT is configured in this new mode. F = Event_in15 .. 0: Event_in0 Note that the number of bits in this field will be reduced to the minimum number to support the parameter selected for Event_Inn:0 inputs. If for example 4 Event_In wires is selected, then only 2 EventSel configuration bits will be required. The remaining bit positions will be grounded to 0.
22	RW	0	SCT1_MODE_1	<b>SCT1_Mode1 is combined with SCT1_Mode0 in bit position 20 to create new SCT modes available for CTS 2.0.</b> If SCT1_Mode1 is set to 0, then the SCT operation is backwards compatible to CTS 1.0 functionality. The mode definitions are now: 11: Reserved 10: dedicated event counter mode new for CTS 2.0 01: backward compatible to CTS 1.0 to operate as a timer 00: backward compatible to CTS 1.0 to operate as a counter
21	RW	0	SCT1_TIMEBASE	<b>SCT1_Timebase select.</b> If this Counter/Timer is running in timer mode, this bit will configure its time-base. 0: increment continuously at LCLK frequency if enabled 1: increment only if Reference_Pulse input is high.



CSR Register Name: SCT1_CONTROL: SCT1 Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03244
Bits	Access	Default	Label	Bit Description
20	RW	0	SCT1_MODE_0	<b>SCT1_Mode0 combined with SCT1_Mode1 in bit position 22 which was added for CTS 2.0.</b> The mode definitions are now: 11: Reserved 10: dedicated event counter mode new for CTS 2.0 01: backward compatible to CTS 1.0 to operate as a timer 00: backward compatible to CTS 1.0 to operate as a counter
19:0	RW	0	SCT1_MATCH_VALUE	<b>Small Counter/Timer 1 SCT1 Match Value.</b> The value programmed in this register will be compared against SCT1 every cycle to create a match event to the clause logic.

#### 14.7.66 SCT2\_CONTROL: SCT2 Control Register

CSR Register Name: SCT2_CONTROL: SCT2 Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03248
Bits	Access	Default	Label	Bit Description
26:23	RW	0	SCT2_EVENTS_EL	<b>New for CTS 2.0</b> is the dedicated event counter mode of operation for the small counter /timers. This new SCT2_EventSel3:0 field selects which Event_In[15:0] input to count if this SCT is configured in this new mode. F = Event_in15 .. 0: Event_in0 Note that the number of bits in this field will be reduced to the minimum number to support the parameter selected for Event_Inn:0 inputs. If for example 4 Event_In wires is selected, then only 2 EventSel configuration bits will be required. The remaining bit positions will be grounded to 0.
22	RW	0	SCT2_MODE_1	<b>SCT2_Mode2 is combined with SCT2_Mode0 in bit position 20 to create new SCT modes available for CTS 2.0.</b> If SCT2_Mode1 is set to 0, then the SCT operation is backwards compatible to CTS 1.0 functionality. The mode definitions are now: 11: Reserved 10: dedicated event counter mode new for CTS 2.0 01: backward compatible to CTS 1.0 to operate as a timer 00: backward compatible to CTS 1.0 to operate as a counter
21	RW	0	SCT2_TIMEBASE	<b>SCT2_Timebase select.</b> If this Counter/Timer is running in timer mode, this bit will configure its time-base. 0: increment continuously at LCLK frequency if enabled 1: increment only if Reference_Pulse input is high.

CSR Register Name: SCT2_CONTROL: SCT2 Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03248
Bits	Access	Default	Label	Bit Description
20	RW	0	SCT2_MODE_0	<b>SCT2_Mode0 combined with SCT2_Mode1 in bit position 22 which was added for CTS 2.0.</b> The mode definitions are now: 11: Reserved 10: dedicated event counter mode new for CTS 2.0 01: backward compatible to CTS 1.0 to operate as a timer 00: backward compatible to CTS 1.0 to operate as a counter
19:0	RW	0	SCT2_MATCH_VALUE	<b>Small Counter/Timer 2 SCT2 Match Value.</b> The value programmed in this register will be compared against SCT2 every cycle to create a match event to the clause logic.

#### 14.7.67 LCT0\_LOWER\_CONTROL: LCT0 Lower Control Register

CSR Register Name: LCT0_LOWER_CONTROL: LCT0 Lower Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03250
Bits	Access	Default	Label	Bit Description
31:0	RW	0	LCT0_MATCH_VALUE	<b>Large Counter/Timer 0 LCT0 Lower Match value.</b> The value programmed in this register will be compared against LCT0 every cycle to create a match event to the clause logic. The contents of this register is combined with LCT0 Upper Control Register contents to create a 45-bit LCT0_match_value

#### 14.7.68 LCT0\_UPPER\_CONTROL: LCT0 Upper Control Register

CSR Register Name: LCT0_UPPER_CONTROL: LCT0 Upper Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03254
Bits	Access	Default	Label	Bit Description
15	RW	0	LCT0_MODE_1	<b>This is a second LCT0_Mode1 bit added for CTS 2.0</b> functionality to be combined with LCT0_Mode0. The LCT0_Mode1:0 is decoded as follows to perform the following functions: 11: LCT operates in Peak Timer mode. This is new functionality for revision 2.0 of the CTS. 10: Reserved. This is new functionality for revision 2.0 of the CTS. 01: LCT operates as a Timer. This is backwards compatible with revision 1.0 of the CTS. 00: LCT operates as a counter. This is backwards compatible with revision 1.0 of the CTS.



CSR Register Name: LCT0_UPPER_CONTROL: LCT0 Upper Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03254
Bits	Access	Default	Label	Bit Description
14	RW	0	LCT0_TIMEBASE	<b>LCT0_Timebase select.</b> If this Counter/Timer is running in timer mode, this bit will configure its timebase. 0: increment continuously at LCLK frequency if enabled 1: increment only if Reference_Pulse input is high.
13	RW	0	LCT0_MODE_0	<b>Mode control for the LCT0.</b> This is now combined with LCT0_Mode1 in bit position 15, which was added for CTS revision 2.0 functionality. See above for a description.

#### 14.7.69 LCT1\_LOWER\_CONTROL: LCT1 Lower Control Register

CSR Register Name: LCT1_LOWER_CONTROL: LCT1 Lower Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03258
Bits	Access	Default	Label	Bit Description
31:0	RW	0	LCT1_MATCH_VALUE	<b>Large Counter/Timer 1 LCT1 Lower Match value.</b> The value programmed in this register will be compared against LCT1 every cycle to create a match event to the clause logic. The contents of this register is combined with LCT1 Upper Control Register contents to create a 45-bit LCT1_match_value

#### 14.7.70 LCT1\_UPPER\_CONTROL: LCT1 Upper Control Register

CSR Register Name: LCT1_UPPER_CONTROL: LCT1 Upper Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0325C
Bits	Access	Default	Label	Bit Description
15	RW	0	LCT1_MODE_1	<b>This is a second LCT1_Mode1 bit added for CTS 2.0 functionality to be combined with LCT1_Mode0</b> The LCT1_Mode1:0 is decoded as follows to perform the following functions: 11: LCT operates in Peak Timer mode. This is new functionality for revision 2.0 of the CTS. 10: Reserved. This is new functionality for revision 2.0 of the CTS. 01: LCT operates as a Timer. This is backwards compatible with revision 1.0 of the CTS. 00: LCT operates as a counter. This is backwards compatible with revision 1.0 of the CTS.



CSR Register Name: LCT1_UPPER_CONTROL: LCT1 Upper Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0325C
Bits	Access	Default	Label	Bit Description
14	RW	0	LCT1_TIMEBASE	<b>LCT1_Timebase select.</b> If this Counter/Timer is running in timer mode, this bit will configure its timebase. 0: increment continuously at LCLK frequency if enabled 1: increment only if Reference_Pulse input is high.
13	RW	0	LCT1_MODE_0	<b>Mode control for the LCT1.</b> This is now combined with LCT1_Mode1 in bit position 15, which was added for CTS revision 2.0 functionality. See above for a description.

#### 14.7.71 SCT0\_CURRENT\_STATUS: SCT0 Current Status Register

CSR Register Name: SCT0_CURRENT_STATUS: SCT0 Current Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03270
Bits	Access	Default	Label	Bit Description
20	RW/V	0	SCT0_RUNNING	The SCT0_Running status register will be high true if the SCT0 is running in timer mode, and is currently active counting due to a SCT0_start_inc action.
19:0	RW/V	0	SCT0_CURRENT_VALUE	<b>Small Counter/Timer 0 SCT0 Current Value.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start.

#### 14.7.72 SCT1\_CURRENT\_STATUS: SCT1 Current Status Register

CSR Register Name: SCT1_CURRENT_STATUS: SCT1 Current Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03274
Bits	Access	Default	Label	Bit Description
20	RW/V	0	SCT1_RUNNING	The SCT1_Running status register will be high true if the SCT1 is running in timer mode, and is currently active counting due to a SCT1_start_inc action.
19:0	RW/V	0	SCT1_CURRENT_VALUE	<b>Small Counter/Timer 1 SCT1 Current Status.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start.



#### 14.7.73 SCT2\_CURRENT\_STATUS: SCT2 Current Status Register

CSR Register Name: SCT2_CURRENT_STATUS: SCT2 Current Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03278	Offset End: 0327B
Bits	Access	Default	Label	Bit Description	
20	RW/V	0	SCT2_RUNNING	The SCT2_Running status register will be high true if the SCT2 is running in timer mode, and is currently active counting due to a SCT2_start_inc action.	
19:0	RW/V	0	SCT2_CURRENT_VALUE	<b>Small Counter/Timer 2 SCT2 Current Status.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start.	

#### 14.7.74 LCT0\_LOWER\_CURRENT\_STATUS: LCT0 Lower Current Status Register

CSR Register Name: LCT0_LOWER_CURRENT_STATUS: LCT0 Lower Current Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03280	Offset End: 03283
Bits	Access	Default	Label	Bit Description	
31:0	RW/V	0	LCT0_CURRENT_VALUE	<b>Large Counter/Timer 0 LCT0 Lower Current Status.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start. The contents of this register is combined with LCT0 Upper Current Status register contents to create a 45-bit LCT0_current_value	

#### 14.7.75 LCT0\_UPPER\_CURRENT\_STATUS: LCT0 Upper Current Status Register

CSR Register Name: LCT0_UPPER_CURRENT_STATUS: LCT0 Upper Current Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03284	Offset End: 03287
Bits	Access	Default	Label	Bit Description	
13	RW/V	0	LCT0_RUNNING	The LCT0_Running status register will be high true if the LCT0 is running in timer mode, and is currently active counting due to a LCT0_start_inc action.	
12:0	RW/V	0	LCT0_CURRENT_VALUE	<b>Large Counter/Timer 0 LCT0 Upper Current Status.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start. The contents of this register is combined with LCT0 Lower Current Status register contents to create a 45-bit LCT0_current_value	



#### 14.7.76 LCT1\_LOWER\_CURRENT\_STATUS: LCT1 Lower Current Status Register

CSR Register Name: LCT1_LOWER_CURRENT_STATUS: LCT1 Lower Current Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03288	Offset End: 0328B
Bits	Access	Default	Label	Bit Description	
31:0	RW/V	0	LCT1_CURRENT_VALUE	<b>Large Counter/Timer 1 LCT1 Lower Current Status.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start. The contents of this register is combined with LCT1 Upper Current Status register contents to create a 45-bit LCT1_current_value	

#### 14.7.77 LCT1\_UPPER\_CURRENT\_STATUS: LCT1 Upper Current Status Register

CSR Register Name: LCT1_UPPER_CURRENT_STATUS: LCT1 Upper Current Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0328C	Offset End: 0328F
Bits	Access	Default	Label	Bit Description	
13	RW/V	0	LCT1_RUNNING	The LCT1_Running status register will be high true if the LCT1 is running in timer mode, and is currently active counting due to a LCT1_start_inc action.	
12:0	RW/V	0	LCT1_CURRENT_VALUE	<b>Large Counter/Timer 1 LCT1 Upper Current Status.</b> This is a shadow status register reflecting the current state of the counter/timer that can be written & forced into the CTS functional register using the Force_State during a re-start. The contents of this register is combined with LCT1 Lower Current Status register contents to create a 45-bit LCT1_current_value	



## 14.7.78 CTS\_STATUS: CTS Status Register

CSR Register Name: CTS_STATUS: CTS Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032A0	Offset End: 032A3
Bits	Access	Default	Label	Bit Description	
22:16	RW/V	0	STORE_QUAL_INITIAL_VALUE	<b>Store_Qual[4:1]</b> output Initial Value. These registers allow control of the Store_Qual outputs upon starting or re-starting the CTS. The Store_Qual[4:1] outputs will take the contents of the respective Store_Qual_Initial_Value[4:1] register upon starting or re-starting the CTS. These registers function as status registers that mirror the state of the Store_Qual[4:1] outputs. This allows for trace storage to be ON when the CTS begins or re-starts operation. These registers are updated with the Store_Qual[4:1] output values when Update_Status is set.	
13	RO	0	CTS_PIPELINE_D	<b>CTS in pipelined mode status.</b> If CTS synthesis has trouble meeting the timing requirements of the target LCLK, then the logic can have a parameter set to pipeline event match within the clause & state logic. This status can be read via S/W to inform the user that any increment action would require an additional cycle before the updated match result would be available to the clause & state logic. This can be hard coded to the parameter setting in the .xml similar to the CTS_Revision.	
12:10	RO	1	CTS_REVISION	<b>Hard coded CTS revision.</b> Start at revision 0. Increment any time a feature change that would impact the S/W occurs. 000: CTS revision 1.0 001: CTS revision 2.0	
9:6	RW/V	0	SIGNAL_OUT_STATUS	<b>Signal_Out[3:0]</b> output status. This status register reflects the state of the signal_out wires. The state of the signal_out wires can be forced by the S/W writing the contents into this register and having the Force_State bit set when the sequencer is enabled, otherwise its updated every cycle with the signal_out values when Update_Status is set.	
5	RW/V	0	STOR_QUAL_INITIAL_VALUE	<b>Stor_Qual0 output Initial Value.</b> This register allows for control of the Stor_Qual0 output upon starting or re-starting the CTS. The Stor_Qual0 output will take the contents of this register upon starting or re-starting the CTS. This register functions as a status register that mirrors the state of the Stor_Qual0 output. This allows for trace storage to be ON when the CTS begins or re-starts operation. This is updated with the Stor_Qual0 value when Update_Status is set.	
4	RW/V	0	SEQUENCER_TRIGGER	<b>Sequencer_Trig is the shadow status register for the internal sticky state bit called sequencer_trig.</b> sequencer_trig is set when the Trig_Out is asserted and held until the sequencer is re-started or until Cold_Reset. Warm_Reset has no effect. This register is updated when Update_Status is set. This register can be written & forced into the sequencer_trig state bit using the Force_State during a re-start.	



CSR Register Name: CTS_STATUS: CTS Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032A0
Bits	Access	Default	Label	Bit Description
3:0	RW/V	8	SEQUENCER_STATE	<b>Sequencer Current State Status.</b> This is a shadow status register that can be written & forced into the CTS functional register using the Force_State during a re-start. This is updated with the sequencer state when Update_Status is set.

#### 14.7.79 CTS\_CONTROL: CTS Control Register

CSR Register Name: CTS_CONTROL: CTS Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032A4
Bits	Access	Default	Label	Bit Description
21:20	RW	0	SIGNAL_OUT_MODE	<b>Signal_Out_Mode[1:0]</b> selects the mode of all the Signal_Out[3:0] wires. The following describes the Signal_Out_Mode[1:0] settings: 11: The Signal_Out output wire level driven will be set or cleared to the value of the Set_Signal_Value action qualifier bit if the corresponding Set_Signal_Out[3:0] action is taken. This effectively creates a signal out qualifier mode for each of the Signal_Out wires regardless of the current state of the Signal_Out[3:0] wire. In any clause the user can set or clear the driven level of individual Signal_Out wires that are selected by the Set_Signal_Out[3:0] action, but no combination of setting and clearing is supported. 10: Reserved for future use. 01: The Signal Out output wire will toggle their logical level when the corresponding Set_Signal_Out[3:0] action is taken. Using bit 0 as an example, if signal_out0 is currently driven to a logical 0, then if Set_signal_out0 action is true, the signal out0 output will transition to a logical 1. It will remain at this value level until the next Set_signal_out0 action is set where the signal_out0 output will transition toggle from a logical 1 to a logical 0 and remain at that new level. 00: The Signal Out output wire will be pulsed to a logical 1 for a single LCLK cycle if the corresponding Set_Signal_Out[3:0] action is taken.
19	RW	0	SIGNAL_IN_E_DGE_DET	<b>Signal_In[3:0]</b> input Edge Detect Enable. If = 1, will select a rising edge detector on the Signal_In[3:0] inputs If = 0, will bypass the edge detector on the Signal_In[3:0] inputs There is a single Signal_In_edge_det configuration bit that controls the operation of all signal_in input signals.
18	RW	0	UPDATE_STATUS	<b>Update_Status</b> enables the shadow status register updates with the functional register. This bit must be set then cleared to read a snapshot of all the status registers. In the cycle before this bit is cleared, all status will be updated.

CSR Register Name: CTS_CONTROL: CTS Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032A4	Offset End: 032A7
Bits	Access	Default	Label	Bit Description	
17	RW	0	FORCE_STATE	<b>Force the state of the sequencer.</b> When set to 1, the CTS will be forced to jam the contents of the RW_V status registers into the functional registers within CTS. These include Stor_Qual state, Sequencer_State[3:0], and all Counter/Timer contents, including the Timer_Running bits	
16:1	RW	0	EVENT_IN_EDGE_DET	<b>Event_In[15:0]</b> input Edge Detect Enable. If = 1, will select a rising edge detector on the Event_In[15:0] inputs. If = 0, will bypass the edge detector on the Event_In[15:0] inputs. There is a separate Event_In_edge_det to control the operation of each Event_In input. This number is parameterized	
0	RW	0	SEQUENCER_ENABLE	The Sequencer_Enable configuration register is ANDed together with the CTS_En input pin to enable normal operation of the sequencer. 1: The state will transition from IDLE to S0 for the normal operating case, and will transition from IDLE to the Sequencer_State[3:0] value if Force_State is set. 0: Disable or hold the sequencer in the IDLE state, clear counters & timers	

#### 14.7.80 LCTO\_PEAK\_TIMER\_LOWER\_STATUS: LCTO Peak Timer Lower Status Register

CSR Register Name: LCTO_PEAK_TIMER_LOWER_STATUS: LCTO Peak Timer Lower Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032A8	Offset End: 032AB
Bits	Access	Default	Label	Bit Description	
31:0	RO	0	LCTO_PEAK_TIMER_VALUE	<b>Large Counter/Timer 0 LCTO Peak Timer Lower Value.</b> This is a status register reflecting the lower 32 bits of the LCTO peak timer value. The contents of this register are combined with LCTO Upper Current Status register contents to create a 45-bit LCTO_current_value. This is a read only register of the peak timer value, and as such does not implement a shadow status register to save gates. It therefore is not affected by update_status, and is not writeable, or jamable.	



#### 14.7.81 LCT0\_PEAK\_TIMER\_UPPER\_STATUS: LCT0 Peak Timer Upper Status Register

CSR Register Name: LCT0_PEAK_TIMER_UPPER_STATUS: LCT0 Peak Timer Upper Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032AC
Bits	Access	Default	Label	Bit Description
12:0	RO	0	LCT0_PEAK_TIMER_VALUE	<b>Large Counter/Timer 0 LCT0 Peak Timer Upper Value.</b> This is a status register reflecting the upper 13 bits of the LCT0 peak timer value. The contents of this register are combined with LCT0 Lower Current Status register contents to create a 45-bit LCT0_current_value. This is a read only register of the peak timer value, and as such does not implement a shadow status register to save gates. It therefore is not affected by update_status, and is not writeable, or jamable.

#### 14.7.82 LCT1\_PEAK\_TIMER\_LOWER\_STATUS: LCT1 Peak Timer Lower Status Register

CSR Register Name: LCT1_PEAK_TIMER_LOWER_STATUS: LCT1 Peak Timer Lower Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032B0
Bits	Access	Default	Label	Bit Description
31:0	RO	0	LCT1_PEAK_TIMER_VALUE	<b>Large Counter/Timer 1 LCT1 Peak Timer Lower Value.</b> This is a status register reflecting the lower 32 bits of the LCT1 peak timer value. The contents of this register are combined with LCT1 Upper Current Status register contents to create a 45-bit LCT1_current_value. This is a read only register of the peak timer value, and as such does not implement a shadow status register to save gates. It therefore is not affected by update_status, and is not writeable, or jamable.



#### 14.7.83 LCT1\_PEAK\_TIMER\_UPPER\_STATUS: LCT1 Peak Timer Upper Status Register

CSR Register Name: LCT1_PEAK_TIMER_UPPER_STATUS: LCT1 Peak Timer Upper Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032B4	Offset End: 032B7
Bits	Access	Default	Label	Bit Description	
12:0	RO	0	LCT1_PEAK_TIMER_VALUE	<b>Large Counter/Timer 1 LCT1 Peak Timer Upper Value.</b> This is a status register reflecting the upper 13 bits of the LCT1 peak timer value. The contents of this register are combined with LCT1 Lower Current Status register contents to create a 45-bit LCT1_current_value. This is a read only register of the peak timer value, and as such does not implement a shadow status register to save gates. It therefore is not affected by update_status, and is not writeable, or jamable.	

