
The PoC-Library Documentation

Release 1.2.0

Patrick Lehmann, Thomas B. Preusser, Martin Zabel

Sep 11, 2018

I	Introduction	1
1	What is PoC?	5
1.1	What is the History of PoC?	6
1.2	Which Tool Chains are supported?	6
1.3	Why should I use PoC?	7
1.4	Who uses PoC?	7
2	Quick Start Guide	9
2.1	Requirements and Dependencies	9
2.2	Download	10
2.3	Configuring PoC on a Local System	10
2.4	Integration	10
2.5	Run a Simulation	12
2.6	Run a Synthesis	13
2.7	Updating	13
3	Get Involved	15
3.1	Report a Bug	15
3.2	Feature Request	15
3.3	Talk to us on Gitter	15
3.4	Contributors License Agreement	16
3.5	Contribute to PoC	16
3.6	Give us Feedback	18
3.7	List of Contributors	18
4	Apache License 2.0	19
4.1	1. Definitions.	19
4.2	2. Grant of Copyright License.	20
4.3	3. Grant of Patent License.	20
4.4	4. Redistribution.	20
4.5	5. Submission of Contributions.	21
4.6	6. Trademarks.	21
4.7	7. Disclaimer of Warranty.	21
4.8	8. Limitation of Liability.	21
4.9	9. Accepting Warranty or Additional Liability.	21
II	Main Documentation	23
5	Using PoC	25
5.1	Requirements	26

5.2	Downloading PoC	28
5.3	Integrating PoC into Projects	31
5.4	Configuring PoC's Infrastructure	32
5.5	Creating my_config/my_project.vhdl	36
5.6	Adding IP Cores to a Project	37
5.7	Simulation	38
5.8	Synthesis	45
5.9	Project Management	49
5.10	Pre-Compiling Vendor Libraries	49
5.11	Miscellaneous	56
6	IP Core Interfaces	57
6.1	Command-Status-Error (PoC.CSE) Interface	57
6.2	PoC.FIFO Interface	57
6.3	PoC.Mem Interface	57
6.4	PoC.Stream Interface	59
7	IP Core Documentations	61
7.1	Common Packages	61
7.2	Simulation Packages	62
7.3	PoC.alt	64
7.4	PoC.arith	64
7.5	PoC.bus	74
7.6	PoC.cache	80
7.7	PoC.comm	91
7.8	PoC.dstruct	92
7.9	PoC.fifo	94
7.10	PoC.io	101
7.11	PoC.mem	118
7.12	PoC.misc	137
7.13	PoC.net	152
7.14	PoC.sort	188
7.15	PoC.xil	196
8	Third Party Libraries	203
8.1	Cocotb	203
8.2	OSVVM	203
8.3	UVVM	204
8.4	VUnit	204
8.5	Updating Linked Git Submodules	204
9	Constraint Files	207
9.1	IP Core Constraint Files	207
9.2	Board Constraint Files	208
10	Tool Chain Specifics	213
10.1	Aldec Active-HDL	213
10.2	Mentor QuestaSim	213
10.3	Xilinx ISE	213
10.4	Xilinx Vivado	213
11	Examples	215
III	References	217
12	Command Reference	219
12.1	PoC Wrapper Scripts	219
12.2	Main Program (PoC.py)	220

12.3	Pre-compile Scripts	220
13	IP Core Database	231
13.1	Overview	231
13.2	Database Structure	232
13.3	Supported Options	234
13.4	Files in detail	235
13.5	User Defined Variables	235
14	More References	237
14.1	List of Supported FPGA Devices	237
14.2	List of Supported Boards	238
14.3	Wrapper Script Hook Files	238
14.4	File Formats	239
14.5	Naming Conventions	243
14.6	Known Issues	245
14.7	Local License Copies	246
IV	Appendix	261
15	Change Log	263
16	Index	265

Part I

Introduction

This library is published and maintained by **Chair for VLSI Design, Diagnostics and Architecture** - Faculty of Computer Science, Technische Universität Dresden, Germany <https://tu-dresden.de/ing/informatik/ti/vlsi>



PoC - “Pile of Cores” provides implementations for often required hardware functions such as Arithmetic Units, Caches, Clock-Domain-Crossing Circuits, FIFOs, RAM wrappers, and I/O Controllers. The hardware modules are typically provided as VHDL or Verilog source code, so it can be easily re-used in a variety of hardware designs.

