**Title**

HW-SW SystemC Co-Simulation SoC Validation Platform

**Analysis Capability Report**

Contract No. 4200022968

by

 TU Braunschweig, Germany

January 25, 2012

**Document No.: IDA-SCSV-ACR-001**

**Table of Contents**

1 introduction 3

1.1 Purpose and Scope 3

1.2 Revisions 3

2 Configuration Analysis 4

2.1 AHBCTRL 4

2.2 APBCTRL 5

2.3 MCTRL 5

2.4 Generic Memory 6

2.5 MMU\_CACHE 6

2.5.1 Cache 6

2.5.2 Localram 7

2.5.3 MMU 7

2.6 GPTimer 7

2.7 IRQMP 8

2.8 SoCWire 8

3 Performance & Communication analysis 8

3.1 AHBCTRL 9

3.2 APBCTRL 9

3.3 MCTRL 9

3.4 Generic Memory 10

3.5 MMU\_CACHE 10

3.5.1 Cache 11

3.5.2 Localram 11

3.5.3 MMU 11

3.6 GPTimer 12

3.7 IRQMP 12

3.8 SoCWire 13

**Table of Tables**

Table 2 - Revisions of this document 3

# introduction

## Purpose and Scope

This document is the Analysis Capability Report of the SystemC Co-Simulation SoC Validation Platform (SoCRocket) and ist contained simulation models. It describes all features provided for the user concerning configuration and runtime analysis.

## Revisions

The following table will be updated during the course of the project.

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| 0.1 | 02/01/11 | Initial draft version |
| 0.2 | 23/01/11 | Configuration analysis / Performance analysis |
|  |  |  |
|  |  |  |

Table 1 - Revisions of this document

# Configuration Analysis

The platform infrastructure produces reports for the configuration of each simulation model in the system. The reports are generated in the constructor of the separate IPs. They are written to std::out, if the system is configured for verbosity-level 4 (info) or above.

## AHBCTRL

The configuration report is generated in the constructor of class ahbctrl. It displays all constructor parameters of the device. The example below shows the report for an AHBCTRL with priority arbitration (rrobin = 0), power monitoring enabled (pow\_mon = 1) and AT abstractions.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (ahbctrl): Info: \* Created AHBCTRL with following parameters:

@0 s /0 (ahbctrl): Info: \* ioaddr/iomask: fff/fff

@0 s /0 (ahbctrl): Info: \* cfgaddr/cfmask: ff0/ff0

@0 s /0 (ahbctrl): Info: \* rrobin: 0

@0 s /0 (ahbctrl): Info: \* split: 0

@0 s /0 (ahbctrl): Info: \* defmast: 0

@0 s /0 (ahbctrl): Info: \* ioen: 0

@0 s /0 (ahbctrl): Info: \* fixbrst: 0

@0 s /0 (ahbctrl): Info: \* fpnpen: 1

@0 s /0 (ahbctrl): Info: \* mcheck: 1

@0 s /0 (ahbctrl): Info: \* pow\_mon: 1

@0 s /0 (ahbctrl): Info: \* abstractionLayer (LT = 8 / AT = 4): 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The AHBCTRL additionally generates a report describing the decoder initialization. This report is produced in the start\_of\_simulation function of class ahbctrl (for verbosity >= 4).

The report lists the up to 16 slaves and up to 16 masters, which are bound to the AHBCTRL. Each of them may contain up to four sub-devices. Sub-devices are identified by a base register entry (BAR0-4) comprising the 12bit MSB address of the component (haddr) and the 12bit AHB address mask (hmask).

The example below shows a report for an AHBCTRL with one master (Testbench) and two slaves: APBCTRL, MCTRL. The MCTRL contains three sub-devices.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.ahbctrl): Info: \* DECODER INITIALIZATION

@0 s /0 (top.ahbctrl): Info: \* ----------------------

@0 s /0 (top.ahbctrl): Info: \* SLAVE name: top.apbctrl

@0 s /0 (top.ahbctrl): Info: \* BAR0 with MSB addr: 0x800 and mask: 0xfff

@0 s /0 (top.ahbctrl): Info: \* BAR1 not used.

@0 s /0 (top.ahbctrl): Info: \* BAR2 not used.