#### 14.7.84 PARAMETER\_STATUS: Parameter Status Register

CSR Register Name: PARAMETER_STATUS: Parameter Status Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 032C8	Offset End: 032CB
Bits	Access	Default	Label	Bit Description	
29	RO	1	PIPELINE_LCT	This is a read only register that reflects the parameter compiled by this design to select additional pipelining in the LCTs.	
28	RO	1	PIPELINE_SCT	This is a read only register that reflects the parameter compiled by this design to select additional pipelining in the SCTs.	
27:25	RO	4	NUM_STATES	This is a read only register that reflects the parameter compiled by this design to select the number of states in the state machine in addition to IDLE. The range is from 1 to 8. The value read from this register is the parameter - 1, so the user will read from 0 to 7.	
24:22	RO	3	NUM_CLAUSES	<b>This is a read only register that reflects the parameter compiled by this design to select the number of clauses per state.</b> The range is from 1 to 6. The value read from this register is the parameter - 1, so the user will read from 0 to 7.	
21:18	RO	F	NUM_EVENTS	<b>This is a read only register that reflects the parameter compiled by this design to select the number of Event inputs.</b> The range is from 1 to 16. The value read from this register is the parameter - 1, so the user will read from 0 to 15.	
17:16	RO	3	NUM_SIGNALS	This is a read only register that reflects the parameter compiled by this design to select the number of Signal_In inputs and Signal_Out outputs. The range is from 1 to 4. The value read from this register is the parameter - 1, so the user will read from 0 to 3.	

<b>CSR Register Name: PARAMETER_STATUS: Parameter Status Register</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 032C8</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
15:13	RO	3	NUM_SCTS	This is a read only register that reflects the parameter compiled by this design to select the number of Small Counter Timers. The range is from 0 to 4. The value read from this register is the parameter without subtracting 1 from it.
12:10	RO	2	NUM_LCTS	This is a read only register that reflects the parameter compiled by this design to select the number of Large Counter Timers. The range is from 0 to 4. The value read from this register is the parameter without subtracting 1 from it.
9:4	RO	1F	LCT_SIZE	<b>This is a read only register that reflects the parameter compiled by this design to select the width size of the LCTs.</b> The range is from 1 to 45. The value read from this register is the parameter - 1, so the user will read from 0 to 44.
3	RO	1	PEAK_TIMER_SELECTED	This is a read only register that reflects the parameter compiled by this design to select the Peak Timer capability for LCTs. 0: disabled, and 1: enabled.
2:0	RO	4	NUM_STOR_QUAL	<b>This is a read only register that reflects the parameter compiled by this design to select the number of Stor_Qual outputs.</b> The range is from 1 to 8. The value read from this register is the parameter - 1, so the user will read from 0 to 7.

#### 14.7.85 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_COSO: Extended Clause Action Control Registers COSO

<b>CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_COSO: Extended Clause Action Control Registers COSO</b>				
<b>Bar: CSR_MTB_BAR</b>		<b>Reset: npk_rst_b</b>		<b>Offset Start: 03300</b>
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.86 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C1SO: Extended Clause Action Control Registers C1SO

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C1SO: Extended Clause Action Control Registers C1SO				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03304
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.87 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C2SO: Extended Clause Action Control Registers C2SO

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C2SO: Extended Clause Action Control Registers C2SO				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03308
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.88 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C3SO: Extended Clause Action Control Registers C3SO

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C3SO: Extended Clause Action Control Registers C3SO				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0330C
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.89 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_COS1: Extended Clause Action Control Registers COS1

<b>CSR Register Name:</b> EXTENDED_CLAUSE_ACTION_CONTROL_COS1: Extended Clause Action Control Registers COS1				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 03318
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.90 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C1S1: Extended Clause Action Control Registers C1S1

<b>CSR Register Name:</b> EXTENDED_CLAUSE_ACTION_CONTROL_C1S1: Extended Clause Action Control Registers C1S1				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 0331C
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.91 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C2S1: Extended Clause Action Control Registers C2S1

<b>CSR Register Name:</b> EXTENDED_CLAUSE_ACTION_CONTROL_C2S1: Extended Clause Action Control Registers C2S1				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 03320
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.92 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C3S1: Extended Clause Action Control Registers C3S1

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C3S1: Extended Clause Action Control Registers C3S1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03324
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.93 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_COS2: Extended Clause Action Control Registers COS2

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_COS2: Extended Clause Action Control Registers COS2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03330
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.94 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C1S2: Extended Clause Action Control Registers C1S2

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C1S2: Extended Clause Action Control Registers C1S2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03334
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.95 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C2S2: Extended Clause Action Control Registers C2S2

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C2S2: Extended Clause Action Control Registers C2S2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03338
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.96 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C3S2: Extended Clause Action Control Registers C3S2

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C3S2: Extended Clause Action Control Registers C3S2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0333C
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.97 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_COS3: Extended Clause Action Control Registers COS3

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_COS3: Extended Clause Action Control Registers COS3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03348
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.98 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C1S3: Extended Clause Action Control Registers C1S3

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C1S3: Extended Clause Action Control Registers C1S3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0334C
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.99 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C2S3: Extended Clause Action Control Registers C2S3

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C2S3: Extended Clause Action Control Registers C2S3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03350
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.100 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C3S3: Extended Clause Action Control Registers C3S3

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C3S3: Extended Clause Action Control Registers C3S3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03354
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.101 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_COS4: Extended Clause Action Control Registers COS4

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_COS4: Extended Clause Action Control Registers COS4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03360
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.102 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C1S4: Extended Clause Action Control Registers C1S4

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C1S4: Extended Clause Action Control Registers C1S4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03364
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

#### 14.7.103 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C2S4: Extended Clause Action Control Registers C2S4

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C2S4: Extended Clause Action Control Registers C2S4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 03368
Bits	Access	Default	Label	Bit Description
12:9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4:1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.



#### 14.7.104 EXTENDED\_CLAUSE\_ACTION\_CONTROL\_C3S4: Extended Clause Action Control Registers C3S4

CSR Register Name: EXTENDED_CLAUSE_ACTION_CONTROL_C3S4: Extended Clause Action Control Registers C3S4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0336C
Bits	Access	Default	Label	Bit Description
12: 9	RW	0	STOP_STORAGE	<b>Stop_storage[4:1]</b> will clear to a logical 0 the respective Store_Qual[4:1] output.
4: 1	RW	0	START_STORAGE	<b>Start_storage[4:1]</b> will set to a logical 1 the respective Store_Qual[4:1] output.

## 14.8 PCI Configuration Registers

### 14.8.1 PCI Configuration Registers Summary

PCI Configuration Registers			
Offset Start	Offset End	Symbol	Register Name/Function
0	3	VID	Device ID Register
0	3	VID	Vendor ID Register
4	7	CMD	Command Register
8	b	RID	Class Code Register
8	b	RID	Revision ID Register
c	f	HT	Header Type Register
10	13	MTB_LBAR	MSC Trace Buffer Lower BAR
14	17	MTB_UBAR	MSC Trace Buffer Upper BAR
18	1b	SW_LBAR	Software Lower BAR
1c	1f	SW_UBAR	Software Upper BAR
20	23	RTIT_LBAR	RTIT Lower BAR
24	27	RTIT_UBAR	RTIT Upper BAR
2c	2f	SVID	Subsystem Vendor ID
34	37	CAP	Capabilities Pointer
3C	3F	INTL	Interrupt Line
40	43	MSCID	MSI Capability ID
44	47	MSILMA	MSI Lower Message Address
48	4b	MSIUMA	MSI Upper Message Address
4c	4f	MSIMD	MSI Message Data
70	73	FW_LBAR	Firmware Lower Bar
74	77	FW_UBAR	Firmware Upper Bar
80	83	NPKDSC	NPK Device Specific Control
90	93	RSVD	NPK Device Specific Defeature



### 14.8.2 VID: Device ID Register

CSR Register Name: VID: Device ID Register					
Bar: n/a		Reset: host_rst_b		Offset Start: 0	Offset End: 1
Bits	Access	Default	Label	Bit Description	
15:0	RO	varies	DID	<b>Device ID:</b> The value that uniquely identifies the North Peak from all other PCI devices. Please consult your product-specific documentation for the Device ID value for this instantiation of the Intel Trace Hub.	

### 14.8.3 VID: Vendor ID Register

CSR Register Name: VID: Vendor ID Register					
Bar: n/a		Reset: host_rst_b		Offset Start: 2	Offset End: 3
Bits	Access	Default	Label	Bit Description	
15:0	RO	8086	VID	Vendor ID: 8086 is Intel Vendor Identification code	

### 14.8.4 CMD: Command Register

CSR Register Name: CMD: Command Register					
Bar: n/a		Reset: host_rst_b		Offset Start: 4	Offset End: 7
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DPE	<b>Detected Parity Error:</b> The Intel Trace Hub does not check parity on incoming transactions; this bit will always read as 0.	
30	RW/1C	0	SSE	<b>Signaled System Error:</b> This bit is set when the device has detected an un-correctable error and reported it via SERR. This requires SERR Enable bit to be set in Command register.	
29	RW/1C	0	RMA	<b>Received Master Abort Status:</b> This bit is set when device receives a Completion transaction with Unsupported Request completion status. No error will be reported	
28	RW/1C	0	RTA	<b>Received Target Abort Status:</b> This bit is set when device receives a Completion transaction with Completer Abort completion status. No error will be reported	
27	RW/1C	0	STA	<b>Signaled Target Abort Status:</b> Set by the device when aborting a request that violates the device programming model. When SERR Enable is set, this will result in the Intel Trace Hub signaling an SERR	
26:25	RO	0	Reserved	Reserved	



CSR Register Name: CMD: Command Register					
Bar: n/a		Reset: host_rst_b		Offset Start: 4	Offset End: 7
Bits	Access	Default	Label	Bit Description	
24	RW/1C	0	MDPE	<b>Master Data Parity Error.</b> The Intel® Trace Hub does not check parity on responses to its requests; this bit will always read as 0.	
23:21	RO	0	Reserved	Reserved	
20	RO	1	CLIST	Capabilities List: Indicates the controller contains a capabilities pointer list and the capability pointer register is implemented at offset 0x40 in the configuration space	
19	RO	0	INSTAT	<b>Interrupt Status:</b> Reflects the state of the interrupt pin at the input of the enable/disable circuit. When the interrupt is asserted, and cleared when the interrupt is cleared (independent of the state of Interrupt Disable bit in command register. This bit is only associated with the INTx messages and has no meaning if the device is using MSI)	
18:16	RO	0	Reserved	Reserved	
15:11	RO	0	Reserved	Reserved	
10	RW	0	IntDis	<b>Interrupt Disable:</b> Disables the function to generate INTx interrupt. A value of 0 enables the function to generate INTA interrupts. Note: this bit has no effect on MSI generation.	
9	RO	0	Reserved	Reserved	
8	RW	0	SERREn	<b>System Error Enable:</b> Setting this bit enables the generation of System Error message, when required.	
7	RO	0	Reserved	Reserved	
6	RW	0	Reserved	Reserved	
5:3	RO	0	Reserved	Reserved	
2	RW	0	BME	<b>Bus Master Enable:</b> When set enables the ability to issue Memory or IO requests, including MSI.	
1	RW	0	MEM	<b>Memory Space Enable:</b> When set, Memory Space Decoding is enabled and memory transactions targeting the device are accepted Note: The MSE has to be set to accept any memory transaction on the primary interface targeting any BARs except the FW_BAR.	
0	RO	0	IOSE	Reserved	



#### 14.8.5 RID: Class Code Register

CSR Register Name: RID: Class Code Register				
Bar: n/a		Reset: host_rst_b		Offset Start: 8
Bits	Access	Default	Label	Bit Description
23:0	RO	130000	Class	<b>Class Code.</b> Indicates the class of devices/functionality to which this device belongs.

#### 14.8.6 RID: Revision ID Register

CSR Register Name: RID: Revision ID Register				
Bar: n/a		Reset: host_rst_b		Offset Start: B
Bits	Access	Default	Label	Bit Description
7:0	RO	1	RID	<b>Revision ID:</b> Indicates the device specific revision identifier.

#### 14.8.7 HT: Header Type Register

CSR Register Name: HT: Header Type Register				
Bar: n/a		Reset: host_rst_b		Offset Start: C
Bits	Access	Default	Label	Bit Description
31:24	RO	0	Reserved	Reserved
23	RO	0	MFD	Reserved
22:16	RO	0	HT	Header Type: Implements a Type 0 configuration header
15:0	RO	0	Reserved	Reserved

#### 14.8.8 MTB\_LBAR: MSC Trace Buffer Lower BAR

CSR Register Name: MTB_LBAR: MSC Trace Buffer Lower BAR				
Bar: n/a		Reset: host_rst_b		Offset Start: 10
Bits	Access	Default	Label	Bit Description
31:20	RW	0	ADDR	<b>CSR and MTB BAR (Lower).</b> This register specifies the lower 32 bits of the configurable base address for CSRs (Configuration and Status Registers) and MTB (Memory Trace Buffer). This BAR is called BAR 0.
19:4	RO	0	Reserved	Hardwired to 0 to indicate the memory space size required by this CSR_MTB_BAR MMIO space is 1MB
3	RO	0	PF	Prefetchable: Value of 0 indicates the BAR cannot be prefetched

CSR Register Name: MTB_LBAR: MSC Trace Buffer Lower BAR					
Bar: n/a		Reset: host_rst_b		Offset Start: 10	Offset End: 13
Bits	Access	Default	Label	Bit Description	
2:1	RO	2	ADRNG	<b>Address Range:</b> Value of 0x2 indicates that the BAR is located anywhere system memory space (i.e. 64-bit addressing)	
0	RO	0	SPTY	Space Type: Value of 0 indicates the BAR is located in memory space	

#### 14.8.9 MTB\_UBAR: MSC Trace Buffer Upper BAR

CSR Register Name: MTB_UBAR: MSC Trace Buffer Upper BAR					
Bar: n/a		Reset: host_rst_b		Offset Start: 14	Offset End: 17
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	ADDR	<b>CSR and MTB BAR (Upper):</b> This register specifies the upper 32 bits of the configurable base address for CSRs (Configuration and Status Registers) and MTB (Memory Trace Buffer). This BAR is called BAR 0.	

#### 14.8.10 SW\_LBAR: Software Lower BAR

CSR Register Name: SW_LBAR: Software Lower BAR					
Bar: n/a		Reset: host_rst_b		Offset Start: 18	Offset End: 1b
Bits	Access	Default	Label	Bit Description	
31:23	RW	0	ADDR	<b>Software Trace BAR (Lower):</b> This register specifies the lower 32 bits of the configurable base address for Software trace data . This BAR is called BAR 1.	
22:4	RO	0	Reserved	Reserved: Hardwired to 0 to indicate the memory space size required by this BAR.	
3	RO	0	PF	Prefetchable: Value of 0 indicates the BAR cannot be prefetched	
2:1	RO	2	ADRNG	<b>Address Range:</b> Value of 0x2 indicates that the BAR is located anywhere system memory space (i.e. 64-bit addressing)	
0	RO	0	SPTY	Space Type: Value of 0 indicates the BAR is located in memory space	



### 14.8.11 SW\_UBAR: Software Upper BAR

CSR Register Name: SW_UBAR: Software Upper BAR				
Bar: n/a		Reset: host_rst_b		Offset Start: 1c
Bits	Access	Default	Label	Bit Description
31:0	RW	0	ADDR	<b>Software Trace BAR (Upper)</b> . This register specifies the upper 32 bits of the configurable base address for Software trace data . This BAR is called BAR 1.

### 14.8.12 RTIT\_LBAR: RTIT Lower BAR

CSR Register Name: RTIT_LBAR: RTIT Lower BAR				
Bar: n/a		Reset: host_rst_b		Offset Start: 20
Bits	Access	Default	Label	Bit Description
31:14	RW	0	ADDR	<b>Intel® Processor Trace BAR (Lower)</b> . This register specifies the lower 32 bits of the configurable base address for Intel® Processor Trace data . This BAR is called BAR 2.
13:4	RO	0	Reserved	Reserved: Hardwired to 0 to indicate the memory space size required by this RTIT_BAR MMIO space. The value of N in this field, and in the ADDR field, is determined by RTITCNT: RTITCNT N Comments 0 - No RTIT Srcs 1 8 256B 2 9 512B 3 10 1024B 4 10 1024B 5-8 11 2048B 9-16 12 4096B
3	RO	0	PF	Prefetchable: Value of 0 indicates the BAR cannot be prefetched
2:1	RO	2	TYPE	<b>Address Range</b> : Value of 0x2 indicates that the BAR is located anywhere system memory space (i.e. 64-bit addressing)
0	RO	0	MEM	Space Type: Value of 0 indicates the BAR is located in memory space



#### 14.8.13 RTIT\_UBAR: RTIT Upper BAR

CSR Register Name: RTIT_UBAR: RTIT Upper BAR					
Bar: n/a		Reset: host_rst_b		Offset Start: 24	Offset End: 27
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	ADDR	<b>Intel Processor Trace BAR (Upper)</b> . This register specifies the upper 32 bits of the configurable base address for Intel® Processor Trace data . This BAR is called BAR 2.	

#### 14.8.14 SVID: Subsystem Vendor ID

CSR Register Name: SVID: Subsystem Vendor ID					
Bar: n/a		Reset: host_rst_b		Offset Start: 2c	Offset End: 2f
Bits	Access	Default	Label	Bit Description	
31:16	RW	0	SID	<b>Subsystem ID</b> . Indicates the Subsystem ID for this device.	
15:0	RW	0	SVID	<b>Subsystem Vendor ID</b> . Indicates the Subsystem Vendor ID for this device.	

#### 14.8.15 CAP: Capabilities Pointer

CSR Register Name: CAP: Capabilities Pointer					
Bar: n/a		Reset: host_rst_b		Offset Start: 34	Offset End: 37
Bits	Access	Default	Label	Bit Description	
31:8	RO	0	Reserved	Reserved	
7:0	RO	40	CAPPTR	<b>Capability Pointer</b> : Pointer to first capability structure.	

#### 14.8.16 INTL: Interrupt Line

CSR Register Name: INTL: Interrupt Line					
Bar: n/a		Reset: host_rst_b		Offset Start: 3C	Offset End: 3F
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	Reserved	Reserved	
15:8	RW/O	1	INTPIN	<b>Interrupt Pin</b> . The Intel® Trace Hub uses a single INTx interrupt bonded to INTA	
7:0	RW	FF	INTL	<b>Interrupt Line</b> : Hardware does not use this field. Rather it is programmed by system software and device drivers to communicate interrupt line routing information	

#### 14.8.17 MSICID: MSI Capability ID

CSR Register Name: MSICID: MSI Capability ID					
Bar: n/a		Reset: host_rst_b		Offset Start: 40	Offset End: 43
Bits	Access	Default	Label	Bit Description	
31:24	RO	0	Reserved	Multiple Message Enable: Indicates the number of messages allocated to the device	
23	RO	1	AC64b	Multiple Message Capable: Value of 0 indicates the device only support single interrupt message	
22:20	RW	0	MME	MSI Enable: If set, MSI is enabled and legacy interrupts (INTx) will not be generated	
19:17	RO	3	MMC	<b>MSI Next Capability Pointer:</b> Value of 0 indicates there are no further capabilities ( i.e. the capability linked list is ended)	
16	RW	0	MSIE	MSI Capability ID: MSI Capability ID with a value of 05h indicating the presence of the MSI capability register set	
15:8	RO	0	MSINCP	64-bit Address Capable: The Intel® Trace Hub is capable of generating 64-bit memory addresses	
7:0	RO	5	MSICID	Reserved	

#### 14.8.18 MSILMA: MSI Lower Message Address

CSR Register Name: MSILMA: MSI Lower Message Address					
Bar: n/a		Reset: host_rst_b		Offset Start: 44	Offset End: 47
Bits	Access	Default	Label	Bit Description	
31:2	RW	0	MSILMA	MSI Message Lower Address: Lower 32-bits of system software assigned message address to the device with bits[1:0] always cleared indicating message address has to always be DW aligned	
1:0	RO	0	Reserved	Value of 0 indicates the memory address is always DW aligned	

#### 14.8.19 MSIUMA: MSI Upper Message Address

CSR Register Name: MSIUMA: MSI Upper Message Address					
Bar: n/a		Reset: host_rst_b		Offset Start: 48	Offset End: 4b
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	MSIUMA	MSI Message Upper Address: Upper 32 bits of system software assigned message address to the device	

#### 14.8.20 MSIMD: MSI Message Data

CSR Register Name: MSIMD: MSI Message Data				
Bar: n/a		Reset: host_rst_b	Offset Start: 4c	Offset End: 4f
Bits	Access	Default	Label	Bit Description
31:16	RO	0	Reserved	Reserved
15:0	RW	0	MSIMD	<b>MSI Message Data:</b> 16 bit message data pattern assigned by the system software to the device. When MSI is generated the actual data is 32 bit and the upper 16 bits are always 0

#### 14.8.21 FW\_LBAR: Firmware Lower Bar

CSR Register Name: FW_LBAR: Firmware Lower Bar				
Bar: n/a		Reset: host_rst_b	Offset Start: 70	Offset End: 73
Bits	Access	Default	Label	Bit Description
31:21	RW	0	ADDR	<b>Firmware Trace BAR (Lower)</b> . This register specifies the lower 32 bits of the configurable base address for Firmware trace data . This BAR is called BAR 3, though it is a non-standard BAR (not a BAR defined by the PCI Specification).
20:4	RO	0	Reserved	Reserved: Hardwired to 0 to indicate the memory space size required by this BAR.
3	RO	0	PF	Prefetchable: Value of 0 indicates the BAR cannot be prefetched
2:1	RO	2	TYPE	<b>Address Range:</b> Value of 0x2 indicates that the BAR is located anywhere system memory space (i.e. 64-bit addressing)
0	RO	0	MEM	Space Type: Value of 0 indicates the BAR is located in memory space

#### 14.8.22 FW\_UBAR: Firmware Upper Bar

CSR Register Name: FW_UBAR: Firmware Upper Bar				
Bar: n/a		Reset: host_rst_b	Offset Start: 74	Offset End: 77
Bits	Access	Default	Label	Bit Description
31:0	RW	0	UADDR	<b>Firmware Trace BAR (Upper)</b> . This register specifies the upper 32 bits of the configurable base address for Firmware trace data . This BAR is called BAR 3, though it is a non-standard BAR (not a BAR defined by the PCI Specification).