All hardware modules use a common set of VHDL packages to share new VHDL types, sub-programs and constants. Additionally, a set of simulation helper packages eases the writing of testbenches. Because PoC hosts a huge amount of IP cores, all cores are grouped into sub-namespaces to build a better hierarchy.

Various simulation and synthesis tool chains are supported to interoperate with PoC. To generalize all supported free and commercial vendor tool chains, PoC is shipped with a Python based infrastructure to offer a command line based frontend.

News

See [Change Log](#) for latest updates.

Cite the PoC-Library

The PoC-Library hosted at [GitHub.com](https://github.com/VLSI-EDA/PoC). Please use the following [bibtex](#) entry to cite us:

```
# BibLaTeX example entry
@online{poc,
  title={PoC - Pile of Cores},
  author={Chair of VLSI Design, Diagnostics and Architecture},
  organization={Technische Universität Dresden},
  year={2016},
  url={https://github.com/VLSI-EDA/PoC},
  urldate={2016-10-28},
}
```


CHAPTER 1

What is PoC?

PoC - “Pile of Cores” provides implementations for often required hardware functions such as Arithmetic Units, Caches, Clock-Domain-Crossing Circuits, FIFOs, RAM wrappers, and I/O Controllers. The hardware modules are typically provided as VHDL or Verilog source code, so it can be easily re-used in a variety of hardware designs.

All hardware modules use a common set of VHDL packages to share new VHDL types, sub-programs and constants. Additionally, a set of simulation helper packages eases the writing of testbenches. Because PoC hosts a huge amount of IP cores, all cores are grouped into sub-namespaces to build a better hierarchy.

Various simulation and synthesis tool chains are supported to interoperate with PoC. To generalize all supported free and commercial vendor tool chains, PoC is shipped with a Python based Infrastructure to offer a command line based frontend.

The PoC-Library pursues the following five goals:

- independence in the platform, target, vendor and tool chain
- generic, efficient, resource sparing and fast implementations of IP cores
- optimized for several device architectures, if suitable
- supportive scripts to ease the IP core handling with all supported vendor tools on all listed operating systems
- ship all IP cores with testbenches for local and online verification

In detail the PoC-Library is:

- synthesizable for ASIC and FPGA devices, e.g. from Altera, Lattice, Xilinx, ... ,
- supports a wide range of simulation and synthesis tool chains, and is
- executable on several host platforms: Darwin, Linux or Windows.

This is achieved by using generic HDL descriptions, which work with most synthesis and simulation tools mentioned above. If this is not the case, then PoC uses vendor or tool dependent work-arounds. These work-arounds can be different implementations switched by VHDL *generate* statements as well as different source files containing modified implementations.

One special feature of PoC is it, that the user has not to take care of such implementation switchings. PoC's IP cores decide on their own what's the *best* implementation for the chosen target platform. For this feature, PoC

implements a configuration package, which accepts a well-known development board name or a target device string. For example a FPGA device string is decoded into: vendor, device, generation, family, subtype, speed grade, pin count, etc. Out of these information, the PoC component can for example implement a vendor specific carry-chain description to speed up an algorithm or group computation units to effectively use 6-input LUTs.

1.1 What is the History of PoC?

In the past years, a lot of “IP cores” were developed at the chair of VLSI design¹. This loose set of HDL designs was gathered in an old-fashioned CVS repository and grew over the years to a collection of basic HDL implementations like ALUs, FIFOs, UARTs or RAM controllers. For their final projects (bachelor, master, diploma thesis) students got access to PoC, so they could focus more on their main tasks than wasting time in developing and testing basic IP implementations from scratch. But the library was initially for internal and educational use only.

As a university chair for VLSI design, we have a wide range of different FPGA prototyping boards from various vendors and device families as well as generations. So most of the IP cores were developed for both major FPGA vendor platforms and their specific vendor tool chains. The main focus was to describe hardware in a more flexible and generic way, so that an IP core could be reused on multiple target platforms.

As the number of cores increased, the set of common functions and types increased too. In the end PoC is not only a collection of IP cores, it also shipped with a set of packages containing utility functions, new types and type conversions, which are used by most of the cores. This makes PoC a *library*, not only a *collection* of IPs.

As we started to search for ways to publish IP cores and maybe the whole PoC-Library, we found several platforms on the Internet, but none was very convincing. Some collective websites contained inactive projects, others were controlled by companies without the possibility to contribute and the majority was a long list of private projects with at most a handful of IP cores. Another disagreement were the used license types for these projects. We decided to use the Apache License, because it has no copyleft rule, a patent clause and allows commercial usage.