@0 s /0 (top.ahbctrl): Info: \* BAR3 not used.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.ahbctrl): Info: \* SLAVE name: top.mctrl

@0 s /0 (top.ahbctrl): Info: \* BAR0 with MSB addr: 0x0 and mask: 0xe00

@0 s /0 (top.ahbctrl): Info: \* BAR1 with MSB addr: 0x200 and mask: 0xe00

@0 s /0 (top.ahbctrl): Info: \* BAR2 with MSB addr: 0x400 and mask: 0xc00

@0 s /0 (top.ahbctrl): Info: \* BAR3 not used.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.ahbctrl): Info: \* Master name: top.testbench

@0 s /0 (top.ahbctrl): Info: \* BAR0 not used.

@0 s /0 (top.ahbctrl): Info: \* BAR1 not used.

@0 s /0 (top.ahbctrl): Info: \* BAR2 not used.

@0 s /0 (top.ahbctrl): Info: \* BAR3 not used.

@0 s /0 (top.ahbctrl): Info: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## APBCTRL

The configuration report for the APBCTRL is generated in the constructor of class apbctrl. It displays all constructor parameters of the device. The example below shows an APBCTRL in LT mode.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.apbctrl): Info: \* Created APBCTRL with following parameters:

@0 s /0 (top.apbctrl): Info: \* haddr/hmask: 800/fff

@0 s /0 (top.apbctrl): Info: \* mcheck: 1

@0 s /0 (top.apbctrl): Info: \* hindex: 1

@0 s /0 (top.apbctrl): Info: \* pow\_mon: 0

@0 s /0 (top.apbctrl): Info: \* ambaLayer (LT = 8/AT = 4): 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The APBCTRL additionally generates a report describing its decoder initialization. This report is produced in the start\_of\_simulation function of class apbctrl (for verbosity >= 4).

The report lists the up to 16 slaves, which are bound to the APBCTRL. Each of them is identified by a 12 bit segment address (paddr) and a 12 bit AMBA address mask (pmask). The example shows a report for an APBCTRL with one slave. The APB interface of the MCTRL is located at address/mask: 0x000/0xfff.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.apbctrl): Info: \* APB DECODER INITIALIZATION

@0 s /0 (top.apbctrl): Info: \* --------------------------

@0 s /0 (top.apbctrl): Info: \* Slave name: top.mctrl

@0 s /0 (top.apbctrl): Info: \* BAR with MSB addr: 0 and mask: fff

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## MCTRL

The configuration report for the memory controller unit (MCTRL) is generated in the constructor of class mctrl. It displays all constructor parameters of the device. The example below shows a MCTRL in AT mode with SDRAM enabled (sden = 1). The SD bus is 32 bits wide (sdbits). The special features of mobile RAM are disabled (mobile = 0).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.mctrl): Info: \* Created MCTRL with following parameters:

@0 s /0 (top.mctrl): Info: \* romasel: 28

@0 s /0 (top.mctrl): Info: \* sdrasel: 29

@0 s /0 (top.mctrl): Info: \* romaddr/rommask: 0/e00

@0 s /0 (top.mctrl): Info: \* ioaddr/iomask: 200/e00

@0 s /0 (top.mctrl): Info: \* ramaddr/rammask: 400/c00

@0 s /0 (top.mctrl): Info: \* paddr/pmask: 0/fff

@0 s /0 (top.mctrl): Info: \* wprot: 0

@0 s /0 (top.mctrl): Info: \* srbanks: 5

@0 s /0 (top.mctrl): Info: \* ram8: 1

@0 s /0 (top.mctrl): Info: \* ram16: 1

@0 s /0 (top.mctrl): Info: \* sepbus: 0

@0 s /0 (top.mctrl): Info: \* sdbits: 32

@0 s /0 (top.mctrl): Info: \* mobile: 0

@0 s /0 (top.mctrl): Info: \* sden: 1

@0 s /0 (top.mctrl): Info: \* hindex: 0

@0 s /0 (top.mctrl): Info: \* pindex: 0

@0 s /0 (top.mctrl): Info: \* pow\_mon: 0

@0 s /0 (top.mctrl): Info: \* abstractionLayer (LT = 8 / AT = 4): 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Generic Memory

The configuration memory for the generic memory is generated in the constructor of class GenericMemory. The generic memory can be configured to represent ROM, IO, SRAM and SDRAM storage. The example below shows the configuration report for a 512 MB SDRAM consisting of one memory bank with 16 columns.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.sdram): Info: \* Created GenericMemory with following parameters:

@0 s /0 (top.sdram): Info: \* device\_type (0-ROM, 1-IO, 2-SRAM, 3-SDRAM): 3

@0 s /0 (top.sdram): Info: \* banks: 1

@0 s /0 (top.sdram): Info: \* bsize: 20000000

@0 s /0 (top.sdram): Info: \* bit width: 32

@0 s /0 (top.sdram): Info: \* cols (SD only): 16

@0 s /0 (top.sdram): Info: \* pow\_mon: 0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## MMU\_CACHE

Depending on the configuration the MMU\_CACHE IP consists of multiple sub-components. Each of them generates a detailed configuration report on their own. Therefore, the configuration report of the MMU\_CACHE top-level module only provides structural information. The report is generated in the constructor of class mmu\_cache. The example shows a configuration with icache and dcache enabled. MMU and localrams (scratchpads) are disabled.

@0 s /0 (mmu\_cache): Info: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (mmu\_cache): Info: \* Created MMU\_CACHE in following configuration:

@0 s /0 (mmu\_cache): Info: \* ---------------------------------------------

@0 s /0 (mmu\_cache): Info: \* instruction cache enable (icen): 1

@0 s /0 (mmu\_cache): Info: \* data cache enable (dcen): 1

@0 s /0 (mmu\_cache): Info: \* mmu enable (mmu\_en): 0

@0 s /0 (mmu\_cache): Info: \* instruction scratchpad enable (ilram): 0

@0 s /0 (mmu\_cache): Info: \* data scratchpad enable (dlram): 0

@0 s /0 (mmu\_cache): Info: \* abstraction Layer (LT = 8 / AT = 4): 8

@0 s /0 (mmu\_cache): Info: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Cache

The configuration reports for the instruction cache and the data cache are generated by the constructor of class vectorcache. The report contains all relevant constructor parameters. Additional information is given concerning the size of the cache in kb, the number of bytes per cache line, the number of cache lines per set and the width of the cache tag in bits. The example below shows the configuration report for a data cache with 4 sets of 1kb:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (mmu\_cache.dcache): Info: \* Created cache memory with following parameters:

@0 s /0 (mmu\_cache.dcache): Info: \* ------------------------------------------

@0 s /0 (mmu\_cache.dcache): Info: \* mmu\_en: 0

@0 s /0 (mmu\_cache.dcache): Info: \* burst\_en: 0

@0 s /0 (mmu\_cache.dcache): Info: \* sets: 4

@0 s /0 (mmu\_cache.dcache): Info: \* setsize: 1

@0 s /0 (mmu\_cache.dcache): Info: \* setlock: 0

@0 s /0 (mmu\_cache.dcache): Info: \* linesize: 4

@0 s /0 (mmu\_cache.dcache): Info: \* repl (0-Direct Mapped, 1-LRU,   
 2-LRR, 3-RANDOM): 3

@0 s /0 (mmu\_cache.dcache): Info: \* ------------------------------------

@0 s /0 (mmu\_cache.dcache): Info: \* Size of each cache set 1 kb

@0 s /0 (mmu\_cache.dcache): Info: \* Bytes per line 16 (offset bits: 4)

@0 s /0 (mmu\_cache.dcache): Info: \* Number of cache lines per set 64   
 (index bits: 6)

@0 s /0 (mmu\_cache.dcache): Info: \* Width of cache tag in bits 22

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Localram

The configuration report for the localram is generated in the constructor of class localram. Because the localram is a very simple storage device, the report only contains two parameters, associated with starting address and capacity. The example shows the reports for an instruction scratchpad and a data scratchpad in default configuration.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (mmu\_cache.ilocalram): Info: \* Created localram with following parameters:

@0 s /0 (mmu\_cache.ilocalram): Info: \* ---------------------------------------

@0 s /0 (mmu\_cache.ilocalram): Info: \* lrstart (start address): 8e000000

@0 s /0 (mmu\_cache.ilocalram): Info: \* lrsize: 80000 bytes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (mmu\_cache.dlocalram): Info: \* Created localram with following parameters:

@0 s /0 (mmu\_cache.dlocalram): Info: \* ---------------------------------------

@0 s /0 (mmu\_cache.dlocalram): Info: \* lrstart (start address): 8f000000

@0 s /0 (mmu\_cache.dlocalram): Info: \* lrsize: 80000 bytes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### MMU

The MMU configuration report is generated in the constructor of class mmu. The report contains all the configuration parameters of the MMU. The example shows the report for a MMU with split TLBs (8x instructions, 8x data).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (mmu\_cache.mmu): Info: \* Created mmu with following parameters:

@0 s /0 (mmu\_cache.mmu): Info: \* --------------------------------------

@0 s /0 (mmu\_cache.mmu): Info: \* itlbnum: 8

@0 s /0 (mmu\_cache.mmu): Info: \* dtlbnum: 8

@0 s /0 (mmu\_cache.mmu): Info: \* tlb\_type (0 - split, 1 - shared): 0

@0 s /0 (mmu\_cache.mmu): Info: \* tlb\_rep: 1

@0 s /0 (mmu\_cache.mmu): Info: \* mmupgsz (0, 2-4kb, 3-8kb, 4-16kb, 5-32kb): 0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## GPTimer

The configuration report for the GPTimer is generated in the constructor of class gptimer. The report contains all the configuration parameters of the GPTimer. The example below shows the report for a GPTimer with 7 parallel counters and watchdog disabled.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.timer): Info: \* Created gptimer with following parameters:

@0 s /0 (top.timer): Info: \* ------------------------------------------

@0 s /0 (top.timer): Info: \* ntimers: 7

@0 s /0 (top.timer): Info: \* pindex: 0

@0 s /0 (top.timer): Info: \* paddr/pmask: 0/fff

@0 s /0 (top.timer): Info: \* pirq: 0

@0 s /0 (top.timer): Info: \* sepirq: 0

@0 s /0 (top.timer): Info: \* sbits: 18

@0 s /0 (top.timer): Info: \* nbits: 22

@0 s /0 (top.timer): Info: \* wdog: 0

@0 s /0 (top.timer): Info: \* pow\_mon: 0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## IRQMP

The configuration report for the IRQMP is generated in the constructor of class irqmp. The report shows all the constructor parameters of the module. The example refers to an instance configured for two master CPUs. The extended interrupt (eirq) is mapped to IRQ line 4.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (top.irqmp): Info: \* Created configuration report with following parameters:

@0 s /0 (top.irqmp): Info: \* --------------------------------------------

@0 s /0 (top.irqmp): Info: \* paddr/pmask: 0/fff

@0 s /0 (top.irqmp): Info: \* ncpu: 2

@0 s /0 (top.irqmp): Info: \* eirq: 4

@0 s /0 (top.irqmp): Info: \* pindex: 0

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## SoCWire

The configuration report for the SoCWire IP contains all the constructor parameter of the module. It is generated in the constructor of class AHB2Socwire. The example shows a report for the AT configuration with APB index 1 (pindex) and AHB index 2 (hindex). In interrupt mode the device utilizes IRQ line 3 (apb\_irq\_id).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@0 s /0 (DUT): Info: \* Created AHB2SoCWire with following parameters:

@0 s /0 (DUT): Info: \* ----------------------------------------------

@0 s /0 (DUT): Info: \* paddr/pmask: fff/fff

@0 s /0 (DUT): Info: \* pindex: 1

@0 s /0 (DUT): Info: \* apb\_irq\_id: 3

@0 s /0 (DUT): Info: \* hindex: 2

@0 s /0 (DUT): Info: \* abstractionLayer (LT = 8 / AT = 4): 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Performance & Communication analysis

Each module of the library provides execution statistics in terms of performance and communication figures. The reports are generated in the end\_of\_simulation functions of the various modules.

## AHBCTRL

The execution statistic of the AHBCTRL contains the number of successful transactions and the number of total transactions. The difference between these figures is the number of transactions, which did not complete with TLM\_OK\_RESPONSE.

The report also shows the total number of simulation cycles and the number of idle cycles. In idle cycles no master requested the bus. From this information the module calculates the bus utilization. A utilization of 1 indicates that no idle cycles occurred during simulation.

The report below was generated for an AT simulation. Information about the maximum and average arbitration time is only given for AT mode.