### 14.8.23 NPKDSC: NPK Device Specific Control

CSR Register Name: NPKDSC: NPK Device Specific Control				
Bar: n/a		Reset: host_rst_b	Offset Start: 80	Offset End: 83
Bits	Access	Default	Label	Bit Description
31:16	RO	0	Reserved2	Reserved
15:11	RO	0	Reserved1	Reserved
10	RW/1C	0	URD	Unsupported Request Detect: This bit is set when an unsupported request is detected
9	RW/1C	0	CTOE	CCB Timeout Timer Expired: This bit is asserted if the Inbound CCB timeout timer has expired at least once
8	RW/1C	0	STOE	Switch Timeout Timer Expired: This bit is asserted if the Inbound Switch timeout timer has expired at least once
7	RW	0	ISACT	<b>Sideband Bus Active:</b> Setting this bit will force the Intel® Trace Hub to request its Sideband bus clock (ISBCLK) independent of any activity.
6	RW	0	IPACT	<b>Primary Bus Active:</b> Setting this bit will force the Intel® Trace Hub to request its Primary bus clock (IPCLK) independent of any activity.
5	RW	0	TSACT	Time Stamping Active: Setting this bit will force the Intel Trace Hub to maintain requesting the time-stamping clock and disable internal clock gating
4	RW	0	TRACT	<b>Tracing Active:</b> Setting this bit will force the Intel Trace Hub to request its NPCLK clock. This effectively disables clock gating
3	RW	0	URRE	Unsupported Request Reporting Enable: When set, this bit enables the reporting unsupported requests as system errors
2	RW/1C	0	CDINTS	<b>Capture Done Interrupt Status:</b> Formerly Legacy Interrupt Asserted. Equivalent to MSUSTS.MSU_INT. For software compatibility, this bit indicates when the capture done event has occurred (see MSU Chapter). Software can clear the capture done interrupt event by writing a 1 to this bit, or writing a 1 to the MSUSTS.MSU_INT bit
1	RW	0	FLR	<b>Software Reset:</b> This is the Intel(R) Trace Hub's software-controlled functional level reset. Writing a 1 to this bit will assert the reset. Reading this bit will always return a zero
0	RO	0	Reserved	Reserved



#### 14.8.24 NPKDSD: NPK Device Specific Defeature

CSR Register Name: RSVD: NPK Device Specific Defeature					
Bar: n/a		Reset: host_rst_b		Offset Start: 90	Offset End: 93
Bits	Access	Default	Label	Bit Description	
31:1	RO	0	reserved	Reserved for future use	
0	RW	0	FON	Force On: If set, this bit will prevent the Intel® Trace Hub logic from gating its clock or de-asserting its power ok signals	

## 14.9 PTI Registers

See LPP\_CTL register in Global Trace Hub Registers section.

## 14.10 Power Control Registers

### 14.10.1 Power Control Registers Summary

Power Control Registers			
Offset Start	Offset End	Symbol	Register Name/Function
FFD00	FFD03	PMTCS	Power Mode Transition Control and status
FFD04	FFD07	PMTTL	Power Mode Transition Time Limit Register

### 14.10.2 PMTCS: Power Mode Transition Control and status

CSR Register Name: PMTCS: Power Mode Transition Control and status				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: FFD00	Offset End: FFD03	
Bits	Access	Default	Label	Bit Description
31:17	RO	0	RSVDD0	PMTCS Reserved
16	RW	1	WSYNC	<b>Wait Sync:</b> If asserted, the PMTC FSM will wait for a valid timestamp before re-enabling trace sources (i.e. before moving back to ACTIVE state).
15:1	RO	0	RSVD	Reserved for future use
0	RW	0	BLKDRNEN	<b>Block and Drain Enable:</b> This bit enables the automatic block and drain/un-block and enable operations. This bit must be set before the low-power state entry and must be saved/restored since it's sampled on the exit of the restore phase. Clearing this bit will prevent North Peak from executing any automated block/un-block operations.

### 14.10.3 PMTTL: Power Mode Transition Time Limit Register

CSR Register Name: PMTTL: Power Mode Transition Time Limit Register				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: FFD04	Offset End: FFD07	
Bits	Access	Default	Label	Bit Description
31:0	RW	FFFF	TTL	<b>Transition Time Limit:</b> This register specifies the number of clocks (in the IOSF primary clock domain) the power mode transition logic will wait for North Peak to drain before it logs an error and stops. Setting this value to zero will force North Peak to wait forever. Note that timeout timer must be programmed to a larger value than that of the NDB_WAITEMPTY timeout timer.



## 14.11 SoCHAP Registers

### 14.11.1 SoCHAP Registers Summary

SoCHAP Registers			
Offset Start	Offset End	Symbol	Register Name/Function
5000	5003	SDC	SoCHAP Device Capabilities
5004	5007	SSS	SoCHAP Software Register
5008	500B	CEC_0	Customizable Event Creation 0
500C	500F	CEC_1	Customizable Event Creation 1
5010	5013	CEC_2	Customizable Event Creation 2
5014	5017	CEC_3	Customizable Event Creation 3
5018	501B	CEC_4	Customizable Event Creation 4
501C	501F	CEC_5	Customizable Event Creation 5
5020	5023	CEC_6	Customizable Event Creation 6
5024	5027	CEC_7	Customizable Event Creation 7
5028	502B	CEC_8	Customizable Event Creation 8
502C	502F	CEC_9	Customizable Event Creation 9
5030	5033	CEC_10	Customizable Event Creation 10
5034	5037	CEC_11	Customizable Event Creation 11
5038	503B	CEC_12	Customizable Event Creation 12
503C	503F	CEC_13	Customizable Event Creation 13
5040	5043	CEC_14	Customizable Event Creation 14
5044	5047	CEC_15	Customizable Event Creation 15
5048	504B	SOCHAPCMD_0	SoCHAP Command Register 0
504C	504F	SOCHAPEV_0	SoCHAP Events Register 0
5050	5053	SOCHAPSTAT_0	SoCHAP Status Register 0
5054	5057	SOCHAPDATA_0	SoCHAP Data Register 0
5058	505B	SOCHAPCMD_1	SoCHAP Command Register 1
505C	505F	SOCHAPEV_1	SoCHAP Events Register 1
5060	5063	SOCHAPSTAT_1	SoCHAP Status Register 1
5064	5067	SOCHAPDATA_1	SoCHAP Data Register 1
5068	506B	SOCHAPCMD_2	SoCHAP Command Register 2
506C	506F	SOCHAPEV_2	SoCHAP Events Register 2
5070	5073	SOCHAPSTAT_2	SoCHAP Status Register 2



SoCHAP Registers			
Offset Start	Offset End	Symbol	Register Name/Function
5074	5077	SOCHAPDATA_2	SoCHAP Data Register 2
5078	507B	SOCHAPCMD_3	SoCHAP Command Register 3
507C	507F	SOCHAPEV_3	SoCHAP Events Register 3
5080	5083	SOCHAPSTAT_3	SoCHAP Status Register 3
5084	5087	SOCHAPDATA_3	SoCHAP Data Register 3
5088	508B	SOCHAPCMD_4	SoCHAP Command Register 4
508C	508F	SOCHAPEV_4	SoCHAP Events Register 4
5090	5093	SOCHAPSTAT_4	SoCHAP Status Register 4
5094	5097	SOCHAPDATA_4	SoCHAP Data Register 4
5098	509B	SOCHAPCMD_5	SoCHAP Command Register 5
509C	509F	SOCHAPEV_5	SoCHAP Events Register 5
50A0	50A3	SOCHAPSTAT_5	SoCHAP Status Register 5
50A4	50A7	SOCHAPDATA_5	SoCHAP Data Register 5
50A8	50AB	SOCHAPCMD_6	SoCHAP Command Register 6
50AC	50AF	SOCHAPEV_6	SoCHAP Events Register 6
50B0	50B3	SOCHAPSTAT_6	SoCHAP Status Register 6
50B4	50B7	SOCHAPDATA_6	SoCHAP Data Register 6
50B8	50BB	SOCHAPCMD_7	SoCHAP Command Register 7
50BC	50BF	SOCHAPEV_7	SoCHAP Events Register 7
50C0	50C3	SOCHAPSTAT_7	SoCHAP Status Register 7
50C4	50C7	SOCHAPDATA_7	SoCHAP Data Register 7
50C8	50CB	SOCHAPCMD_8	SoCHAP Command Register 8
50CC	50CF	SOCHAPEV_8	SoCHAP Events Register 8
50D0	50D3	SOCHAPSTAT_8	SoCHAP Status Register 8
50D4	50D7	SOCHAPDATA_8	SoCHAP Data Register 8
50D8	50DB	SOCHAPCMD_9	SoCHAP Command Register 9
50DC	50DF	SOCHAPEV_9	SoCHAP Events Register 9
50E0	50E3	SOCHAPSTAT_9	SoCHAP Status Register 9
50E4	50E7	SOCHAPDATA_9	SoCHAP Data Register 9
50E8	50EB	SOCHAPCMD_10	SoCHAP Command Register 10
50EC	50EF	SOCHAPEV_10	SoCHAP Events Register 10
50F0	50F3	SOCHAPSTAT_10	SoCHAP Status Register 10

SoCHAP Registers			
Offset Start	Offset End	Symbol	Register Name/Function
50F4	50F7	SOCHAPDATA_10	SoCHAP Data Register 10
50F8	50FB	SOCHAPCMD_11	SoCHAP Command Register 11
50FC	50FF	SOCHAPEV_11	SoCHAP Events Register 11
5100	5103	SOCHAPSTAT_11	SoCHAP Status Register 11
5104	5107	SOCHAPDATA_11	SoCHAP Data Register 11
5108	510B	SOCHAPCMD_12	SoCHAP Command Register 12
510C	510F	SOCHAPEV_12	SoCHAP Events Register 12
5110	5113	SOCHAPSTAT_12	SoCHAP Status Register 12
5114	5117	SOCHAPDATA_12	SoCHAP Data Register 12
5118	511B	SOCHAPCMD_13	SoCHAP Command Register 13
511C	511F	SOCHAPEV_13	SoCHAP Events Register 13
5120	5123	SOCHAPSTAT_13	SoCHAP Status Register 13
5124	5127	SOCHAPDATA_13	SoCHAP Data Register 13
5128	512B	SOCHAPCMD_14	SoCHAP Command Register 14
512C	512F	SOCHAPEV_14	SoCHAP Events Register 14
5130	5133	SOCHAPSTAT_14	SoCHAP Status Register 14
5134	5137	SOCHAPDATA_14	SoCHAP Data Register 14
5138	513B	SOCHAPCMD_15	SoCHAP Command Register 15
513C	513F	SOCHAPEV_15	SoCHAP Events Register 15
5140	5143	SOCHAPSTAT_15	SoCHAP Status Register 15
5144	5147	SOCHAPDATA_15	SoCHAP Data Register 15
5148	514B	SOCHAPPKTCTRL	SoCHAP Packet Control Register
514C	514F	SOCHAPPKTCAP	SoCHAP Packet Capabilities Register

#### 14.11.2 SDC: SoCHAP Device Capabilities

CSR Register Name: SDC: SoCHAP Device Capabilities				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5000
Bits	Access	Default	Label	Bit Description
31:16	RO	0	Reserved	SDC Reserved
15	RW	0	CE	<b>CHAP Enable:</b> This register enables the SoCHAP block. When this bit is cleared, the SoCHAP block is disabled except for this register.



CSR Register Name: SDC: SoCHAP Device Capabilities					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5000	Offset End: 5003
Bits	Access	Default	Label	Bit Description	
14	RW	0	IU	<b>All Counters In Use:</b> Software Mutex bit. After a full RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. Writing a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the entire SoCHAP. This bit has no other effect on any CHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize the SoCHAPs. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
13	RW	0	Counter_Freeze	Reserved	
12:8	RO	F	VR	<b>VIS Resources:</b> This register indicates how many VIS input signals are being fed into this block. Up to 32 inputs are supported through VIS. Note, actual resources = VR + 1.	
7:4	RO	F	CR	<b>Counter Resources:</b> This value exposes the number of counters available to the SoCHAP block. For each counter exposed, there exists 4 registers for command, event, status, and data. The default value of 0h corresponds to 1 SoCHAP counter resource. Note, actual resources = CR + 1.	
3:0	RO	F	CECR	<b>Customizable Event Creation Resources:</b> This register exposes the number of custom events available. The default value of 0h corresponds to one SoCHAP customizable event resource. For each customizable event 1 control register is present. Note, actual resources = CECR + 1.	

#### 14.11.3 SSS: SoCHAP Software Register

CSR Register Name: SSS: SoCHAP Software Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5004	Offset End: 5007
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	SRTC_HPAD	<b>Scratchpad Register:</b> This register is made available to keep track of the Counter and Custom event usage within software.	



## 14.11.4 CEC\_0: Customizable Event Creation 0

CSR Register Name: CEC_0: Customizable Event Creation 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5008	Offset End: 500B
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.  0: The input resource bit is considered in event calculations.  1: The VIS bit is delayed by 1 clock before considering it in event calculations. This is particularly useful for doing state machine arc coverage. For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals. The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters	
23:16	RW	0	MSKE	<b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:  0: This input resource bit is considered in event calculations.  1: This input resource bit is ignored in event calculations.	
15:8	RW	0	CMPVAL	<b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function. When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters. When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used. CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4. CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240. CMPVAL > 8b1: Reserved for future use	

CSR Register Name: CEC_0: Customizable Event Creation 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5008	Offset End: 500B
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<p><b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned.</p> <p>00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain.</p> <p>01: Evaluate expression when any of the expressions corresponding valid pulses are asserted.</p> <p>10: Reserved.</p> <p>11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.</p>	
5:3	RW	0	IRN	<p><b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.</p> <p>000: Resource #1 VIS lane 0 (xbar[7:0])</p> <p>001: Resource #2 Previous CEC flopped path</p> <p>010: Resource #3 CEC compare out [7:0]</p> <p>011: Resource #4 Threshold Event [7:0]</p> <p>100: Resource #5 VIS Lane 1 (xbar[15:8])</p> <p>101: Resource #6 VIS Lane 2 (xbar[23:16])</p> <p>110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources</p> <p>111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources</p>	
2:0	RW	0	CMPFCN	<p><b>Compare Function:</b></p> <p>000: Compare and assert if any are equal (Can be used as OR function)</p> <p>001: Compare and output signal if greater than</p> <p>010: Compare and assert output if equal to (Can also be used as AND function)</p> <p>011: Compare and assert output if greater than or equal</p> <p>100: Compare and assert output if less than</p> <p>101: Compare and assert output if not equal</p> <p>110: Compare and assert output if less than or equal</p> <p>111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings).</p> <p>* NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger</p>	

#### 14.11.5 CEC\_1: Customizable Event Creation 1

CSR Register Name: CEC_1: Customizable Event Creation 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 500C	Offset End: 500F
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_1: Customizable Event Creation 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 500C	Offset End: 500F
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.6 CEC\_2: Customizable Event Creation 2

CSR Register Name: CEC_2: Customizable Event Creation 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5010	Offset End: 5013
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	



CSR Register Name: CEC_2: Customizable Event Creation 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5010	Offset End: 5013
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register. 000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

### 14.11.7 CEC\_3: Customizable Event Creation 3

CSR Register Name: CEC_3: Customizable Event Creation 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5014	Offset End: 5017
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_3: Customizable Event Creation 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5014	Offset End: 5017
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.8 CEC\_4: Customizable Event Creation 4

CSR Register Name: CEC_4: Customizable Event Creation 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5018	Offset End: 501B
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_4: Customizable Event Creation 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5018	Offset End: 501B
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

### 14.11.9 CEC\_5: Customizable Event Creation 5

CSR Register Name: CEC_5: Customizable Event Creation 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 501C	Offset End: 501F
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_5: Customizable Event Creation 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 501C	Offset End: 501F
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.10 CEC\_6: Customizable Event Creation 6

CSR Register Name: CEC_6: Customizable Event Creation 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5020	Offset End: 5023
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_6: Customizable Event Creation 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5020	Offset End: 5023
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

### 14.11.11 CEC\_7: Customizable Event Creation 7

CSR Register Name: CEC_7: Customizable Event Creation 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5024	Offset End: 5027
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_7: Customizable Event Creation 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5024	Offset End: 5027
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.12 CEC\_8: Customizable Event Creation 8

CSR Register Name: CEC_8: Customizable Event Creation 8				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 5028	Offset End: 502B
Bits	Access	Default	Label	Bit Description
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>

CSR Register Name: CEC_8: Customizable Event Creation 8					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5028	Offset End: 502B
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

### 14.11.13 CEC\_9: Customizable Event Creation 9

CSR Register Name: CEC_9: Customizable Event Creation 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 502C	Offset End: 502F
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest.</p> <p>Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_9: Customizable Event Creation 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 502C	Offset End: 502F
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.14 CEC\_10: Customizable Event Creation 10

CSR Register Name: CEC_10: Customizable Event Creation 10				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5030	Offset End: 5033	
Bits	Access	Default	Label	Bit Description
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest.</p> <p>Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>



CSR Register Name: CEC_10: Customizable Event Creation 10					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5030	Offset End: 5033
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

### 14.11.15 CEC\_11: Customizable Event Creation 11

CSR Register Name: CEC_11: Customizable Event Creation 11				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5034	Offset End: 5037	
Bits	Access	Default	Label	Bit Description
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest.</p> <p>Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>



CSR Register Name: CEC_11: Customizable Event Creation 11					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5034	Offset End: 5037
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0])  001: Resource #2 Previous CEC flopped path  010: Resource #3 CEC compare out [7:0]  011: Resource #4 Threshold Event [7:0]  100: Resource #5 VIS Lane 1 (xbar[15:8])  101: Resource #6 VIS Lane 2 (xbar[23:16])  110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources  111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.16 CEC\_12: Customizable Event Creation 12

CSR Register Name: CEC_12: Customizable Event Creation 12					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5038	Offset End: 503B
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

<b>CSR Register Name: CEC_12: Customizable Event Creation 12</b>					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5038	<b>Offset End:</b> 503B
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
7:6	RW	0	CVG	<p><b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned.</p> <p>00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain.</p> <p>01: Evaluate expression when any of the expressions corresponding valid pulses are asserted.</p> <p>10: Reserved.</p> <p>11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.</p>	
5:3	RW	0	IRN	<p><b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.</p> <p>000: Resource #1 VIS lane 0 (xbar[7:0])</p> <p>001: Resource #2 Previous CEC flopped path</p> <p>010: Resource #3 CEC compare out [7:0]</p> <p>011: Resource #4 Threshold Event [7:0]</p> <p>100: Resource #5 VIS Lane 1 (xbar[15:8])</p> <p>101: Resource #6 VIS Lane 2 (xbar[23:16])</p> <p>110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources</p> <p>111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources</p>	
2:0	RW	0	CMPFCN	<p><b>Compare Function:</b></p> <p>000: Compare and assert if any are equal (Can be used as OR function)</p> <p>001: Compare and output signal if greater than</p> <p>010: Compare and assert output if equal to (Can also be used as AND function)</p> <p>011: Compare and assert output if greater than or equal</p> <p>100: Compare and assert output if less than</p> <p>101: Compare and assert output if not equal</p> <p>110: Compare and assert output if less than or equal</p> <p>111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings).</p> <p>* NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger</p>	

#### 14.11.17 CEC\_13: Customizable Event Creation 13

CSR Register Name: CEC_13: Customizable Event Creation 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 503C	Offset End: 503F
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_13: Customizable Event Creation 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 503C	Offset End: 503F
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	