@400 us (ahbctrl): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@400 us (ahbctrl): Report: \* AHBCtrl Statistic:

@400 us (ahbctrl): Report: \* ------------------

@400 us (ahbctrl): Report: \* Successful Transactions: 32000

@400 us (ahbctrl): Report: \* Total Transactions: 32000

@400 us (ahbctrl): Report: \*

@400 us (ahbctrl): Report: \* Simulation cycles: 40000

@400 us (ahbctrl): Report: \* Idle cycles: 7999

@400 us (ahbctrl): Report: \* Bus utilization: 0.800025

@400 us (ahbctrl): Report: \* Maximum arbiter waiting time:   
 150000001 ps (15000 cycles)

@400 us (ahbctrl): Report: \* Master with maximum waiting time: 0

@400 us (ahbctrl): Report: \* Average arbitration time / transaction:   
 84990 ps (8.499 cycles)

@400 us (ahbctrl): Report: \*

@400 us (ahbctrl): Report: \* AHB Master interface reports:

@400 us (ahbctrl): Report: \* Bytes read: 64000

@400 us (ahbctrl): Report: \* Bytes written: 64000

@400 us (ahbctrl): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## APBCTRL

The APBCTRL provides information about the total number of transactions and the total number successful transactions. The report also shows how many bytes have been read and written via the AHB slave interface.

@1809990 ns (top.apbctrl): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@1809990 ns (top.apbctrl): Report: \* APBCtrl Statistic:

@1809990 ns (top.apbctrl): Report: \* ------------------

@1809990 ns (top.apbctrl): Report: \* Successful Transactions: 10

@1809990 ns (top.apbctrl): Report: \* Total Transactions: 10

@1809990 ns (top.apbctrl): Report: \*

@1809990 ns (top.apbctrl): Report: \* AHB Slave interface reports:

@1809990 ns (top.apbctrl): Report: \* Bytes read: 24

@1809990 ns (top.apbctrl): Report: \* Bytes written: 16

@1809990 ns (top.apbctrl): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## MCTRL

The MCTRL also provides information about the total number of transactions and the number of successful transactions. In the example below these figures are not equal. The tests of the MCTRL contains vectors, which are intended to fail to improve the test coverage.

The report also shows information about the time spent in the various SDRAM operation modes, and the volume of data transmitted via the AHB slave interface.

@476470 ns (top.mctrl): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.mctrl): Report: \* Mctrl Statistic:

@476470 ns (top.mctrl): Report: \* ----------------

@476470 ns (top.mctrl): Report: \* Successful Transactions: 1672

@476470 ns (top.mctrl): Report: \* Total Transactions: 1756

@476470 ns (top.mctrl): Report: \*

@476470 ns (top.mctrl): Report: \* Time in nom. operation: 433250 ns (90.9291%)

@476470 ns (top.mctrl): Report: \* Time self-refresh mode: 0 s (0%)

@476470 ns (top.mctrl): Report: \* Time in power down mode: 17220 ns (3.61408%)

@476470 ns (top.mctrl): Report: \* Time in deep pwr down mode: 26 us (5.4568%)

@476470 ns (top.mctrl): Report: \*

@476470 ns (top.mctrl): Report: \* AHB Slave interface reports:

@476470 ns (top.mctrl): Report: \* Bytes read: 2609

@476470 ns (top.mctrl): Report: \* Bytes written: 2465

@476470 ns (top.mctrl): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Generic Memory

The Generic Memory (GM) can be configured to represent ROM, IO, SRAM and SDRAM. Depending on the intended memory layout one or multiple instances of the GM can be connected to the MCTRL. The execution report of the GM provides information about the data volume transferred to/from the particular memory.

@476470 ns (top.rom): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.rom): Report: \* ROM Memory Statistic:

@476470 ns (top.rom): Report: \* -----------------------------------------

@476470 ns (top.rom): Report: \* Bytes read: 1172

@476470 ns (top.rom): Report: \* Bytes written: 1100

@476470 ns (top.rom): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.io): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.io): Report: \* IO Memory Statistic:

@476470 ns (top.io): Report: \* -----------------------------------------

@476470 ns (top.io): Report: \* Bytes read: 1168

@476470 ns (top.io): Report: \* Bytes written: 1164

@476470 ns (top.io): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.sram): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.sram): Report: \* SRAM Memory Statistic:

@476470 ns (top.sram): Report: \* -----------------------------------------

@476470 ns (top.sram): Report: \* Bytes read: 2216

@476470 ns (top.sram): Report: \* Bytes written: 2142

@476470 ns (top.sram): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.sdram): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@476470 ns (top.sdram): Report: \* SDRAM Memory Statistic:

@476470 ns (top.sdram): Report: \* -----------------------------------------

@476470 ns (top.sdram): Report: \* Bytes read: 837

@476470 ns (top.sdram): Report: \* Bytes written: 769

@476470 ns (top.sdram): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## MMU\_CACHE

Depending on the intended configuration, the MMU\_CACHE top-level instantiates different sub-components such as caches and the mmu. All of them generate detailed execution statistics on their own. Therefore, the MMU\_CACHE execution statistic only provides information on the number of transactions and the data volume transferred over the AHB master interface.

@12640 ns /218 (mmu\_cache): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@12640 ns /218 (mmu\_cache): Report: \* MMU\_CACHE Statistics:

@12640 ns /218 (mmu\_cache): Report: \* ---------------------

@12640 ns /218 (mmu\_cache): Report: \* Successful Transactions: 72

@12640 ns /218 (mmu\_cache): Report: \* Total Transactions: 72

@12640 ns /218 (mmu\_cache): Report: \*

@12640 ns /218 (mmu\_cache): Report: \* AHB Master interface reports:

@12640 ns /218 (mmu\_cache): Report: \* Bytes read: 112

@12640 ns /218 (mmu\_cache): Report: \* Bytes written: 128

@12640 ns /218 (mmu\_cache): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Cache

The execution statistic of the caches contain information about the hits and misses in each particular cache-set. The report also shows the number of bypass operations and calculates the hit-rate for read and write operations. A hit-rate of 1 relates to a perfect cache.

The example below shows no write-hit rate, because the number of writes (with caches enabled) is zero.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@246700 ns (mmu\_cache.dcache): Report: \* Caching statistic:

@246700 ns (mmu\_cache.dcache): Report: \* -------------------

@246700 ns (mmu\_cache.dcache): Report: \* Read hits set0: 256

@246700 ns (mmu\_cache.dcache): Report: \* Read hits set1: 256

@246700 ns (mmu\_cache.dcache): Report: \* Read hits set2: 256

@246700 ns (mmu\_cache.dcache): Report: \* Read hits set3: 256

@246700 ns (mmu\_cache.dcache): Report: \* Total Read Hits: 1024

@246700 ns (mmu\_cache.dcache): Report: \* Read Misses: 1024

@246700 ns (mmu\_cache.dcache): Report: \* Read Hit Rate: 0.5

@246700 ns (mmu\_cache.dcache): Report: \* Write hits set0: 0

@246700 ns (mmu\_cache.dcache): Report: \* Write hits set1: 0

@246700 ns (mmu\_cache.dcache): Report: \* Write hits set2: 0

@246700 ns (mmu\_cache.dcache): Report: \* Write hits set3: 0

@246700 ns (mmu\_cache.dcache): Report: \* Total Write Hits: 0

@246700 ns (mmu\_cache.dcache): Report: \* Write Misses: 0

@246700 ns (mmu\_cache.dcache): Report: \* Bypass ops: 4096

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Localram

Two localrams, for instructions and data, can be connected to the caches. The localrams represent simple and fast storage devices. The localram execution statistic shows how many bytes have been transferred to/from a particular instance.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@34390002 ps (mmu\_cache.dlocalram): Report: \* Scratchpad statisitics:

@34390002 ps (mmu\_cache.dlocalram): Report: \* -----------------------

@34390002 ps (mmu\_cache.dlocalram): Report: \* Read accesses: 134 (Bytes: 520)

@34390002 ps (mmu\_cache.dlocalram): Report: \* Write accesses: 133 (Bytes: 516) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### MMU

The MMU can be configured for up the 32 instruction TLBs and 32 data TLBs (split mode), or 32 shared instruction and data TLBs. The MMU execution report shows the number of hits in each particular TLB and the accumulated number of TLB misses. From this information the module calculates the TLB hit rate, which is given in percent. A hit-rate of 100% relates to a perfect MMU.