#### 14.11.18 CEC\_14: Customizable Event Creation 14

CSR Register Name: CEC_14: Customizable Event Creation 14					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5040	Offset End: 5043
Bits	Access	Default	Label	Bit Description	
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest. Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>	
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>	
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>	

CSR Register Name: CEC_14: Customizable Event Creation 14					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5040	Offset End: 5043
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<p><b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned.</p> <p>00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain.</p> <p>01: Evaluate expression when any of the expressions corresponding valid pulses are asserted.</p> <p>10: Reserved.</p> <p>11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.</p>	
5:3	RW	0	IRN	<p><b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.</p> <p>000: Resource #1 VIS lane 0 (xbar[7:0])</p> <p>001: Resource #2 Previous CEC flopped path</p> <p>010: Resource #3 CEC compare out [7:0]</p> <p>011: Resource #4 Threshold Event [7:0]</p> <p>100: Resource #5 VIS Lane 1 (xbar[15:8])</p> <p>101: Resource #6 VIS Lane 2 (xbar[23:16])</p> <p>110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources</p> <p>111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources</p>	
2:0	RW	0	CMPFCN	<p><b>Compare Function:</b></p> <p>000: Compare and assert if any are equal (Can be used as OR function)</p> <p>001: Compare and output signal if greater than</p> <p>010: Compare and assert output if equal to (Can also be used as AND function)</p> <p>011: Compare and assert output if greater than or equal</p> <p>100: Compare and assert output if less than</p> <p>101: Compare and assert output if not equal</p> <p>110: Compare and assert output if less than or equal</p> <p>111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings).</p> <p>* NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger</p>	

### 14.11.19 CEC\_15: Customizable Event Creation 15

CSR Register Name: CEC_15: Customizable Event Creation 15				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5044	Offset End: 5047	
Bits	Access	Default	Label	Bit Description
31:24	RW	0	DLYSEL	<p><b>Delay Selection:</b> Bit field LSB corresponds to VIS bit 0.</p> <p>0: The input resource bit is considered in event calculations.</p> <p>1: The VIS bit is delayed by 1 clock before considering it in event calculations.</p> <p>This is particularly useful for doing state machine arc coverage.</p> <p>For example, VIS bits 3:0 and VIS 7:4 could be programmed to the same 4 present state, state machine signals. The appropriate inversion selections would be made depending on which state transition is of interest.</p> <p>Bits 31:28 in the delay selection would be programmed to 1111, indicating use a pipe delayed version of the state signals.</p> <p>The resulting AND of the now preconditioned VIS 7:4 and VIS 3:0 signals would indicate the number of times the arc of interest was taken. This could be recorded with the SoCHAP counters</p>
23:16	RW	0	MSKE	<p><b>Mask Enable:</b> Bit field LSB corresponds to input resource bit 0. These 8 bits are used to mask off entries from the comparison. For each bit:</p> <p>0: This input resource bit is considered in event calculations.</p> <p>1: This input resource bit is ignored in event calculations.</p>
15:8	RW	0	CMPVAL	<p><b>Compare Value:</b> Bit field LSB corresponds to input resource bit 0. This field is loaded to compare against the 8 input resource signals that are fed into this block. The type of comparison that is done is controlled by the Compare Function.</p> <p>When the compare function is true, then the signal for the input resource event is asserted. This in turn can be counted by any of the SoCHAP counters.</p> <p>When the CEC is programmed for alternative mode (CMPFCN == ALT), this register is used to control the type of mode that is being used.</p> <p>CMPVAL == 8b0: ONES mode. In this mode, the number of signals that are asserted are added up and passed as a binary value to the counters as the event. Example: input = b11110000, count by 4.</p> <p>CMPVAL == 8b1: COUNTER mode. The masked value is passed directly to the counters as a binary value. This allows the counters to increment or decrement by up to 255 at any one time. Example: input = b11110000, count by 240.</p> <p>CMPVAL &gt; 8b1: Reserved for future use</p>

CSR Register Name: CEC_15: Customizable Event Creation 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5044	Offset End: 5047
Bits	Access	Default	Label	Bit Description	
7:6	RW	0	CVG	<b>Clock Valid Generation:</b> This selects which mode is being used for qualification of the data with the corresponding valid bits. This is needed since signals from different domains may not have their valid pulses aligned. 00: Only evaluate expression when all 'valid' signals are asserted. This mode is used when signals come from the same clock domain. 01: Evaluate expression when any of the expressions corresponding valid pulses are asserted. 10: Reserved. 11: Bypass the valid and evaluate expression based on the raw data alone. The output of the Custom Event block asserts its valid signal continuously.	
5:3	RW	0	IRN	<b>Input Resource Number:</b> Each Custom Event can handle up to 8 input lanes. The mapping of the lanes is given by the CEC lane mapping fields in the SoCHAP capabilities register.  000: Resource #1 VIS lane 0 (xbar[7:0]) 001: Resource #2 Previous CEC flopped path 010: Resource #3 CEC compare out [7:0] 011: Resource #4 Threshold Event [7:0] 100: Resource #5 VIS Lane 1 (xbar[15:8]) 101: Resource #6 VIS Lane 2 (xbar[23:16]) 110: Resource #7 CEC Compare out [15:8] or VIS Lane 3 if only 8 CEC Resources 111: Resource #8 Threshold Event [15:8] or VIS Lane 3 (xbar[31:24]) if only 8 Counter Resources	
2:0	RW	0	CMPFCN	<b>Compare Function:</b> 000: Compare and assert if any are equal (Can be used as OR function) 001: Compare and output signal if greater than 010: Compare and assert output if equal to (Can also be used as AND function) 011: Compare and assert output if greater than or equal 100: Compare and assert output if less than 101: Compare and assert output if not equal 110: Compare and assert output if less than or equal 111: Alternative mode (non-compare modes). This mode enables the alternative mode for the counters. These modes use the compare value (CMPVAL) as an encoding to determine what to do (see CMPVAL for alternative encodings). * NOTE: When using the additive mode, the delay selection bits must be set to not take the flopped version of the signals. Also, when used as the trigger event, the add event only uses bit 0 for the trigger	



## 14.11.20 SOCHAPCMD\_0: SoCHAP Command Register 0

CSR Register Name: SOCHAPCMD_0: SoCHAP Command Register 0				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5048	Offset End: 504B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_0: SoCHAP Command Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5048	Offset End: 504B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_0: SoCHAP Command Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5048	Offset End: 504B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	Reserved	
14:13	RW	0	Reserved	Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.21 SOCHAPEV\_0: SoCHAP Events Register 0

CSR Register Name: SOCHAPEV_0: SoCHAP Events Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 504C	Offset End: 504F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_0: SoCHAP Events Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 504C	Offset End: 504F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.22 SOCHAPSTAT\_0: SoCHAP Status Register 0

CSR Register Name: SOCHAPSTAT_0: SoCHAP Status Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5050	Offset End: 5053
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_0: SoCHAP Status Register 0				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5050
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.23 SOCHAPDATA\_0: SoCHAP Data Register 0

<b>CSR Register Name:</b> SOCHAPDATA_0: SoCHAP Data Register 0				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5054
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 0 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.24 SOCHAPCMD\_1: SoCHAP Command Register 1

CSR Register Name: SOCHAPCMD_1: SoCHAP Command Register 1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5058	Offset End: 505B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_1: SoCHAP Command Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5058	Offset End: 505B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_1: SoCHAP Command Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5058	Offset End: 505B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.25 SOCHAPEV\_1: SoCHAP Events Register 1

CSR Register Name: SOCHAPEV_1: SoCHAP Events Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 505C	Offset End: 505F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_1: SoCHAP Events Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 505C	Offset End: 505F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.26 SOCHAPSTAT\_1: SoCHAP Status Register 1

CSR Register Name: SOCHAPSTAT_1: SoCHAP Status Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5060	Offset End: 5063
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_1: SoCHAP Status Register 1				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5060
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.27 SOCHAPDATA\_1: SoCHAP Data Register 1

<b>CSR Register Name:</b> SOCHAPDATA_1: SoCHAP Data Register 1				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5064
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 1 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.28 SOCHAPCMD\_2: SoCHAP Command Register 2

CSR Register Name: SOCHAPCMD_2: SoCHAP Command Register 2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5068	Offset End: 506B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_2: SoCHAP Command Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5068	Offset End: 506B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_2: SoCHAP Command Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5068	Offset End: 506B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.29 SOCHAPEV\_2: SoCHAP Events Register 2

CSR Register Name: SOCHAPEV_2: SoCHAP Events Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 506C	Offset End: 506F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_2: SoCHAP Events Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 506C	Offset End: 506F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.30 SOCHAPSTAT\_2: SoCHAP Status Register 2

CSR Register Name: SOCHAPSTAT_2: SoCHAP Status Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5070	Offset End: 5073
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_2: SoCHAP Status Register 2				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5070
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.31 SOCHAPDATA\_2: SoCHAP Data Register 2

<b>CSR Register Name:</b> SOCHAPDATA_2: SoCHAP Data Register 2				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5074
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 2 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.32 SOCHAPCMD\_3: SoCHAP Command Register 3

CSR Register Name: SOCHAPCMD_3: SoCHAP Command Register 3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5078	Offset End: 507B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)

CSR Register Name: SOCHAPCMD_3: SoCHAP Command Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5078	Offset End: 507B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_3: SoCHAP Command Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5078	Offset End: 507B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.33 SOCHAPEV\_3: SoCHAP Events Register 3

CSR Register Name: SOCHAPEV_3: SoCHAP Events Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 507C	Offset End: 507F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_3: SoCHAP Events Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 507C	Offset End: 507F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.34 SOCHAPSTAT\_3: SoCHAP Status Register 3

CSR Register Name: SOCHAPSTAT_3: SoCHAP Status Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5080	Offset End: 5083
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_3: SoCHAP Status Register 3				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5080
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.35 SOCHAPDATA\_3: SoCHAP Data Register 3

<b>CSR Register Name:</b> SOCHAPDATA_3: SoCHAP Data Register 3				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5084
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 3 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.36 SOCHAPCMD\_4: SoCHAP Command Register 4

CSR Register Name: SOCHAPCMD_4: SoCHAP Command Register 4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5088	Offset End: 508B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_4: SoCHAP Command Register 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5088	Offset End: 508B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_4: SoCHAP Command Register 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5088	Offset End: 508B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.37 SOCHAPEV\_4: SoCHAP Events Register 4

CSR Register Name: SOCHAPEV_4: SoCHAP Events Register 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 508C	Offset End: 508F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_4: SoCHAP Events Register 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 508C	Offset End: 508F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.38 SOCHAPSTAT\_4: SoCHAP Status Register 4

CSR Register Name: SOCHAPSTAT_4: SoCHAP Status Register 4					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5090	Offset End: 5093
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_4: SoCHAP Status Register 4				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5090
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.39 SOCHAPDATA\_4: SoCHAP Data Register 4

<b>CSR Register Name:</b> SOCHAPDATA_4: SoCHAP Data Register 4				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5094
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 4 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.

#### 14.11.40 SOCHAPCMD\_5: SoCHAP Command Register 5

CSR Register Name: SOCHAPCMD_5: SoCHAP Command Register 5				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5098	Offset End: 509B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_5: SoCHAP Command Register 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5098	Offset End: 509B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_5: SoCHAP Command Register 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5098	Offset End: 509B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.41 SOCHAPEV\_5: SoCHAP Events Register 5

CSR Register Name: SOCHAPEV_5: SoCHAP Events Register 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 509C	Offset End: 509F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_5: SoCHAP Events Register 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 509C	Offset End: 509F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.42 SOCHAPSTAT\_5: SoCHAP Status Register 5

CSR Register Name: SOCHAPSTAT_5: SoCHAP Status Register 5					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50A0	Offset End: 50A3
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_5: SoCHAP Status Register 5				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50A0
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.43 SOCHAPDATA\_5: SoCHAP Data Register 5

<b>CSR Register Name:</b> SOCHAPDATA_5: SoCHAP Data Register 5				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50A4
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 5 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.44 SOCHAPCMD\_6: SoCHAP Command Register 6

CSR Register Name: SOCHAPCMD_6: SoCHAP Command Register 6				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 50A8	Offset End: 50AB	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_6: SoCHAP Command Register 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50A8	Offset End: 50AB
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_6: SoCHAP Command Register 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50A8	Offset End: 50AB
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.45 SOCHAPEV\_6: SoCHAP Events Register 6

CSR Register Name: SOCHAPEV_6: SoCHAP Events Register 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50AC	Offset End: 50AF
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_6: SoCHAP Events Register 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50AC	Offset End: 50AF
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.46 SOCHAPSTAT\_6: SoCHAP Status Register 6

CSR Register Name: SOCHAPSTAT_6: SoCHAP Status Register 6					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50B0	Offset End: 50B3
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_6: SoCHAP Status Register 6				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50B0
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.47 SOCHAPDATA\_6: SoCHAP Data Register 6

<b>CSR Register Name:</b> SOCHAPDATA_6: SoCHAP Data Register 6				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50B4
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 6 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.48 SOCHAPCMD\_7: SoCHAP Command Register 7

CSR Register Name: SOCHAPCMD_7: SoCHAP Command Register 7				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 50B8	Offset End: 50BB	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_7: SoCHAP Command Register 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50B8	Offset End: 50BB
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_7: SoCHAP Command Register 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50B8	Offset End: 50BB
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.49 SOCHAPEV\_7: SoCHAP Events Register 7

CSR Register Name: SOCHAPEV_7: SoCHAP Events Register 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50BC	Offset End: 50BF
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_7: SoCHAP Events Register 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50BC	Offset End: 50BF
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.50 SOCHAPSTAT\_7: SoCHAP Status Register 7

CSR Register Name: SOCHAPSTAT_7: SoCHAP Status Register 7					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50C0	Offset End: 50C3
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name:</b> SOCHAPSTAT_7: SoCHAP Status Register 7				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50C0
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.51 SOCHAPDATA\_7: SoCHAP Data Register 7

<b>CSR Register Name:</b> SOCHAPDATA_7: SoCHAP Data Register 7				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50C4
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 7 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



### 14.11.52 SOCHAPCMD\_8: SoCHAP Command Register 8

CSR Register Name: SOCHAPCMD_8: SoCHAP Command Register 8				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 50C8	Offset End: 50CB	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_8: SoCHAP Command Register 8					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50C8	Offset End: 50CB
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_8: SoCHAP Command Register 8					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50C8	Offset End: 50CB
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.53 SOCHAPEV\_8: SoCHAP Events Register 8

CSR Register Name: SOCHAPEV_8: SoCHAP Events Register 8					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50CC	Offset End: 50CF
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_8: SoCHAP Events Register 8				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50CC
Bits	Access	Default	Label	Bit Description
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.
15	RW	0	INCOCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.
14	RW	0	Reserved	SOCHAPEV Reserved
13	RW	0	INCEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8
8:0	RW	0	INCE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh



## 14.11.54 SOCHAPSTAT\_8: SoCHAP Status Register 8

CSR Register Name: SOCHAPSTAT_8: SoCHAP Status Register 8					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50D0	Offset End: 50D3
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

CSR Register Name: SOCHAPSTAT_8: SoCHAP Status Register 8				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50D0
Bits	Access	Default	Label	Bit Description
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.55 SOCHAPDATA\_8: SoCHAP Data Register 8

CSR Register Name: SOCHAPDATA_8: SoCHAP Data Register 8				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50D4
Bits	Access	Default	Label	Bit Description
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 8 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.56 SOCHAPCMD\_9: SoCHAP Command Register 9

CSR Register Name: SOCHAPCMD_9: SoCHAP Command Register 9				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50D8
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_9: SoCHAP Command Register 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50D8	Offset End: 50DB
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_9: SoCHAP Command Register 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50D8	Offset End: 50DB
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.57 SOCHAPEV\_9: SoCHAP Events Register 9

CSR Register Name: SOCHAPEV_9: SoCHAP Events Register 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50DC	Offset End: 50DF
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_9: SoCHAP Events Register 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50DC	Offset End: 50DF
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.58 SOCHAPSTAT\_9: SoCHAP Status Register 9

CSR Register Name: SOCHAPSTAT_9: SoCHAP Status Register 9					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50E0	Offset End: 50E3
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	



CSR Register Name: SOCHAPSTAT_9: SoCHAP Status Register 9				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50E0
Bits	Access	Default	Label	Bit Description
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.59 SOCHAPDATA\_9: SoCHAP Data Register 9

CSR Register Name: SOCHAPDATA_9: SoCHAP Data Register 9				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50E4
Bits	Access	Default	Label	Bit Description
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 9 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.60 SOCHAPCMD\_10: SoCHAP Command Register 10

CSR Register Name: SOCHAPCMD_10: SoCHAP Command Register 10				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 50E8	Offset End: 50EB	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_10: SoCHAP Command Register 10					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50E8	Offset End: 50EB
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_10: SoCHAP Command Register 10					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50E8	Offset End: 50EB
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.61 SOCHAPEV\_10: SoCHAP Events Register 10

CSR Register Name: SOCHAPEV_10: SoCHAP Events Register 10					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50EC	Offset End: 50EF
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_10: SoCHAP Events Register 10					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50EC	Offset End: 50EF
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	INCOCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	INCEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	INCE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.62 SOCHAPSTAT\_10: SoCHAP Status Register 10

CSR Register Name: SOCHAPSTAT_10: SoCHAP Status Register 10					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50F0	Offset End: 50F3
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name: SOCHAPSTAT_10: SoCHAP Status Register 10</b>				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50F0
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.63 SOCHAPDATA\_10: SoCHAP Data Register 10

<b>CSR Register Name: SOCHAPDATA_10: SoCHAP Data Register 10</b>				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 50F4
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 10 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.64 SOCHAPCMD\_11: SoCHAP Command Register 11

CSR Register Name: SOCHAPCMD_11: SoCHAP Command Register 11				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 50F8	Offset End: 50FB	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_11: SoCHAP Command Register 11					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50F8	Offset End: 50FB
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_11: SoCHAP Command Register 11					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50F8	Offset End: 50FB
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.65 SOCHAPEV\_11: SoCHAP Events Register 11

CSR Register Name: SOCHAPEV_11: SoCHAP Events Register 11					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50FC	Offset End: 50FF
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_11: SoCHAP Events Register 11					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 50FC	Offset End: 50FF
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	INCOCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	INCEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	INCE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.66 SOCHAPSTAT\_11: SoCHAP Status Register 11

CSR Register Name: SOCHAPSTAT_11: SoCHAP Status Register 11					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5100	Offset End: 5103
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name: SOCHAPSTAT_11: SoCHAP Status Register 11</b>				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5100
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.67 SOCHAPDATA\_11: SoCHAP Data Register 11

<b>CSR Register Name: SOCHAPDATA_11: SoCHAP Data Register 11</b>				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5104
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 11 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.



## 14.11.68 SOCHAPCMD\_12: SoCHAP Command Register 12

CSR Register Name: SOCHAPCMD_12: SoCHAP Command Register 12				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5108	Offset End: 510B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_12: SoCHAP Command Register 12					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5108	Offset End: 510B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_12: SoCHAP Command Register 12					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5108	Offset End: 510B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.69 SOCHAPEV\_12: SoCHAP Events Register 12

CSR Register Name: SOCHAPEV_12: SoCHAP Events Register 12					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 510C	Offset End: 510F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_12: SoCHAP Events Register 12					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 510C	Offset End: 510F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	INCOCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	INCEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	INCE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.70 SOCHAPSTAT\_12: SoCHAP Status Register 12

CSR Register Name: SOCHAPSTAT_12: SoCHAP Status Register 12					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5110	Offset End: 5113
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

<b>CSR Register Name: SOCHAPSTAT_12: SoCHAP Status Register 12</b>				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5110
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.
23:0	RO	0	Reserved	SOCHAPSTAT Reserved

#### 14.11.71 SOCHAPDATA\_12: SoCHAP Data Register 12

<b>CSR Register Name: SOCHAPDATA_12: SoCHAP Data Register 12</b>				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 5114
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 12 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.

### 14.11.72 SOCHAPCMD\_13: SoCHAP Command Register 13

CSR Register Name: SOCHAPCMD_13: SoCHAP Command Register 13				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5118	Offset End: 511B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_13: SoCHAP Command Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5118	Offset End: 511B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_13: SoCHAP Command Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5118	Offset End: 511B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.73 SOCHAPEV\_13: SoCHAP Events Register 13

CSR Register Name: SOCHAPEV_13: SoCHAP Events Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 511C	Offset End: 511F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_13: SoCHAP Events Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 511C	Offset End: 511F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.74 SOCHAPSTAT\_13: SoCHAP Status Register 13

CSR Register Name: SOCHAPSTAT_13: SoCHAP Status Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5120	Offset End: 5123
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

CSR Register Name: SOCHAPSTAT_13: SoCHAP Status Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5120	Offset End: 5123
Bits	Access	Default	Label	Bit Description	
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.	
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	
23:0	RO	0	Reserved	SOCHAPSTAT Reserved	

#### 14.11.75 SOCHAPDATA\_13: SoCHAP Data Register 13

CSR Register Name: SOCHAPDATA_13: SoCHAP Data Register 13					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5124	Offset End: 5127
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 13 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.	



## 14.11.76 SOCHAPCMD\_14: SoCHAP Command Register 14

CSR Register Name: SOCHAPCMD_14: SoCHAP Command Register 14				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5128	Offset End: 512B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition:  000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_14: SoCHAP Command Register 14				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 512B
Bits	Access	Default	Label	Bit Description
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.

CSR Register Name: SOCHAPCMD_14: SoCHAP Command Register 14				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 512B
Bits	Access	Default	Label	Bit Description
15	RO	0	Reserved	SOCHAPCMD Reserved
14:13	RW	0	Reserved	SOCHAPCMD Reserved
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.
11:10	RW	0	Reserved	SOCHAPCMD Reserved
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh

#### 14.11.77 SOCHAPEV\_14: SoCHAP Events Register 14

CSR Register Name: SOCHAPEV_14: SoCHAP Events Register 14				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 512C
Bits	Access	Default	Label	Bit Description
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.
30	RW	0	Reserved	SOCHAPEV Reserved
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.



CSR Register Name: SOCHAPEV_14: SoCHAP Events Register 14					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 512C	Offset End: 512F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.78 SOCHAPSTAT\_14: SoCHAP Status Register 14

CSR Register Name: SOCHAPSTAT_14: SoCHAP Status Register 14					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5130	Offset End: 5133
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

CSR Register Name: SOCHAPSTAT_14: SoCHAP Status Register 14					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5130	Offset End: 5133
Bits	Access	Default	Label	Bit Description	
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.	
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	
23:0	RO	0	Reserved	SOCHAPSTAT Reserved	

#### 14.11.79 SOCHAPDATA\_14: SoCHAP Data Register 14

CSR Register Name: SOCHAPDATA_14: SoCHAP Data Register 14					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5134	Offset End: 5137
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 14 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.	

### 14.11.80 SOCHAPCMD\_15: SoCHAP Command Register 15

CSR Register Name: SOCHAPCMD_15: SoCHAP Command Register 15				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 5138	Offset End: 513B	
Bits	Access	Default	Label	Bit Description
31:30	RO	0	Reserved	SOCHAPCMD Reserved
29:28	RW	0	Reserved	SOCHAPCMD Reserved
27	RW	0	PE	<b>PMI Enable:</b> 0: PMIout is not asserted when the indicators are valid. 1: The PMIout signal is asserted when the conditions enabled via the indicator enable bits are true.
26	RW	0	OUIE	<b>Overflow/Underflow Indicator Enable:</b> 0: No indication provided when a counter overflow or underflow occurs except for setting the Overflow/Underflow Indicator (OUI) status bit. 1: When the overflow/underflow condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
25	RW	0	CTIE	<b>Command Trigger Indicator Enable:</b> 0: No indication provided when a command is triggered except for setting the Command Trigger Indicator (CTI) status bit. 1: When the command trigger condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
24	RW	0	THIE	<b>Threshold Indicator Enable:</b> 0: No indication provided when threshold condition is true except for setting the Threshold Indicator (TI) status bit. 1: When the threshold condition is true, the counter will set the status bit and possibly assert the PMIout pin, which depending on SoC implementation, can be fed to another logic block or interrupt source.
23:21	RW	0	CC	<b>Condition Code:</b> This field contains the code that indicates what type of threshold compare is done between the counter and the data register. For all non-0 values of this field, the counter's data register will contain the threshold value. The outcome of this compare will generate a threshold event and potentially an interrupt if that capability is enabled. Bit 23 is for less than (<) Bit 22 is for equal (=) Bit 21 is for greater than (>) Select the proper bit mask for desired threshold condition: 000: False (no threshold compare) 001: Greater Than 010: Equal 011: Greater Than or Equal 100: Less Than 101: Not Equal 110: Less Than or Equal 111: True (always generate threshold event)



CSR Register Name: SOCHAPCMD_15: SoCHAP Command Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5138	Offset End: 513B
Bits	Access	Default	Label	Bit Description	
20	RW	0	SAC	<b>Select ALL Counters:</b> This bit has a slightly different function in Counter #0 compared to the other counters. When this bit is set in a given counter, then the opcode written to Counter #0, bits 19:16, is applied to all counters with bit 20 set. When this bit is cleared for a given counter, an opcode written to Counter #0, bits 19:16, does NOT apply to the given counter. If the SAC bit is not set when writing to Counter #0, the opcode is only applied to Counter #0, regardless of other counters' SAC bits. The rest of the SoCHAP command register is not affected by the setting of this bit. 0: The Counter #0 opcode is NOT applied to the counter associated with this command register. 1: The opcode IS applied to the counter associated with this command register. This means that every command register with a SAC bit set is written to with the same value that was written to Counter #0 command register. This is particularly handy for resetting all counters with a single command or starting or stopping all counters simultaneously.	
19:16	RW	0	OPCODE	<b>Opcode:</b> 0000: Stop. The corresponding counter does not count. 0001: Start. The corresponding counter begins counting. A counter increments by INCVAL+1 if the corresponding increment event occurs or decrements by DECVAL+1 if the corresponding decrement event occurs. All duration type events toggle every SoCHAP unit clock tick for which the event is true. The desired increment and decrement events must be selected before this command executes. 0010: Sample. The corresponding counter value is latched into the corresponding data register, which can then be accessed by reading the appropriate data register. Counter functionality is unaffected by a Sample Opcode. The counter continues to count without being reset. If the Condition Code is NOT False then the Data Register is not written when a sample takes place. In other words the Data Register will NOT be updated with counter value if the Condition Code is > 0. 0100: Reset. The corresponding counter is reset to 0000 0000h. The 32 bit wide counter allows 4 billion clock ticks or occurrences to be counted between sample commands. When the counter rolls over, the overflow status bit will be set in the corresponding status register. 0101: Restart. The corresponding counter resets, and begins counting again. This is essentially a Reset & Start command. This functionality will facilitate histogram creation by allowing an event to trigger to clear the counter and resume counting with no further intervention. 0110: Sample & Restart. The Sample command happens and is followed immediately by the Restart command. 1111: Preload. The corresponding counter is set to the value that is located in the associated Data Register. This facilitates rollover and overflow validation. The counter remains in the same state when preloaded. If the counter was counting before the preload was executed it will continue to count after the preload. It is softwares responsibility to ensure that the counter is in the desired state (example: execute stop command) prior to issuing a preload command. All others reserved.	

CSR Register Name: SOCHAPCMD_15: SoCHAP Command Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5138	Offset End: 513B
Bits	Access	Default	Label	Bit Description	
15	RO	0	Reserved	SOCHAPCMD Reserved	
14:13	RW	0	Reserved	SOCHAPCMD Reserved	
12	RW	0	TBV	<b>Trigger Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter counts every clock regardless of valid data. Since signals from VIS may span multiple clocks this mode is not recommended for triggers since multiple back to back triggers will likely occur.	
11:10	RW	0	Reserved	SOCHAPCMD Reserved	
9	RW	0	TTM	<b>Threshold Trigger Mode:</b> When this bit is set, if the operation of the trigger includes a sample (Sample, Sample and Restart opcodes), then the data value is only written if the threshold comparison is true. This mode can be used to store maximum and minimum values into the data register.	
8:0	RW	0	CT	<b>Command Trigger:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before executing the opcode. The previously programmed opcode continues to execute until this command trigger is detected. The ESC of 000h (the default value) is a special case. This special ESC causes the command to be triggered immediately upon a write to the command register. The special command trigger also causes the target command to be executed (triggered) once and only once. All other (non-0) ESCs cause the command to be re-executed every time the trigger is detected. Only one trigger can occur per VIS clock. If the trigger source clock is running faster than the VIS clock and more than one trigger event occurs per VIS clock, these additional triggers will be dropped. See the Event Tables for valid values. ESC values are limited to the range of 000-1FFh	

#### 14.11.81 SOCHAPEV\_15: SoCHAP Events Register 15

CSR Register Name: SOCHAPEV_15: SoCHAP Events Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 513C	Offset End: 513F
Bits	Access	Default	Label	Bit Description	
31	RW	0	DECOCE	<b>Decrement Occurrence Count Enable:</b> 0: Decrement Duration Count: the counter is decremented for each clock for which the decrement event signal is asserted logic high. 1: Decrement Occurrence Count: the counter is decremented each time a rising edge of the decrement event signal is detected.	
30	RW	0	Reserved	SOCHAPEV Reserved	
29	RW	0	DECEVOVR	<b>Decrement Event Override:</b> When this bit is set, the decrement event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	



CSR Register Name: SOCHAPEV_15: SoCHAP Events Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 513C	Offset End: 513F
Bits	Access	Default	Label	Bit Description	
28	RW	0	DEC BYP	<b>Decrement Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter decrements each clock regardless of valid data.	
27:25	RW	0	DEC VAL	<b>Decrement Counting Value:</b> By setting this, the decrement value used by the counter can be changed. Note, actual decrement value = DECVAL+1. 000 - Decrement by 1 001 - Decrement by 2 010 - Decrement by 3 ... 111 - Decrement by 8	
24:16	RW	0	DEC E	<b>Decrement Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before decrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh.	
15	RW	0	IN COCE	<b>Increment Occurrence Count Enable:</b> 0: Increment Duration Count: the counter is incremented for each clock for which the increment event signal is asserted logic high. 1: Increment Occurrence Count: the counter is incremented each time a rising edge of the increment event signal is detected.	
14	RW	0	Reserved	SOCHAPEV Reserved	
13	RW	0	IN CEVOVR	<b>Increment Event Override:</b> When this bit is set, the increment event is tied to logical 1. This in turn causes the counter to count the number clocks that occurred in the native domain of the signal.	
12	RW	0	INC BYP	<b>Increment Bypass Valid Protocol:</b> In this mode, the VIS valid protocol is not used. The counter increments each clock regardless of valid data.	
11:9	RW	0	INC VAL	<b>Increment Counting Value:</b> By setting this, the increment value used by the counter can be changed. Note, actual increment value = INCVAL + 1. 000 - Increment by 1 001 - Increment by 2 010 - Increment by 3 ... 111 - Increment by 8	
8:0	RW	0	IN CE	<b>Increment Event:</b> This field contains the Event Selection Code (ESC) that the unit will be required to detect before incrementing the associated counter. This field is only applicable for opcodes that put the counter in a state where it is counting. See the Event tables for valid values. This limits ESC codes to the range of 000-1FFh	



## 14.11.82 SOCHAPSTAT\_15: SoCHAP Status Register 15

CSR Register Name: SOCHAPSTAT_15: SoCHAP Status Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5140	Offset End: 5143
Bits	Access	Default	Label	Bit Description	
31	RW/1C	0	DCI	<b>Decrement Counter Indicator:</b> The decrement event was detected in the absence of the increment event. The counter may or may not have decremented depending on the counter state when the decrement event occurred.	
30	RW/1C	0	ICI	<b>Increment Counter Indicator:</b> The increment event was detected in the absence of the decrement event. The counter may or may not have incremented depending on the counter state when the increment event occurred.	
29	RO	0	CAI	<b>Counter Active Indicator:</b> 0: The associated counter is in a state that does NOT allow it to be incremented or decremented if the appropriate event(s) are detected. 1: The associated counter is in a state that allows it to be incremented or decremented if the appropriate event(s) are detected. This bit is updated every clock.	
28	RW/1C	0	IU	<b>In Use:</b> Software Mutex bit. After a RESET, a read to this bit returns a 0. After the first read, subsequent reads will return a 1. A write of a 1 to this bit will reset the next read value to 0. Writing a 0 to this bit has no effect. Software can poll this bit until it reads a 0, and will then own the usage of the corresponding counter. This bit has no other effect on any SoCHAP Counter registers, and is only used as a semaphore among various independent software threads that may need to utilize this performance counter. Software that reads this register, but does not intend to claim exclusive access, must write a 1 to this bit (if it reads a 0), in order to allow other software threads to claim it. This bit is particularly useful when supporting multi-threaded environments.	
27	RO	0	Reserved	SOCHAPSTAT Reserved	
26	RW/1C	0	OUI	<b>Overflow/Underflow Indicator:</b> 0: The associated 32 bit counter has NOT rolled over since the last time it was cleared. 1: The associated 32 bit counter HAS rolled over since the last time it was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	

CSR Register Name: SOCHAPSTAT_15: SoCHAP Status Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5140	Offset End: 5143
Bits	Access	Default	Label	Bit Description	
25	RW/1C	0	CTI	<b>Command Trigger Indicator:</b> 0: NO commands have been triggered since the last time this bit was cleared. 1: A command WAS triggered since the last time this bit was cleared. Software can use this bit to know that a command that was pending earlier has been triggered. Once a command has been triggered, another command can be triggered to execute. This bit is set by hardware, and cleared by software writing a 1 to it.	
24	RW/1C	0	THI	<b>Threshold Indicator:</b> 0: No threshold event has been generated since the last time this bit was cleared. 1: This counter generated a threshold event due to a true threshold condition compare since the last time this bit was cleared. This bit is set by hardware, and cleared by software writing a 1 to it.	
23:0	RO	0	Reserved	SOCHAPSTAT Reserved	

#### 14.11.83 SOCHAPDATA\_15: SoCHAP Data Register 15

CSR Register Name: SOCHAPDATA_15: SoCHAP Data Register 15					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5144	Offset End: 5147
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	CNTVAL	<b>Counter Value:</b> Contains either duration (number of clock ticks high) or occurrences (number of rising edges) contained in SoCHAP event counter 15 at time of sampling. The register is programmed to contain the threshold value that will be compared to the value in the event counter when non-0 condition codes have been selected.	



#### 14.11.84 SOCHAPPKTCTRL: SoCHAP Packet Control Register

CSR Register Name: SOCHAPPKTCTRL: SoCHAP Packet Control Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 5148
Bits	Access	Default	Label	Bit Description
31:16	RW	0	PKTTYPESEL	<b>Packet Type Select.</b> Each bit location corresponds to a SoCHAP counter and specifies the type field value to be written in each of the counter's output packets. The encoding for each bit is as follows: 0: Don't timestamp this counter's packets. 1: Timestamp all packets for this counter
15:0	RW	0	SPLCNTREN	<b>Each bit location corresponds to a SoCHAP counter to be sampled when its trigger asserts if the SoCHAP storeen is asserted.</b> 0: Means the corresponding counter will not be packetized and sent to GTH 1: Means the corresponding counters data register, OUI field, and THI field will be packetized and sent to GTH

#### 14.11.85 SOCHAPPKTCAP: SoCHAP Packet Capabilities Register

CSR Register Name: SOCHAPPKTCAP: SoCHAP Packet Capabilities Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 514C
Bits	Access	Default	Label	Bit Description
4:0	RO	2	MasterID	<b>SoCHAP_MASTERID parameter value.</b> Specifies the STP Master number assigned to the SoCHAP unit.

## 14.12 ODLA Registers

### 14.12.1 ODLA Registers Summary

ODLA Registers			
Offset Start	Offset End	Symbol	Register Name/Function
6000	6003	ODLACAP	ODLA Capability Register
6004	6007	OSMC	ODLA Storage Mode Control
6010	6013	TTSS	Timestamp Trigger Snap Shot
6014	6017	OSTAT	ODLA Status Register
6018	601B	ARBGNTCRED	Arbiter Grant Credit Register
602C	602F	TTSSEXT	Timestamp Trigger Snap Shot Extended Register
6030	6033	CTSS	Current Timestamp Snap Shot Register
6034	6037	CTSSEXT	Current Timestamp Snap Shot Extended Register
6080	6083	VCAPCTL	VISA Capture Control Register
6084	6087	VCAPREG_0	VISA Capture Register 0
6100	6103	LNSTORECTRL_0	Lane Storage Control Register 0
6104	6107	LNSTORECTRL_1	Lane Storage Control Register 1
6108	610B	LNSTORECTRL_2	Lane Storage Control Register 2
610C	610F	LNSTORECTRL_3	Lane Storage Control Register 3
6180	6183	LNFTLCTL_0	Lane Filter Control Register 0
6184	6187	LNFTLCTL_1	Lane Filter Control Register 1
6188	618B	LNFTLCTL_2	Lane Filter Control Register 2
618C	618F	LNFTLCTL_3	Lane Filter Control Register 3

### 14.12.2 ODLACAP: ODLA Capability Register

CSR Register Name: ODLACAP: ODLA Capability Register					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 6000	Offset End: 6003
Bits	Access	Default	Label	Bit Description	
31:28	RO	1	MasterID	ODLA_MASTERID parameter value.	
27:26	RO	0	GblBufDepthVal	<b>This bit field specifies the buffer depth of each global packet resource buffer.</b> 00: Depth is 4 packets 01: Depth is 5 packets 10: Depth is 6 packets 11: Depth is 7 packets	
25:14	RO	0	Reserved	ODLACAP Reserved	
13:12	RO	2	LaneBufDepthVal	<b>This bit field identifies the depth of the packetization buffers in the ODLA controller.</b> 00: Depth is 2 bytes 01: Depth is 3 bytes 10: Depth is 4 bytes 11: Reserved.	
11:5	RO	0	Reserved	ODLACAP Reserved	
4:0	RO	4	NumOfLanesVal	<b>Identifies the number of lanes that the ODLA controller</b> <b>[0h - 1h]:</b> Not applicable. A minimum of 2 lanes is necessary for ODLA to operate. <b>[2h]:</b> Number of lanes is equal to 2 <b>[3h]:</b> Number of lanes is equal to 3 <b>[4h]:</b> Number of lanes is equal to 4. A typical implementation number of lanes is 4. <b>---</b> <b>[Fh]:</b> Number of lanes is equal to 15 <b>[10h]:</b> Number of lanes is equal to 16 <b>[11-1Fh]:</b> Reserved, these values are not allowed.	



## 14.12.3 OSMC: ODLA Storage Mode Control

CSR Register Name: OSMC: ODLA Storage Mode Control					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 6004	Offset End: 6007
Bits	Access	Default	Label	Bit Description	
31	RW	0	TSCSrcSel	<b>Timestamp counter source select.</b> This field specifies the source for the timestamp value written to timestamp packets. 0: Internal timestamp counter 1: External timestamp counter	
30	RW	0	DeepCompDis	<b>Deep Compression Disable.</b> When the compressed count (CC) exceeds the maximum 8-bit value of 255, deep compression may be requested from the global logic. When deep compression is granted, the lane can enter deep compression and discontinue sending lane packets. Setting this bit to 1 will disable deep compression requests for all lanes which results in lanes storing data samples and compression counts in data packets instead of entering deep compression. 0: Deep compression is enabled 1: Deep compression is disabled, CC-sample pairs will continue been stored.	
29:28	RO	0	Reserved	OSMC Reserved	
27	RW	0	SampleTS	<b>Sample Timestamp.</b> Writing 1b1 causes the current timestamp value to be written to the CTSS and CTSSEXT registers so that it can be read by software. This bit is automatically cleared one clock after being written to remove the need for a separate write to clear it before getting subsequent snapshot updates. Writing zero has no effect.	
26	RO	0	Reserved	OSMC Reserved	
25:23	RW	3	TSinjRate	<b>Timestamp injection rate.</b> This field specifies the maximum number of ODLA clocks that can occur between global timestamp injections by configuring a periodic timestamp injection counter. When the counter reaches the specified value, a global timestamp injection will occur and the counter will be reset. Global timestamp injections occurring for reasons other than the counter reaching its terminal count value will also reset the counter. This provides a minimum global timestamp injection rate. 000: Do not inject additional global timestamps. Normal ODLA operation will insert timestamps when required. 001: Inject a global timestamp at least every 1k ODLA clocks. 010: Inject a global timestamp at least every 2k ODLA clocks. ... 111: Inject a global timestamp at least every 64k ODLA clocks.	
22	RW	0	ClkOffsetPktEn	<b>Clock Offset Packet Enable.</b> This bit allows for the creation of Clock Offset Packets on global timestamp injections. 0: Disable Clock Offset Packets 1: Enable Clock Offset Packets	
21	RO	0	Reserved	OSMC Reserved	

<b>CSR Register Name:</b> OSMC: ODLA Storage Mode Control				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 6004
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
20	RW	0	ClkOffDetDis	<b>Clock-Off Detection Disable.</b> This bit disables detection of a missing clock on all the lanes while global storage is enabled. 0: Clock-Off detection enabled. 1: Clock-Off detection disabled.
19	RO	0	Reserved	OSMC Reserved
18	RW	0	CompEn	<b>Compression Enable.</b> This bit enables store-on-change compression. 0: Compression is disabled 1: Compression is enabled
17:15	RW	0	LatAdj	<b>Latency Adjustment.</b> The ODLA control block and the Intel(R) Trace Hub trigger unit must be aligned for the ODLA to be effective. Therefore, the Intel(R) TH trigger unit may require additional latency states. Maximum number is defined by LATENCY parameter, $0 < \text{LATENCY} \leq 7$ . LatAdj=0 will mean no additional latency states, data will be captured on the same trigger clock cycle. If LatAdj=1, data one clock earlier than the trigger will be captured, if LatAdj=2, two clocks earlier and so on. 000: Zero additional ODLA latency clocks. 001: One additional ODLA latency clocks. : 111: Seven additional ODLA latency clocks.
14	RW	0	SwRst	<b>Software Reset.</b> When this is asserted the ODLA control logic is forced into reset. It remains in reset until this bit is cleared. Setting this bit will not reset the configuration registers or the Global Timestamp Counter. Only a hardware reset will reset the Global Timestamp Counter and clear some of the configuration registers.
13	RW	0	SwSusp	<b>Software Suspend.</b> This bit ends a data capture that has not finished yet. If some data was stored, SW Suspend will allow OSMC.DataReady get set and data could be extracted. 0: Normal ODLA operation. 1: ODLA operation suspended.
12:0	RO	0	Reserved	OSMC Reserved

#### 14.12.4 TTSS: Timestamp Trigger Snap Shot

<b>CSR Register Name:</b> TTSS: Timestamp Trigger Snap Shot				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 6010
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RW/1C	0	TrigTsVal	Trigger Timestamp Snapshot DWord Value.