@164770 ns /16445 (mmu\_cache.mmu): Info: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@164770 ns /16445 (mmu\_cache.mmu): Info: \* MMU statistic:

@164770 ns /16445 (mmu\_cache.mmu): Info: \* -------------------

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB0: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB1: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB2: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB3: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB4: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB5: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB6: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in ITLB7: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Misses in ITLB: 0

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB0: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB1: 1022

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB2: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB3: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB4: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB5: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB6: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Hits in DTLB7: 1023

@164770 ns /16445 (mmu\_cache.mmu): Info: \* Misses in DTLB: 8

@164770 ns /16445 (mmu\_cache.mmu): Info: \* DTLB hit rate: 99%

@164770 ns /16445 (mmu\_cache.mmu): Info: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## GPTimer

The GPTimer module consists of a top level class GPtimer, which contains multiple instances of independent counters. The number of counters can be configured using a constructor parameter (ncounters). Each counter generates a small report showing the number of underflows during simulation.

@1030 ns /135 (top.timer.GPCounter): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@1030 ns /135 (top.timer.GPCounter): Report: \* GPCounter Statistic:

@1030 ns /135 (top.timer.GPCounter): Report: \* ----------------

@1030 ns /135 (top.timer.GPCounter): Report: \* Counter Underflows: 4

@1030 ns /135 (top.timer.GPCounter): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@1030 ns /135 (top.timer.GPCounter\_1): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@1030 ns /135 (top.timer.GPCounter\_1): Report: \* GPCounter Statistic:

@1030 ns /135 (top.timer.GPCounter\_1): Report: \* ----------------

@1030 ns /135 (top.timer.GPCounter\_1): Report: \* Counter Underflows: 1

@1030 ns /135 (top.timer.GPCounter\_1): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## IRQMP

The IRQMP provides 32 input lines (16 regular + 16 extended) for the connection of interrupt sources. Depending on the configuration and various internal masks, incoming interrupts are forwarded to one or multiple CPUs (max. 16).

The IRQMP execution statistic shows how many interrupts have been generated by which source, and how many interrupts have been sent to which CPU.

@164966 ns /952 (top.irqmp): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@164966 ns /952 (top.irqmp): Report: \* IRQMP statistic:

@164966 ns /952 (top.irqmp): Report: \* ================

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 1: 3

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 2: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 3: 3

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 4: 0

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 5: 3

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 6: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 7: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 8: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 9: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 10: 3

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 11: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 12: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 13: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 14: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 15: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 16: 2

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 17: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 18: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 19: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 20: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 21: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 22: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 23: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 24: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 25: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 26: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 27: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 28: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 29: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 30: 1

@164966 ns /952 (top.irqmp): Report: \* + IRQ Line 31: 1

@164966 ns /952 (top.irqmp): Report: \* -------------------------------

@164966 ns /952 (top.irqmp): Report: \* = Sum : 49

@164966 ns /952 (top.irqmp): Report: \*

@164966 ns /952 (top.irqmp): Report: \* + CPU Line 0: 49

@164966 ns /952 (top.irqmp): Report: \* + CPU Line 1: 56

@164966 ns /952 (top.irqmp): Report: \* -------------------------------

@164966 ns /952 (top.irqmp): Report: \* = Sum : 105

@164966 ns /952 (top.irqmp): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## SoCWire

The SoCWire execution statistic shows how many SoCWire packets have been sent and received. It also indicates how many errors occured during simulation. Moreover, the report shows the volume of data transferred over the SoCWire link and the number of consumed RX and TX descriptors.

@3050 ns (DUT): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@3050 ns (DUT): Report: \* AHB2SoCWire Statistic:

@3050 ns (DUT): Report: \* -----------------------------------------

@3050 ns (DUT): Report: \* SoCWire packets received (with/without error,   
 with error RE\_ReceiverError,   
 without error): 1, 0, 1

@3050 ns (DUT): Report: \* Bytes received successfully on SoCWire link: 12

@3050 ns (DUT): Report: \* SoCWire packets sent (with/without error): 0

@3050 ns (DUT): Report: \* Bytes sent on SoCWire link: 0

@3050 ns (DUT): Report: \* Packets received without   
 available RX descriptor (SS\_SocWireStall): 1

@3050 ns (DUT): Report: \* RX descriptors consumed: 0

@3050 ns (DUT): Report: \* TX descriptors consumed: 0

@3050 ns (DUT): Report: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*