#### 14.12.5 OSTAT: ODLA Status Register

CSR Register Name: OSTAT: ODLA Status Register					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 6014	Offset End: 6017
Bits	Access	Default	Label	Bit Description	
31:5	RO	0	Reserved	OSTAT Reserved	
4	RO/V	0	TsOvrFlw	<b>Time stamp Overflow.</b> This bit indicates that an overflow of the entire time stamp counter has occurred. 0: no overflow detected 1: An overflow has occurred.	
3	RO/V	0	WeStat	<b>ODLA Write Enable Status.</b> This bit indicates that at least one output write enable assertion has occurred to trace buffer during the current capture cycle. 0: Write Enable has not asserted. 1: Write Enable has asserted at least once. Only the first assertion is captured. There may have been more than one assertion.	
2	RO/V	0	Reserved	OSTAT Reserved	
1	RO/V	0	TrigStat	<b>ODLA Trigger Status.</b> This bit indicates that at least one trigger assertion has occurred. 0: Trigger has not asserted. 1: Trigger has asserted at least once. There may have been more than one assertion.	
0	RO/V	0	DataReady	<b>ODLA Data Ready status.</b> This bit indicates that the capture cycle is complete and debug data is ready for unload. Should be driven high whenever storeen is low and the content of all packet buffers are empty 0: ODLA disabled or enabled but capture cycle is not completed yet. 1: ODLA capture cycle completed.	

#### 14.12.6 ARBGNTCRED: Arbiter Grant Credit Register

CSR Register Name: ARBGNTCRED: Arbiter Grant Credit Register				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 6018	Offset End: 601B	
Bits	Access	Default	Label	Bit Description
31:30	RW	0	GntCreditsLn_15	<b>Grant Credits Lane 15.</b> GntCreditsLn[0] for details.
29:28	RW	0	GntCreditsLn_14	<b>Grant Credits Lane 14.</b> GntCreditsLn[0] for details.
27:26	RW	0	GntCreditsLn_13	<b>Grant Credits Lane 13.</b> GntCreditsLn[0] for details.
25:24	RW	0	GntCreditsLn_12	<b>Grant Credits Lane 12.</b> GntCreditsLn[0] for details.
23:22	RW	0	GntCreditsLn_11	<b>Grant Credits Lane 11.</b> GntCreditsLn[0] for details.
21:20	RW	0	GntCreditsLn_10	<b>Grant Credits Lane 10.</b> GntCreditsLn[0] for details.
19:18	RW	0	GntCreditsLn_9	<b>Grant Credits Lane 9.</b> GntCreditsLn[0] for details.
17:16	RW	0	GntCreditsLn_8	<b>Grant Credits Lane 8.</b> GntCreditsLn[0] for details.
15:14	RW	0	GntCreditsLn_7	<b>Grant Credits Lane 7.</b> GntCreditsLn[0] for details.
13:12	RW	0	GntCreditsLn_6	<b>Grant Credits Lane 6.</b> GntCreditsLn[0] for details.
11:10	RW	0	GntCreditsLn_5	<b>Grant Credits Lane 5.</b> GntCreditsLn[0] for details.
9:8	RW	0	GntCreditsLn_4	<b>Grant Credits Lane 4.</b> GntCreditsLn[0] for details.
7:6	RW	0	GntCreditsLn_3	<b>Grant Credits Lane 3.</b> GntCreditsLn[0] for details.
5:4	RW	0	GntCreditsLn_2	<b>Grant Credits Lane 2.</b> GntCreditsLn[0] for details.
3:2	RW	0	GntCreditsLn_1	<b>Grant Credits Lane 1.</b> GntCreditsLn[0] for details.
1:0	RW	0	GntCreditsLn_0	<b>Grant Credits for Lane 0.</b> 00: No back-to-back grants. The arbiter will grant one request at a time for this lane. 01: One back-to-back grant. The arbiter will grant one request and if there are more packets in the buffer, it will grant one more back-to-back request for this lane. 10: Two back-to-back grants. 11: Three back-to-back grants.



#### 14.12.7 TTSSEXT: Timestamp Trigger Snap Shot Extended Register

CSR Register Name: TTSSEXT: Timestamp Trigger Snap Shot Extended Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 602C	Offset End: 602F
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	Reserved	TTSSEXT Reserved	
15:8	RW/1C	0	TrigTsByt6	<b>Trigger Timestamp Snapshot High DWord Value.</b> This field contains the value of the upper timestamp at the first trigger assertion. (timestamp[47:40]) If multiple trigger assertions occurred, it will contain the value of the upper timestamp at the 1st trigger.	
7:0	RW/1C	0	TrigTsByt5	<b>Trigger Timestamp Snapshot High DWord Value.</b> This field contains the value of the upper timestamp at the first trigger assertion (timestamp[39:32]). If multiple trigger assertions occurred, it will contain the value of the upper timestamp at the 1st trigger.	

#### 14.12.8 CTSS: Current Timestamp Snap Shot Register

CSR Register Name: CTSS: Current Timestamp Snap Shot Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 6030	Offset End: 6033
Bits	Access	Default	Label	Bit Description	
31:0	RW/1C	0	CurTsVal	<b>Current Timestamp Snapshot DWord Value.</b> Updated with current timestamp value whenever OSMC register bit SampleTs is written with a 1b1	

#### 14.12.9 CTSSEXT: Current Timestamp Snap Shot Extended Register

CSR Register Name: CTSSEXT: Current Timestamp Snap Shot Extended Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 6034	Offset End: 6037
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	Reserved	CTSSEXT Reserved	
15:0	RW/1C	0	CurTsValExt	Upper 2 bytes of Current Timestamp Snapshot Value	

#### 14.12.10 VCAPCTL: VISA Capture Control Register

CSR Register Name: VCAPCTL: VISA Capture Control Register					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 6080	Offset End: 6083
Bits	Access	Default	Label	Bit Description	
31:4	RO	0	Reserved	VCAPCTL Reserved	

<b>CSR Register Name:</b> VCAPCTL: VISA Capture Control Register					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 6080	<b>Offset End:</b> 6083
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
3	RW	0	VcapClr	<b>VIS Clear capture register.</b> This bit will clear all bits in all byte lanes of the VCAPREG[3:0]. 0: Do not clear 1: Clear all	
2:1	RW	0	VcapMode	<b>VIS Capture Mode:</b> This bit determines the mode of operation. 00: Disable capture 01: Capture on trigger edge. Capture on the rising edge of the trigger source. VcapReg will continue to capture on each subsequent edge. 10: Capture on trigger level. Capture continuously when the trigger is logic 1. VcapReg will continue to capture on each subsequent level. 11: Capture continuously upon entering mode (no trigger). The sticky VCAPREG registers will retain the value until cleared.	
0	RW	0	VcapEn	<b>VIS Capture Enable:</b> This bit enables the VIS capture DFV feature. 0: Disable VIS capture 1: Enable VIS capture	

#### 14.12.11 VCAPREG\_0: VISA Capture Register 0

<b>CSR Register Name:</b> VCAPREG_0: VISA Capture Register 0					
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> npk_rst_b		<b>Offset Start:</b> 6084	<b>Offset End:</b> 6087
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>	
31:24	RW/1C	0	VcapByte3	VIS Capture Value Byte3: description same as Byte0	
23:16	RW/1C	0	VcapByte2	VIS Capture Value Byte2: description same as Byte0	
15:8	RW/1C	0	VcapByte1	VIS Capture Value Byte1: description same as Byte0	
7:0	RW/1C	0	VcapByte0	<b>VIS Capture Value Byte0:</b> This field captures one byte on the input VIS bus at the Lakemore top level. All bytes operate with same mode either trigger based or saturated based. This bit field may be cleared by writing logic1 into the specific bits/byte or with a software clear in the control register.	

#### 14.12.12 LNSTORECTRL\_0: Lane Storage Control Register 0

CSR Register Name: LNSTORECTRL_0: Lane Storage Control Register 0					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 6100	Offset End: 6103
Bits	Access	Default	Label	Bit Description	
31:9	RO	0	Reserved	LNSTORECTRL Reserved	
8	RW	0	LaneEn	<p><b>Lane Enable.</b> This bit will enable this lane for arbiter control.            0: Disable. This lane will not be observed by the arbitration logic and no data from this lane will be stored by the ODLA controller            1: Enable. This lane will be arbitrated with other enabled lanes and its data consumed by the ODLA controller.</p>	
7:0	RW	1F	ClkOffComp	<p><b>Clock Off Compare.</b> This bit field contains the 8-bit match value used by the clock off detection compare logic to set the maximum number of ODLA clocks since the last lane valid. When the clock off counter matches the compare value, a global timestamp injection will occur to indicate the time that the lane's source clock turned off.            0: Means the corresponding bit must be logic 0 to match.            1: Means the corresponding bit must be logic 1 to match.</p>	

#### 14.12.13 LNSTORECTRL\_1: Lane Storage Control Register 1

CSR Register Name: LNSTORECTRL_1: Lane Storage Control Register 1					
Bar:	CSR_MTB_BAR	Reset:	npk_rst_b	Offset Start: 6104	Offset End: 6107
Bits	Access	Default	Label	Bit Description	
31:9	RO	0	Reserved	LNSTORECTRL Reserved	
8	RW	0	LaneEn	<p><b>Lane Enable.</b> This bit will enable this lane for arbiter control.            0: Disable. This lane will not be observed by the arbitration logic and no data from this lane will be stored by the ODLA controller            1: Enable. This lane will be arbitrated with other enabled lanes and its data consumed by the ODLA controller.</p>	
7:0	RW	1F	ClkOffComp	<p><b>Clock Off Compare.</b> This bit field contains the 8-bit match value used by the clock off detection compare logic to set the maximum number of ODLA clocks since the last lane valid. When the clock off counter matches the compare value, a global timestamp injection will occur to indicate the time that the lane's source clock turned off.            0: Means the corresponding bit must be logic 0 to match.            1: Means the corresponding bit must be logic 1 to match.</p>	

#### 14.12.14 LNSTORECTRL\_2: Lane Storage Control Register 2

CSR Register Name: LNSTORECTRL_2: Lane Storage Control Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 6108	Offset End: 610B
Bits	Access	Default	Label	Bit Description	
31:9	RO	0	Reserved	LNSTORECTRL Reserved	
8	RW	0	LaneEn	<p><b>Lane Enable.</b> This bit will enable this lane for arbiter control.            0: Disable. This lane will not be observed by the arbitration logic and no data from this lane will be stored by the ODLA controller            1: Enable. This lane will be arbitrated with other enabled lanes and its data consumed by the ODLA controller.</p>	
7:0	RW	1F	ClkOffComp	<p><b>Clock Off Compare.</b> This bit field contains the 8-bit match value used by the clock off detection compare logic to set the maximum number of ODLA clocks since the last lane valid. When the clock off counter matches the compare value, a global timestamp injection will occur to indicate the time that the lane's source clock turned off.            0: Means the corresponding bit must be logic 0 to match.            1: Means the corresponding bit must be logic 1 to match.</p>	

#### 14.12.15 LNSTORECTRL\_3: Lane Storage Control Register 3

CSR Register Name: LNSTORECTRL_3: Lane Storage Control Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 610C	Offset End: 610F
Bits	Access	Default	Label	Bit Description	
31:9	RO	0	Reserved	LNSTORECTRL Reserved	
8	RW	0	LaneEn	<p><b>Lane Enable.</b> This bit will enable this lane for arbiter control.            0: Disable. This lane will not be observed by the arbitration logic and no data from this lane will be stored by the ODLA controller            1: Enable. This lane will be arbitrated with other enabled lanes and its data consumed by the ODLA controller.</p>	
7:0	RW	1F	ClkOffComp	<p><b>Clock Off Compare.</b> This bit field contains the 8-bit match value used by the clock off detection compare logic to set the maximum number of ODLA clocks since the last lane valid. When the clock off counter matches the compare value, a global timestamp injection will occur to indicate the time that the lane's source clock turned off.            0: Means the corresponding bit must be logic 0 to match.            1: Means the corresponding bit must be logic 1 to match.</p>	

#### 14.12.16 LNFTLCTL\_0: Lane Filter Control Register 0

CSR Register Name: LNFTLCTL_0: Lane Filter Control Register 0				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 6180	Offset End: 6183
Bits	Access	Default	Label	Bit Description
31:25	RO	0	Reserved	LNFTLCTL Reserved
24	RW	0	LaneFltInv	<b>Lane Filter Match Invert.</b> This bit inverts the resulting match equation. 0: Do not invert the resultant match equation. 1: Invert the resultant match equation.
23:16	RW	0	LaneFltForce	<b>Lane Filter Force Value.</b> This bit field contains the 8-bit force value used by the lane filter logic to force the lane filter bits to zero. Per Bit Force Definition: 0: Means the corresponding bit is passed as-is through to the match/mask filter logic 1: Means the corresponding bit is forced to 0 prior to the match/mask filter logic
15:8	RW	0	LaneFltMask	<b>Lane Filter Mask Value.</b> This bit field contains the 8-bit mask value used by the lane filter logic to mask the LaneFltMatch value. Each bit in the mask field corresponds to a match bit in the LaneFltMatch. Per Bit Mask Definition: 0: Means the corresponding bit is not included in the match equation. 1: Means the corresponding bit is included in the match equation. Corner case conditions that affect the match equation output: 0x00 = This byte lane ignores the match value and all debug data is sent to the next stage. The filter logic is disabled. The LaneFltInv must be cleared to zero otherwise it is a programming error. 0xFF = This byte lane must match exactly to the match value to be of use.
7:0	RW	0	LaneFltMatch	<b>Lane Filter Match Value.</b> This bit field contains the 8-bit match value used by the lane filter logic to develop an overall match value for all the lanes. Each bit in the match field corresponds to a match bit in the LaneFltMatch. 0: Means the corresponding bit must be logic 0 to match. 1: Means the corresponding bit must be logic 1 to match.

### 14.12.17 LNFTLCTL\_1: Lane Filter Control Register 1

CSR Register Name: LNFTLCTL_1: Lane Filter Control Register 1				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 6184	Offset End: 6187
Bits	Access	Default	Label	Bit Description
31:25	RO	0	Reserved	LNFTLCTL Reserved
24	RW	0	LaneFltInv	<b>Lane Filter Match Invert.</b> This bit inverts the resulting match equation. 0: Do not invert the resultant match equation. 1: Invert the resultant match equation.
23:16	RW	0	LaneFltForce	<b>Lane Filter Force Value.</b> This bit field contains the 8-bit force value used by the lane filter logic to force the lane filter bits to zero. Per Bit Force Definition: 0: Means the corresponding bit is passed as-is through to the match/mask filter logic 1: Means the corresponding bit is forced to 0 prior to the match/mask filter logic
15:8	RW	0	LaneFltMask	<b>Lane Filter Mask Value.</b> This bit field contains the 8-bit mask value used by the lane filter logic to mask the LaneFltMatch value. Each bit in the mask field corresponds to a match bit in the LaneFltMatch. Per Bit Mask Definition: 0: Means the corresponding bit is not included in the match equation. 1: Means the corresponding bit is included in the match equation. Corner case conditions that affect the match equation output: 0x00 = This byte lane ignores the match value and all debug data is sent to the next stage. The filter logic is disabled. The LaneFltInv must be cleared to zero otherwise it is a programming error. 0xFF = This byte lane must match exactly to the match value to be of use.
7:0	RW	0	LaneFltMatch	<b>Lane Filter Match Value.</b> This bit field contains the 8-bit match value used by the lane filter logic to develop an overall match value for all the lanes. Each bit in the match field corresponds to a match bit in the LaneFltMatch. 0: Means the corresponding bit must be logic 0 to match. 1: Means the corresponding bit must be logic 1 to match.

### 14.12.18 LNFTLCTL\_2: Lane Filter Control Register 2

CSR Register Name: LNFTLCTL_2: Lane Filter Control Register 2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 6188	Offset End: 618B
Bits	Access	Default	Label	Bit Description
31:25	RO	0	Reserved	LNFTLCTL Reserved
24	RW	0	LaneFltInv	<b>Lane Filter Match Invert.</b> This bit inverts the resulting match equation. 0: Do not invert the resultant match equation. 1: Invert the resultant match equation.
23:16	RW	0	LaneFltForce	<b>Lane Filter Force Value.</b> This bit field contains the 8-bit force value used by the lane filter logic to force the lane filter bits to zero. Per Bit Force Definition: 0: Means the corresponding bit is passed as-is through to the match/mask filter logic 1: Means the corresponding bit is forced to 0 prior to the match/mask filter logic
15:8	RW	0	LaneFltMask	<b>Lane Filter Mask Value.</b> This bit field contains the 8-bit mask value used by the lane filter logic to mask the LaneFltMatch value. Each bit in the mask field corresponds to a match bit in the LaneFltMatch. Per Bit Mask Definition: 0: Means the corresponding bit is not included in the match equation. 1: Means the corresponding bit is included in the match equation. Corner case conditions that affect the match equation output: 0x00 = This byte lane ignores the match value and all debug data is sent to the next stage. The filter logic is disabled. The LaneFltInv must be cleared to zero otherwise it is a programming error. 0xFF = This byte lane must match exactly to the match value to be of use.
7:0	RW	0	LaneFltMatch	<b>Lane Filter Match Value.</b> This bit field contains the 8-bit match value used by the lane filter logic to develop an overall match value for all the lanes. Each bit in the match field corresponds to a match bit in the LaneFltMatch. 0: Means the corresponding bit must be logic 0 to match. 1: Means the corresponding bit must be logic 1 to match.

### 14.12.19 LNFTLCTL\_3: Lane Filter Control Register 3

CSR Register Name: LNFTLCTL_3: Lane Filter Control Register 3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 618C	Offset End: 618F
Bits	Access	Default	Label	Bit Description
31:25	RO	0	Reserved	.
24	RW	0	LaneFltInv	<b>Lane Filter Match Invert.</b> This bit inverts the resulting match equation. 0: Do not invert the resultant match equation. 1: Invert the resultant match equation.
23:16	RW	0	LaneFltForce	<b>Lane Filter Force Value.</b> This bit field contains the 8-bit force value used by the lane filter logic to force the lane filter bits to zero. Per Bit Force Definition: 0: Means the corresponding bit is passed as-is through to the match/mask filter logic 1: Means the corresponding bit is forced to 0 prior to the match/mask filter logic
15:8	RW	0	LaneFltMask	<b>Lane Filter Mask Value.</b> This bit field contains the 8-bit mask value used by the lane filter logic to mask the LaneFltMatch value. Each bit in the mask field corresponds to a match bit in the LaneFltMatch. Per Bit Mask Definition: 0: Means the corresponding bit is not included in the match equation. 1: Means the corresponding bit is included in the match equation. Corner case conditions that affect the match equation output: 0x00 = This byte lane ignores the match value and all debug data is sent to the next stage. The filter logic is disabled. The LaneFltInv must be cleared to zero otherwise it is a programming error. 0xFF = This byte lane must match exactly to the match value to be of use.
7:0	RW	0	LaneFltMatch	<b>Lane Filter Match Value.</b> This bit field contains the 8-bit match value used by the lane filter logic to develop an overall match value for all the lanes. Each bit in the match field corresponds to a match bit in the LaneFltMatch. 0: Means the corresponding bit must be logic 0 to match. 1: Means the corresponding bit must be logic 1 to match.



## 14.13 TAP Registers

### 14.13.1 TAP Registers Summary

TAP Registers			
Offset Start	Offset End	Symbol	Register Name/Function
C	C	SLVIDCODE	Slave TAP Identification Register
40	40	REQMSG	Request Message Register
41	41	RSPMSG	Response Message Register
42	42	CLKOVR	Clock Override Register
60	60	REQMSG	Request Message Register
61	61	RSPMSG	Response Message Register
62	62	CLKOVR	Clock Override Register

### 14.13.2 SLVIDCODE: Slave TAP Identification Register

CSR Register Name: SLVIDCODE: Slave TAP Identification Register				
Bar: n/a		Reset: trst_b		Offset Start: C
Bits	Access	Default	Label	Bit Description
31:0	RO	strap	SLVIDCODE	Slave ID Code

### 14.13.3 REQMSG: Request Message Register

CSR Register Name: REQMSG: Request Message Register				
Bar: n/a		Reset: trst_b		Offset Start: 40
Bits	Access	Default	Label	Bit Description
113:112	W+R	0	BAR	Base Address Register number (0, 1, 2, or 3) for register read/write requests
111:104	W+R	0	BE	Request data byte enables
103:40	W+R	0	REQDATA	<b>Request Data (write data)</b> . If accessing a 32-bit CSR, only the least significant 32 bits will be written
17:4	W+R	0	ADDR	Request address: the offset within the BAR
3:0	W+R	0	RSVD	Reserved for future use.
28	W+R	0	REQTYPE	Request type: 0001: Memory (Register) Read 0010: Memory (Register) Write 0100: PCI Configuration Space Read 1000: PCI Configuration Space Write

#### 14.13.4 RSPMSG: Response Message Register

CSR Register Name: RSPMSG: Response Message Register					
Bar: n/a		Reset: trst_b		Offset Start: 41	Offset End: 41
Bits	Access	Default	Label	Bit Description	
65:2	W+R	0	RSPDATA	<b>Response Data (read data) which is only valid if STAT is equal to 10b.</b> If reading a 32-bit CSR, the most significant 32 bits of this field have to be ignored	
65:2	W+R	0	RSPDATA	<b>Response Data (read data) which is only valid if STAT is equal to 10b.</b> If reading a 32-bit CSR, the most significant 32 bits of this field have to be ignored	
1:0	W+R	0	STAT	Status of the current request with the following encoding: 00: TAP Idle (TAP has not received UpdateDR for REQMSG – 0x30), or a request has been initiated but is still in progress 01: Non-Posted request completed successfully indicating that response data, if any, is not available in the RSP field 10: Posted request completed successfully indicating posted request has been claimed by its destination All others are reserved for future use	

#### 14.13.5 CLKOVR: Clock Override Register

CSR Register Name: CLKOVR: Clock Override Register					
Bar: n/a		Reset: trst_b		Offset Start: 42	Offset End: 42
Bits	Access	Default	Label	Bit Description	
3:0	W+R	0	CLKSEL	Clock select to select the clock used to initiate requests and return responses with the following encoding: 0000: IOSF-Primary clock (IPCLK) 0101: TAP Clock (TCK) All others are reserved for future use	

#### 14.13.6 REQMSG: Request Message Register

CSR Register Name: REQMSG: Request Message Register					
Bar: n/a		Reset: trst_b		Offset Start: 60	Offset End: 60
Bits	Access	Default	Label	Bit Description	
113:112	RO	0	BAR	Base Address Register number (0, 1, 2, or 3) for register read/write requests	
111:104	RO	0	BE	Request data byte enables	
103:40	RO	0	REQDATA	<b>Request Data (write data).</b> If accessing a 32-bit CSR, only the least significant 32 bits will be written	
17:4	RO	0	ADDR	Request address: the offset within the BAR	
3:0	RO	0	RSVD	Reserved for future use.	



CSR Register Name: REQMSG: Request Message Register					
Bar: n/a		Reset: trst_b		Offset Start: 60	Offset End: 60
Bits	Access	Default	Label	Bit Description	
28	RO	0	REQTYPE	Request type: 0001: Memory (Register) Read 0010: Memory (Register) Write 0100: PCI Configuration Space Read 1000: PCI Configuration Space Write	

#### 14.13.7 RSPMSG: Response Message Register

CSR Register Name: RSPMSG: Response Message Register					
Bar: n/a		Reset: trst_b		Offset Start: 61	Offset End: 61
Bits	Access	Default	Label	Bit Description	
65:2	RO	0	RSPDATA	<b>Response Data (read data) which is only valid if STAT is equal to 10b.</b> If reading a 32-bit CSR, the most significant 32 bits of this field have to be ignored	
65:2	RO	0	RSPDATA	<b>Response Data (read data) which is only valid if STAT is equal to 10b.</b> If reading a 32-bit CSR, the most significant 32 bits of this field have to be ignored	
1:0	RO	0	STAT	Status of the current request with the following encoding: 00: TAP Idle (TAP has not received UpdateDR for REQMSG – 0x30), or a request has been initiated but is still in progress 01: Non-Posted request completed successfully indicating that response data, if any, is not available in the RSP field 10: Posted request completed successfully indicating posted request has been claimed by its destination All others are reserved for future use	

#### 14.13.8 CLKOVR: Clock Override Register

CSR Register Name: CLKOVR: Clock Override Register					
Bar: n/a		Reset: trst_b		Offset Start: 62	Offset End: 62
Bits	Access	Default	Label	Bit Description	
3:0	RO	0	CLKSEL	Clock select to select the clock used to initiate requests and return responses with the following encoding: 0000: IOSF-Primary clock (IPCLK) 0101: TAP Clock (TCK) All others are reserved for future use	

## 14.14 VIS Event Recognizer Registers

### 14.14.1 VIS Event Recognizer Registers Summary

VIS Event Recognizer Registers			
Offset Start	Offset End	Symbol	Register Name/Function
07000	07003	VERCAP	VISA Event Recognizer Capabilities Register
07004	07007	VERCTL	VISA Event Recognizer Control Register
07008	0700B	VERSTAT	VISA Event Recognizer Status Register
07100	07103	EVRCCTL0	Event Recognizer Control Register 0
07104	07107	EVRCCTL1	Event Recognizer Control Register 1
07108	0710B	EVRCCTL2	Event Recognizer Control Register 2
0710C	0710F	EVRCCTL3	Event Recognizer Control Register 3
07110	07113	EVRCCTL4	Event Recognizer Control Register 4
07114	07117	EVRCCTL5	Event Recognizer Control Register 5
07118	0711B	EVRCCTL6	Event Recognizer Control Register 2
0711C	0711F	EVRCCTL7	Event Recognizer Control Register 7
07140	07143	EVRMCH0	Event Recognizer Match Register 0
07144	07147	EVRMCH1	Event Recognizer Match Register 1
07148	0714B	EVRMCH2	Event Recognizer Match Register 2
0714C	0714F	EVRMCH3	Event Recognizer Match Register 3
07150	07153	EVRMCH4	Event Recognizer Match Register 0
07154	07157	EVRMCH5	Event Recognizer Match Register 1
07158	0715B	EVRMCH6	Event Recognizer Match Register 2
0715C	0715F	EVRMCH7	Event Recognizer Match Register 3
07180	07183	EVRMSK0	Event Recognizer Mask Register 0
07184	07187	EVRMSK1	Event Recognizer Mask Register 1
07188	0718B	EVRMSK2	Event Recognizer Mask Register 2
0718C	0718F	EVRMSK3	Event Recognizer Mask Register 3
07190	07193	EVRMSK4	Event Recognizer Mask Register 0
07194	07197	EVRMSK5	Event Recognizer Mask Register 1
07198	0719B	EVRMSK6	Event Recognizer Mask Register 2
0719C	0719F	EVRMSK7	Event Recognizer Mask Register 3
071C0	071C3	EVRFBMCH0	Event Recognizer Feedback Match Register 0
071C4	071C7	EVRFBMCH1	Event Recognizer Feedback Match Register 1

VIS Event Recognizer Registers			
Offset Start	Offset End	Symbol	Register Name/Function
071C8	071CB	EVRFBMCH2	Event Recognizer Feedback Match Register 2
071CC	071CF	EVRFBMCH3	Event Recognizer Feedback Match Register 3
071D0	071D3	EVRFBMCH4	Event Recognizer Feedback Match Register 4
071D4	071D7	EVRFBMCH5	Event Recognizer Feedback Match Register 5
071D8	071DB	EVRFBMCH6	Event Recognizer Feedback Match Register 6
071DC	071DF	EVRFBMCH7	Event Recognizer Feedback Match Register 7
07200	07203	EVRFBMSK0	Event Recognizer Feedback Mask Register 0
07204	07207	EVRFBMSK1	Event Recognizer Feedback Mask Register 1
07208	0720B	EVRFBMSK2	Event Recognizer Feedback Mask Register 2
0720C	0720F	EVRFBMSK3	Event Recognizer Feedback Mask Register 3
07210	07213	EVRFBMSK4	Event Recognizer Feedback Mask Register 0
07214	07217	EVRFBMSK5	Event Recognizer Feedback Mask Register 1
07218	0721B	EVRFBMSK6	Event Recognizer Feedback Mask Register 2
0721C	0721F	EVRFBMSK7	Event Recognizer Feedback Mask Register 3

#### 14.14.2 VERCAP: VISA Event Recognizer Capabilities Register

CSR Register Name: VERCAP: VISA Event Recognizer Capabilities Register				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 07000	Offset End: 07003	
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:8	RO	10	VisaCImXbarWidth	
7:5	RO	0	Reserved	
4:0	RO	8	ResrcNum	

#### 14.14.3 VERCTL: VISA Event Recognizer Control Register

CSR Register Name: VERCTL: VISA Event Recognizer Control Register				
Bar: CSR_MTB_BAR	Reset: npk_rst_b	Offset Start: 07004	Offset End: 07007	
Bits	Access	Default	Label	Bit Description
31:2	RO	0	RSVD	
1	RW	0	EnableEvents	
0	RW	0	SwRst	



#### 14.14.4 VERSTAT: VISA Event Recognizer Status Register

CSR Register Name: VERSTAT: VISA Event Recognizer Status Register				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07008
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW/1C	0	VERStat	

#### 14.14.5 EVRCTL0: Event Recognizer Control Register 0

CSR Register Name: EVRCTL0: Event Recognizer Control Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07100
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	

#### 14.14.6 EVRCTL1: Event Recognizer Control Register 1

CSR Register Name: EVRCTL1: Event Recognizer Control Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07104
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	



#### 14.14.7 EVRCTL2: Event Recognizer Control Register 2

CSR Register Name: EVRCTL2: Event Recognizer Control Register 2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 07108	Offset End: 0710B
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	

#### 14.14.8 EVRCTL3: Event Recognizer Control Register 3

CSR Register Name: EVRCTL3: Event Recognizer Control Register 3				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 0710C	Offset End: 0710F
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	

#### 14.14.9 EVRCTL4: Event Recognizer Control Register 4

CSR Register Name: EVRCTL4: Event Recognizer Control Register 4				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 07110	Offset End: 07113
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	



#### 14.14.10 EVRCTL5: Event Recognizer Control Register 5

CSR Register Name: EVRCTL5: Event Recognizer Control Register 5				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 07114	Offset End: 07117
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	

#### 14.14.11 EVRCTL6: Event Recognizer Control Register 2

CSR Register Name: EVRCTL6: Event Recognizer Control Register 2				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 07118	Offset End: 0711B
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	

#### 14.14.12 EVRCTL7: Event Recognizer Control Register 7

CSR Register Name: EVRCTL7: Event Recognizer Control Register 7				
Bar: CSR_MTB_BAR	Reset: npk_rst_b		Offset Start: 0711C	Offset End: 0711F
Bits	Access	Default	Label	Bit Description
31:9	RO	0	RSVD	
8	RW	0	DelaySel	
7:3	RW	0	EvVldSel	
2	RW	0	FrcDvld	
1	RW	0	InvMatch	
0	RW	0	ANDmatch	



#### 14.14.13 EVRMCHO: Event Recognizer Match Register 0

CSR Register Name: EVRMCHO: Event Recognizer Match Register 0					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07140	Offset End: 07143
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	RSVD		
15:0	RW	0	EvMchVal		

#### 14.14.14 EVRMCH1: Event Recognizer Match Register 1

CSR Register Name: EVRMCH1: Event Recognizer Match Register 1					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07144	Offset End: 07147
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	RSVD		
15:0	RW	0	EvMchVal		

#### 14.14.15 EVRMCH2: Event Recognizer Match Register 2

CSR Register Name: EVRMCH2: Event Recognizer Match Register 2					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07148	Offset End: 0714B
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	RSVD		
15:0	RW	0	EvMchVal		

#### 14.14.16 EVRMCH3: Event Recognizer Match Register 3

CSR Register Name: EVRMCH3: Event Recognizer Match Register 3					
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0714C	Offset End: 0714F
Bits	Access	Default	Label	Bit Description	
31:16	RO	0	RSVD		
15:0	RW	0	EvMchVal		



#### 14.14.17 EVRMCH4: Event Recognizer Match Register 0

CSR Register Name: EVRMCH4: Event Recognizer Match Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07150
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMchVal	

#### 14.14.18 EVRMCH5: Event Recognizer Match Register 1

CSR Register Name: EVRMCH5: Event Recognizer Match Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07154
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMchVal	

#### 14.14.19 EVRMCH6: Event Recognizer Match Register 2

CSR Register Name: EVRMCH6: Event Recognizer Match Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07158
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMchVal	

#### 14.14.20 EVRMCH7: Event Recognizer Match Register 3

CSR Register Name: EVRMCH7: Event Recognizer Match Register 3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0715C
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMchVal	



#### 14.14.21 EVRMSK0: Event Recognizer Mask Register 0

CSR Register Name: EVRMSK0: Event Recognizer Mask Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07180
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	

#### 14.14.22 EVRMSK1: Event Recognizer Mask Register 1

CSR Register Name: EVRMSK1: Event Recognizer Mask Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07184
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	

#### 14.14.23 EVRMSK2: Event Recognizer Mask Register 2

CSR Register Name: EVRMSK2: Event Recognizer Mask Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07188
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	

#### 14.14.24 EVRMSK3: Event Recognizer Mask Register 3

CSR Register Name: EVRMSK3: Event Recognizer Mask Register 3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0718C
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	



#### 14.14.25 EVRMSK4: Event Recognizer Mask Register 0

CSR Register Name: EVRMSK4: Event Recognizer Mask Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07190
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	

#### 14.14.26 EVRMSK5: Event Recognizer Mask Register 1

CSR Register Name: EVRMSK5: Event Recognizer Mask Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07194
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	

#### 14.14.27 EVRMSK6: Event Recognizer Mask Register 2

CSR Register Name: EVRMSK6: Event Recognizer Mask Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07198
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	

#### 14.14.28 EVRMSK7: Event Recognizer Mask Register 3

CSR Register Name: EVRMSK7: Event Recognizer Mask Register 3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0719C
Bits	Access	Default	Label	Bit Description
31:16	RO	0	RSVD	
15:0	RW	0	EvMskVal	



#### 14.14.29 EVRFBMCH0: Event Recognizer Feedback Match Register 0

CSR Register Name: EVRFBMCH0: Event Recognizer Feedback Match Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071C0
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.30 EVRFBMCH1: Event Recognizer Feedback Match Register 1

CSR Register Name: EVRFBMCH1: Event Recognizer Feedback Match Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071C4
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.31 EVRFBMCH2: Event Recognizer Feedback Match Register 2

CSR Register Name: EVRFBMCH2: Event Recognizer Feedback Match Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071C8
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.32 EVRFBMCH3: Event Recognizer Feedback Match Register 3

CSR Register Name: EVRFBMCH3: Event Recognizer Feedback Match Register 3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071CC
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	



#### 14.14.33 EVRFBMCH4: Event Recognizer Feedback Match Register 4

CSR Register Name: EVRFBMCH4: Event Recognizer Feedback Match Register 4				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071D0
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.34 EVRFBMCH5: Event Recognizer Feedback Match Register 5

CSR Register Name: EVRFBMCH5: Event Recognizer Feedback Match Register 5				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071D4
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.35 EVRFBMCH6: Event Recognizer Feedback Match Register 6

CSR Register Name: EVRFBMCH6: Event Recognizer Feedback Match Register 6				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071D8
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.36 EVRFBMCH7: Event Recognizer Feedback Match Register 7

CSR Register Name: EVRFBMCH7: Event Recognizer Feedback Match Register 7				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 071DC
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMchVal	

#### 14.14.37 EVRFBMSK0: Event Recognizer Feedback Mask Register 0

CSR Register Name: EVRFBMSK0: Event Recognizer Feedback Mask Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07200
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

#### 14.14.38 EVRFBMSK1: Event Recognizer Feedback Mask Register 1

CSR Register Name: EVRFBMSK1: Event Recognizer Feedback Mask Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07204
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

#### 14.14.39 EVRFBMSK2: Event Recognizer Feedback Mask Register 2

CSR Register Name: EVRFBMSK2: Event Recognizer Feedback Mask Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07208
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

#### 14.14.40 EVRFBMSK3: Event Recognizer Feedback Mask Register 3

CSR Register Name: EVRFBMSK3: Event Recognizer Feedback Mask Register 3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0720C
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	



#### 14.14.41 EVRFBMSK4: Event Recognizer Feedback Mask Register 0

CSR Register Name: EVRFBMSK4: Event Recognizer Feedback Mask Register 0				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07210
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

#### 14.14.42 EVRFBMSK5: Event Recognizer Feedback Mask Register 1

CSR Register Name: EVRFBMSK5: Event Recognizer Feedback Mask Register 1				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07214
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

#### 14.14.43 EVRFBMSK6: Event Recognizer Feedback Mask Register 2

CSR Register Name: EVRFBMSK6: Event Recognizer Feedback Mask Register 2				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 07218
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

#### 14.14.44 EVRFBMSK7: Event Recognizer Feedback Mask Register 3

CSR Register Name: EVRFBMSK7: Event Recognizer Feedback Mask Register 3				
Bar: CSR_MTB_BAR		Reset: npk_rst_b		Offset Start: 0721C
Bits	Access	Default	Label	Bit Description
31:8	RO	0	RSVD	
7:0	RW	0	FdbkMskVal	

## 14.15 VIS Register Controller Registers

### 14.15.1 VIS Register Controller Registers Summary

VIS Register Controller Registers			
Offset Start	Offset End	Symbol	Register Name/Function
20000	20003	VISACTLADDR	VISA2 Controller Address/Index Register
20004	20007	VISACTLDATA	VISA2 Controller Data Register
20008	2000B	VISARPLYSEQ0	VISA2 Controller Replay Sequence Control Register 0
20040	20043	VISACTLCAP	VISA2 Controller Capabilities Register
20080	20083	VISACTLCLM0	VISACTLCLM0
20084	20087	VISACTLCLM1	VISACTLCLM1
20088	2008B	VISACTLCLM2	VISACTLCLM2
2008C	2008F	VISACTLCLM3	VISACTLCLM3
20090	20093	VISACTLCLM4	VISACTLCLM4
20094	20097	VISACTLCLM5	VISACTLCLM5
20098	2009B	VISACTLCLM6	VISACTLCLM6
2009C	2009F	VISACTLCLM7	VISACTLCLM7
200A0	200A3	VISACTLCLM8	VISACTLCLM8
200A4	200A7	VISACTLCLM9	VISACTLCLM9
22814	22817	VISACTLULM0	VISACTLULM0
22818	2281B	VISACTLULM1	VISACTLULM1
2281C	2281F	VISACTLULM2	VISACTLULM2
22820	22823	VISACTLULM3	VISACTLULM3
22824	22827	VISACTLULM4	VISACTLULM4
22828	2282B	VISACTLULM5	VISACTLULM5
2282C	2282F	VISACTLULM6	VISACTLULM6
22830	22833	VISACTLULM7	VISACTLULM7
22834	22837	VISACTLULM8	VISACTLULM8
22838	2283B	VISACTLULM9	VISACTLULM9
30000	30003	RRL_Data	VISA Replay RAM Lower
30004	30007	RSVD	VISA Replay RAM Upper
30008	3000B	RRL_Data	VISA Replay RAM Lower
3000C	3000F	RSVD	VISA Replay RAM Upper

VIS Register Controller Registers			
Offset Start	Offset End	Symbol	Register Name/Function
30010	30013	RRL_Data	VISA Replay RAM Lower
30014	30017	RSVD	VISA Replay RAM Upper
30018	3001B	RRL_Data	VISA Replay RAM Lower
3001C	3001F	RSVD	VISA Replay RAM Upper
30020	30023	RRL_Data	VISA Replay RAM Lower
30024	30027	RSVD	VISA Replay RAM Upper
30028	3002B	RRL_Data	VISA Replay RAM Lower
3002C	3002F	RSVD	VISA Replay RAM Upper
30030	30033	RRL_Data	VISA Replay RAM Lower
30034	30037	RSVD	VISA Replay RAM Upper
30038	3003B	RRL_Data	VISA Replay RAM Lower
3003C	3003F	RSVD	VISA Replay RAM Upper
30040	30043	RRL_Data	VISA Replay RAM Lower
30044	30047	RSVD	VISA Replay RAM Upper
30048	3004B	RRL_Data	VISA Replay RAM Lower
3004C	3004F	RSVD	VISA Replay RAM Upper
30050	30053	RRL_Data	VISA Replay RAM Lower
30054	30057	RSVD	VISA Replay RAM Upper
30058	3005B	RRL_Data	VISA Replay RAM Lower
3005C	3005F	RSVD	VISA Replay RAM Upper
30060	30063	RRL_Data	VISA Replay RAM Lower
30064	30067	RSVD	VISA Replay RAM Upper
30068	3006B	RRL_Data	VISA Replay RAM Lower
3006C	3006F	RSVD	VISA Replay RAM Upper
30070	30073	RRL_Data	VISA Replay RAM Lower
30074	30077	RSVD	VISA Replay RAM Upper
30078	3007B	RRL_Data	VISA Replay RAM Lower
3007C	3007F	RSVD	VISA Replay RAM Upper
30080	30083	RRL_Data	VISA Replay RAM Lower
30084	30087	RSVD	VISA Replay RAM Upper
30088	3008B	RRL_Data	VISA Replay RAM Lower
3008C	3008F	RSVD	VISA Replay RAM Upper

VIS Register Controller Registers			
Offset Start	Offset End	Symbol	Register Name/Function
30090	30093	RRL_Data	VISA Replay RAM Lower
30094	30097	RSVD	VISA Replay RAM Upper
30098	3009B	RRL_Data	VISA Replay RAM Lower
3009C	3009F	RSVD	VISA Replay RAM Upper
300A0	300A3	RRL_Data	VISA Replay RAM Lower
300A4	300A7	RSVD	VISA Replay RAM Upper
300A8	300AB	RRL_Data	VISA Replay RAM Lower
300AC	300AF	RSVD	VISA Replay RAM Upper
300B0	300B3	RRL_Data	VISA Replay RAM Lower
300B4	300B7	RSVD	VISA Replay RAM Upper
300B8	300BB	RRL_Data	VISA Replay RAM Lower
300BC	300BF	RSVD	VISA Replay RAM Upper
300C0	300C3	RRL_Data	VISA Replay RAM Lower
300C4	300C7	RSVD	VISA Replay RAM Upper
300C8	300CB	RRL_Data	VISA Replay RAM Lower
300CC	300CF	RSVD	VISA Replay RAM Upper
300D0	300D3	RRL_Data	VISA Replay RAM Lower
300D4	300D7	RSVD	VISA Replay RAM Upper
300D8	300DB	RRL_Data	VISA Replay RAM Lower
300DC	300DF	RSVD	VISA Replay RAM Upper
300E0	300E3	RRL_Data	VISA Replay RAM Lower
300E4	300E7	RSVD	VISA Replay RAM Upper
300E8	300EB	RRL_Data	VISA Replay RAM Lower
300EC	300EF	RSVD	VISA Replay RAM Upper
300F0	300F3	RRL_Data	VISA Replay RAM Lower
300F4	300F7	RSVD	VISA Replay RAM Upper
300F8	300FB	RRL_Data	VISA Replay RAM Lower
300FC	300FF	RSVD	VISA Replay RAM Upper
30100	30103	RRL_Data	VISA Replay RAM Lower
30104	30107	RSVD	VISA Replay RAM Upper
30108	3010B	RRL_Data	VISA Replay RAM Lower
3010C	3010F	RSVD	VISA Replay RAM Upper

VIS Register Controller Registers			
Offset Start	Offset End	Symbol	Register Name/Function
30110	30113	RRL_Data	VISA Replay RAM Lower
30114	30117	RSVD	VISA Replay RAM Upper
30118	3011B	RRL_Data	VISA Replay RAM Lower
3011C	3011F	RSVD	VISA Replay RAM Upper
30120	30123	RRL_Data	VISA Replay RAM Lower
30124	30127	RSVD	VISA Replay RAM Upper
30128	3012B	RRL_Data	VISA Replay RAM Lower
3012C	3012F	RSVD	VISA Replay RAM Upper
30130	30133	RRL_Data	VISA Replay RAM Lower
30134	30137	RSVD	VISA Replay RAM Upper
30138	3013B	RRL_Data	VISA Replay RAM Lower
3013C	3013F	RSVD	VISA Replay RAM Upper
30140	30143	RRL_Data	VISA Replay RAM Lower
30144	30147	RSVD	VISA Replay RAM Upper
30148	3014B	RRL_Data	VISA Replay RAM Lower
3014C	3014F	RSVD	VISA Replay RAM Upper
30150	30153	RRL_Data	VISA Replay RAM Lower
30154	30157	RSVD	VISA Replay RAM Upper
30158	3015B	RRL_Data	VISA Replay RAM Lower
3015C	3015F	RSVD	VISA Replay RAM Upper
30160	30163	RRL_Data	VISA Replay RAM Lower
30164	30167	RSVD	VISA Replay RAM Upper
30168	3016B	RRL_Data	VISA Replay RAM Lower
3016C	3016F	RSVD	VISA Replay RAM Upper
30170	30173	RRL_Data	VISA Replay RAM Lower
30174	30177	RSVD	VISA Replay RAM Upper
30178	3017B	RRL_Data	VISA Replay RAM Lower
3017C	3017F	RSVD	VISA Replay RAM Upper
30180	30183	RRL_Data	VISA Replay RAM Lower
30184	30187	RSVD	VISA Replay RAM Upper
30188	3018B	RRL_Data	VISA Replay RAM Lower
3018C	3018F	RSVD	VISA Replay RAM Upper

VIS Register Controller Registers			
Offset Start	Offset End	Symbol	Register Name/Function
30190	30193	RRL_Data	VISA Replay RAM Lower
30194	30197	RSVD	VISA Replay RAM Upper
30198	3019B	RRL_Data	VISA Replay RAM Lower
3019C	3019F	RSVD	VISA Replay RAM Upper
301A0	301A3	RRL_Data	VISA Replay RAM Lower
301A4	301A7	RSVD	VISA Replay RAM Upper
301A8	301AB	RRL_Data	VISA Replay RAM Lower
301AC	301AF	RSVD	VISA Replay RAM Upper
301B0	301B3	RRL_Data	VISA Replay RAM Lower
301B4	301B7	RSVD	VISA Replay RAM Upper
301B8	301BB	RRL_Data	VISA Replay RAM Lower
301BC	301BF	RSVD	VISA Replay RAM Upper
301C0	301C3	RRL_Data	VISA Replay RAM Lower
301C4	301C7	RSVD	VISA Replay RAM Upper
301C8	301CB	RRL_Data	VISA Replay RAM Lower
301CC	301CF	RSVD	VISA Replay RAM Upper
301D0	301D3	RRL_Data	VISA Replay RAM Lower
301D4	301D7	RSVD	VISA Replay RAM Upper
301D8	301DB	RRL_Data	VISA Replay RAM Lower
301DC	301DF	RSVD	VISA Replay RAM Upper
301E0	301E3	RRL_Data	VISA Replay RAM Lower
301E4	301E7	RSVD	VISA Replay RAM Upper
301E8	301EB	RRL_Data	VISA Replay RAM Lower
301EC	301EF	RSVD	VISA Replay RAM Upper
301F0	301F3	RRL_Data	VISA Replay RAM Lower
301F4	301F7	RSVD	VISA Replay RAM Upper
301F8	301FB	RRL_Data	VISA Replay RAM Lower
301FC	301FF	RSVD	VISA Replay RAM Upper
30200	30203	RRL_Data	VISA Replay RAM Lower
30204	30207	RSVD	VISA Replay RAM Upper
30208	3020B	RRL_Data	VISA Replay RAM Lower
3020C	3020F	RSVD	VISA Replay RAM Upper

VIS Register Controller Registers			
Offset Start	Offset End	Symbol	Register Name/Function
30210	30213	RRL_Data	VISA Replay RAM Lower
30214	30217	RSVD	VISA Replay RAM Upper
30218	3021B	RRL_Data	VISA Replay RAM Lower
3021C	3021F	RSVD	VISA Replay RAM Upper
30220	30223	RRL_Data	VISA Replay RAM Lower
30224	30227	RSVD	VISA Replay RAM Upper
30228	3022B	RRL_Data	VISA Replay RAM Lower
3022C	3022F	RSVD	VISA Replay RAM Upper
30230	30233	RRL_Data	VISA Replay RAM Lower
30234	30237	RSVD	VISA Replay RAM Upper
30238	3023B	RRL_Data	VISA Replay RAM Lower
3023C	3023F	RSVD	VISA Replay RAM Upper
30240	30243	RRL_Data	VISA Replay RAM Lower
30244	30247	RSVD	VISA Replay RAM Upper
30248	3024B	RRL_Data	VISA Replay RAM Lower
3024C	3024F	RSVD	VISA Replay RAM Upper
30250	30253	RRL_Data	VISA Replay RAM Lower
30254	30257	RSVD	VISA Replay RAM Upper
30258	3025B	RRL_Data	VISA Replay RAM Lower
3025C	3025F	RSVD	VISA Replay RAM Upper
30260	30263	RRL_Data	VISA Replay RAM Lower
30264	30267	RSVD	VISA Replay RAM Upper
30268	3026B	RRL_Data	VISA Replay RAM Lower
3026C	3026F	RSVD	VISA Replay RAM Upper
30270	30273	RRL_Data	VISA Replay RAM Lower
30274	30277	RSVD	VISA Replay RAM Upper
30278	3027B	RRL_Data	VISA Replay RAM Lower

#### 14.15.2 VISACTLADDR: VISA2 Controller Address/Index Register

CSR Register Name: VISACTLADDR: VISA2 Controller Address/Index Register					
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 20000	Offset End: 20003
Bits	Access	Default	Label	Bit Description	
31	RW/O	0	LockDirectAccess		
30	RW/O	0	LockRam		
29:28	RW	0	LinkSpeed		
27:26	RW	0	VisaCmd		
25	RW	0	GlobTrigEn		
24	RO/V	0	GlobalReplayPend		
23:16	RW	0	SramIndex		
15	RW	0	FreeRunEn		
14	RW	0	LockLockdown		
13:5	RW	0	UnitId		
4:0	RW	0	RegOffset		

#### 14.15.3 VISACTLDATA: VISA2 Controller Data Register

CSR Register Name: VISACTLDATA: VISA2 Controller Data Register					
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 20004	Offset End: 20007
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RegData		

#### 14.15.4 VISARPLYSEQ0: VISA2 Controller Replay Sequence Control Register 0

CSR Register Name: VISARPLYSEQ0: VISA2 Controller Replay Sequence Control Register 0					
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 20008	Offset End: 2000B
Bits	Access	Default	Label	Bit Description	
31:29	RO	0	Reserved		
28	RW	0	ReplayValid		
27	RW	0	ReplayForce		
26	RO	0	Reserved		
25	RO/V	0	ReplayInProg		

<b>CSR Register Name:</b> VISARPLYSEQ0: VISA2 Controller Replay Sequence Control Register 0				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> vrc_rst_b		<b>Offset Start:</b> 20008
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
24	RO/V	0	ReplayPend	
23:16	RW	0	ReplayTrigEn	
15:8	RW	0	SramStopIndex	
7:0	RW	0	SramStartIndex	

#### 14.15.5 VISACTLCAP: VISA2 Controller Capabilities Register

<b>CSR Register Name:</b> VISACTLCAP: VISA2 Controller Capabilities Register				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> vrc_rst_b		<b>Offset Start:</b> 20040
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31	RW/O	0	SECURE_WRITE_DIS	
30:24	RO	0	RSVD	
23:16	RO	4	LOCKDOWN_SIZE	
15:8	RO	50	REPLAY_RAM_SIZE	
7:4	RO	1	NUM_REPLAY_REGS	
3:0	RO	3	VERSION	

#### 14.15.6 VISACTLCLMO: VISACTLCLMO

<b>CSR Register Name:</b> VISACTLCLMO: VISACTLCLMO				
<b>Bar:</b> CSR_MTB_BAR		<b>Reset:</b> vrc_rst_b		<b>Offset Start:</b> 20080
<b>Bits</b>	<b>Access</b>	<b>Default</b>	<b>Label</b>	<b>Bit Description</b>
31:0	RO	0	VisaCLMData	



#### 14.15.7 VISACTLCLM1: VISACTLCLM1

CSR Register Name: VISACTLCLM1: VISACTLCLM1				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 20084
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.8 VISACTLCLM2: VISACTLCLM2

CSR Register Name: VISACTLCLM2: VISACTLCLM2				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 20088
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.9 VISACTLCLM3: VISACTLCLM3

CSR Register Name: VISACTLCLM3: VISACTLCLM3				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 2008C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.10 VISACTLCLM4: VISACTLCLM4

CSR Register Name: VISACTLCLM4: VISACTLCLM4				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 20090
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.11 VISACTLCLM5: VISACTLCLM5

CSR Register Name: VISACTLCLM5: VISACTLCLM5				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 20094
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	



#### 14.15.12 VISACTLCLM6: VISACTLCLM6

CSR Register Name: VISACTLCLM6: VISACTLCLM6				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 20098
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.13 VISACTLCLM7: VISACTLCLM7

CSR Register Name: VISACTLCLM7: VISACTLCLM7				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 2009C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.14 VISACTLCLM8: VISACTLCLM8

CSR Register Name: VISACTLCLM8: VISACTLCLM8				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 200A0
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.15 VISACTLCLM9: VISACTLCLM9

CSR Register Name: VISACTLCLM9: VISACTLCLM9				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 200A4
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaCLMData	

#### 14.15.16 VISACTLULMO: VISACTLULMO

CSR Register Name: VISACTLULMO: VISACTLULMO				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22814
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	



#### 14.15.17 VISACTLULM1:VISACTLULM1

CSR Register Name: VISACTLULM1:VISACTLULM1				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22818
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.18 VISACTLULM2:VISACTLULM2

CSR Register Name: VISACTLULM2:VISACTLULM2				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 2281C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.19 VISACTLULM3:VISACTLULM3

CSR Register Name: VISACTLULM3:VISACTLULM3				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22820
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.20 VISACTLULM4:VISACTLULM4

CSR Register Name: VISACTLULM4:VISACTLULM4				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22824
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.21 VISACTLULM5:VISACTLULM5

CSR Register Name: VISACTLULM5:VISACTLULM5				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22828
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	



#### 14.15.22 VISACTLULM6: VISACTLULM6

CSR Register Name: VISACTLULM6: VISACTLULM6				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 2282C
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.23 VISACTLULM7: VISACTLULM7

CSR Register Name: VISACTLULM7: VISACTLULM7				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22830
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.24 VISACTLULM8: VISACTLULM8

CSR Register Name: VISACTLULM8: VISACTLULM8				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22834
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.25 VISACTLULM9: VISACTLULM9

CSR Register Name: VISACTLULM9: VISACTLULM9				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 22838
Bits	Access	Default	Label	Bit Description
31:0	RO	0	VisaUlmData	

#### 14.15.26 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30000
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.27 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30004	Offset End: 30007
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.28 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30008	Offset End: 3000B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.29 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3000C	Offset End: 3000F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.30 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30010	Offset End: 30013
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.31 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30014	Offset End: 30017
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30014	Offset End: 30017
Bits	Access	Default	Label	Bit Description	
14:0	RW	0	RSVD		

#### 14.15.32 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30018	Offset End: 3001B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.33 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3001C	Offset End: 3001F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.34 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30020	Offset End: 30023
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.35 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30024	Offset End: 30027
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		



#### 14.15.36 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30028
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.37 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3002C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.38 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30030
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.39 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30034
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.40 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30038
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.41 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3003C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.42 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30040
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.43 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30044
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.44 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30048
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.45 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3004C	Offset End: 3004F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.46 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30050	Offset End: 30053
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.47 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30054	Offset End: 30057
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.48 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30058	Offset End: 3005B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.49 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3005C	Offset End: 3005F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3005C
Bits	Access	Default	Label	Bit Description
14:0	RW	0	RSVD	

#### 14.15.50 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30060
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.51 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30064
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.52 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30068
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.53 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3006C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	



#### 14.15.54 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30070
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.55 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30074
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.56 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30078
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.57 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3007C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.58 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30080
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.59 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30084	Offset End: 30087
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.60 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30088	Offset End: 3008B
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.61 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 3008C	Offset End: 3008F
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.62 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30090	Offset End: 30093
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.63 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30094	Offset End: 30097
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.64 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30098	Offset End: 3009B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.65 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3009C	Offset End: 3009F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.66 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300A0	Offset End: 300A3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.67 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300A4	Offset End: 300A7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300A4	Offset End: 300A7
Bits	Access	Default	Label	Bit Description	
14:0	RW	0	RSVD		

#### 14.15.68 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300A8	Offset End: 300AB
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.69 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300AC	Offset End: 300AF
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.70 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300B0	Offset End: 300B3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.71 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300B4	Offset End: 300B7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		



#### 14.15.72 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300B8
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.73 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300BC
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.74 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300C0
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.75 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300C4
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.76 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300C8
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.77 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 300CC	Offset End: 300CF
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.78 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 300D0	Offset End: 300D3
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.79 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 300D4	Offset End: 300D7
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.80 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 300D8	Offset End: 300DB
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.81 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300DC	Offset End: 300DF
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.82 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300E0	Offset End: 300E3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.83 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300E4	Offset End: 300E7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.84 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300E8	Offset End: 300EB
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.85 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 300EC	Offset End: 300EF
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300EC
Bits	Access	Default	Label	Bit Description
14:0	RW	0	RSVD	

#### 14.15.86 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300F0
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.87 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300F4
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.88 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300F8
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.89 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 300FC
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.90 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30100	Offset End: 30103
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.91 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30104	Offset End: 30107
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.92 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30108	Offset End: 3010B
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.93 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 3010C	Offset End: 3010F
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.94 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30110	Offset End: 30113
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.95 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30114	Offset End: 30117
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.96 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30118	Offset End: 3011B
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.97 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 3011C	Offset End: 3011F
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.98 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30120	Offset End: 30123
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.99 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30124	Offset End: 30127
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.100 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30128	Offset End: 3012B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.101 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3012C	Offset End: 3012F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.102 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30130	Offset End: 30133
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.103 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30134	Offset End: 30137
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30134	Offset End: 30137
Bits	Access	Default	Label	Bit Description	
14:0	RW	0	RSVD		

#### 14.15.104 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30138	Offset End: 3013B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.105 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3013C	Offset End: 3013F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.106 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30140	Offset End: 30143
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.107 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30144	Offset End: 30147
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		



#### 14.15.108 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30148
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.109 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3014C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.110 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30150
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.111 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30154
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.112 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30158
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.113 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 3015C	Offset End: 3015F
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.114 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30160	Offset End: 30163
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.115 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30164	Offset End: 30167
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.116 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b	Offset Start: 30168	Offset End: 3016B
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.117 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3016C	Offset End: 3016F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.118 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30170	Offset End: 30173
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.119 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30174	Offset End: 30177
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.120 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30178	Offset End: 3017B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.121 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3017C	Offset End: 3017F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3017C	Offset End: 3017F
Bits	Access	Default	Label	Bit Description	
14:0	RW	0	RSVD		

#### 14.15.122 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30180	Offset End: 30183
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.123 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30184	Offset End: 30187
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.124 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30188	Offset End: 3018B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.125 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3018C	Offset End: 3018F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		



#### 14.15.126 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30190	Offset End: 30193
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.127 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30194	Offset End: 30197
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.128 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30198	Offset End: 3019B
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.129 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3019C	Offset End: 3019F
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.130 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301A0	Offset End: 301A3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.131 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301A4	Offset End: 301A7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.132 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301A8	Offset End: 301AB
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.133 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301AC	Offset End: 301AF
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.134 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301B0	Offset End: 301B3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.135 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301B4	Offset End: 301B7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		



#### 14.15.136 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301B8
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.137 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301BC
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.138 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301C0
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.139 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301C4
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.140 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301C8
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.141 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301CC
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.142 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301D0
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.143 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301D4
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.144 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301D8
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.145 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301DC
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	



#### 14.15.146 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301E0	Offset End: 301E3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.147 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301E4	Offset End: 301E7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.148 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301E8	Offset End: 301EB
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.149 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301EC	Offset End: 301EF
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.150 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 301F0	Offset End: 301F3
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.151 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301F4	Offset End: 301F7
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.152 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301F8	Offset End: 301FB
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.153 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 301FC	Offset End: 301FF
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		

#### 14.15.154 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30200	Offset End: 30203
Bits	Access	Default	Label	Bit Description	
31:0	RW	0	RSVD		

#### 14.15.155 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper					
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30204	Offset End: 30207
Bits	Access	Default	Label	Bit Description	
31:15	RO	0	RSVD		
14:0	RW	0	RSVD		



#### 14.15.156 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30208
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.157 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3020C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.158 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30210
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.159 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30214
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.160 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30218
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.161 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3021C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.162 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30220
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.163 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30224
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.164 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30228
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.165 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3022C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	



#### 14.15.166 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 30230
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.167 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 30234
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.168 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 30238
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.169 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 3023C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.170 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar:	CSR_MTB_BAR	Reset:	vrc_rst_b	Offset Start: 30240
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.171 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30244
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.172 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30248
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.173 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3024C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.174 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30250
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.175 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30254
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	



#### 14.15.176 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30258
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.177 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 3025C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.178 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30260
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.179 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30264
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.180 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_rst_b		Offset Start: 30268
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	



#### 14.15.181 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 3026C
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.182 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30270
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	

#### 14.15.183 RSVD: VISA Replay RAM Upper

CSR Register Name: RSVD: VISA Replay RAM Upper				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30274
Bits	Access	Default	Label	Bit Description
31:15	RO	0	RSVD	
14:0	RW	0	RSVD	

#### 14.15.184 RRL\_Data: VISA Replay RAM Lower

CSR Register Name: RRL_Data: VISA Replay RAM Lower				
Bar: CSR_MTB_BAR		Reset: vrc_RST_B		Offset Start: 30278
Bits	Access	Default	Label	Bit Description
31:0	RW	0	RSVD	