We transformed the old CVS repository into three Git repositories: An internal repository for the full set of IP cores (incl. classified code), a public one and a repository for examples, called PoC-Examples, both hosted on GitHub. PoC itself can be integrated into other HDL projects as a library directory or a Git submodule. The preferred usage is the submodule integration, which has the advantage of linked repository versions from hosting Git and the submodule Git. This is already exemplified by our PoC-Examples repository.

1.2 Which Tool Chains are supported?

The PoC-Library and its Python-based infrastructure currently supports the following free and commercial vendor tool chains:

- Synthesis Tool Chains:
 - **Altera Quartus** Tested with Quartus-II ≥ 13.0 . Tested with Quartus Prime ≥ 15.1 .
 - **Intel Quartus** Tested with Quartus Prime ≥ 16.1 .
 - **Lattice Diamond** Tested with Diamond ≥ 3.6 .
 - **Xilinx ISE** Only ISE 14.7 inclusive Core Generator 14.7 is supported.
 - **Xilinx PlanAhead** Only PlanAhead 14.7 is supported.
 - **Xilinx Vivado** Tested with Vivado ≥ 2015.4 . Due to a limited VHDL language support compared to ISE 14.7, some PoC IP cores need special work arounds. See the synthesis documentation section for Vivado for more details.
- Simulation Tool Chains:

¹ The PoC-Library is published and maintained by the **Chair for VLSI Design, Diagnostics and Architecture** - Faculty of Computer Science, Technische Universität Dresden, Germany <http://tu-dresden.de/inf/vlsi-edu>

- **Aldec Active-HDL** Tested with Active-HDL (or Student-Edition) ≥ 10.3 Tested with Active-HDL Lattice Edition ≥ 10.2
- **Cocotb with Mentor QuestaSim backend** Tested with Mentor QuestaSim 10.4d
- **Mentor Graphics ModelSim** Tested with ModelSim PE (or Student Edition) $\geq 10.5c$ Tested with ModelSim SE $\geq 10.5c$ Tested with ModelSim Altera Edition 10.3d (or Starter Edition)
- **Mentor Graphics QuestaSim/ModelSim** Tested with Mentor QuestaSim $\geq 10.4d$
- **Xilinx ISE Simulator** Tested with ISE Simulator (iSim) 14.7. The Python infrastructure supports isim, but PoC's simulation helper packages and testbenches rely on VHDL-2008 features, which are not supported by isim.
- **Xilinx Vivado Simulator** Tested with Vivado Simulator (xsim) ≥ 2016.3 . The Python infrastructure supports xsim, but PoC's simulation helper packages and testbenches rely on VHDL-2008 features, which are not fully supported by xsim, yet.
- **GHDL + GTKWave** Tested with [GHDL](#) $\geq 0.34dev$ and [GTKWave](#) $\geq 3.3.70$ Due to ongoing development and bugfixes, we encourage to use the newest GHDL version.

1.3 Why should I use PoC?

Here is a brief list of advantages:

- We explicitly use the wording *PoC-Library* rather than *collection*, because PoC's packages and IP cores build an ecosystem. Complex IP cores are build on-top of basic IP cores - they are no lose set of cores. The cores offer a clean interface and can be configured by many generic parameters.
- PoC is target independent: It's possible to switch the target device or even the device vendor without switching the IP core.

Todo: Use a well tested set of packages to ease the use of VHDL

Use a well tested set of simulation helpers

Run testbenches in various simulators.

Run synthesis tests in various synthesis tools.

Compare hardware usage for different target platforms.

Supports simulation with vendor primitive libraries, ships with script to pre-compile vendor libraries.

Vendor tools have bugs, check you IP cores when a new tool release is available, before changing code base

1.4 Who uses PoC?

PoC has a related Git repository called [PoC-Examples](#) on GitHub. This repository hosts a list of example and reference implementations of the PoC-Library. Additional to reading an IP cores documention and viewing its characteristic stimulus waveform in a simulation, it can helper to investigate an IP core usage example from that repository.

- [The Q27 Project](#) 27-Queens Puzzle: Massively Parellel Enumeration and Solution Counting
- [Reconfigurable Cloud Computing Framework \(RC2F\)](#) An FPGA computing framework for virtualization and cloud integration.
- [PicoBlaze-Library](#) The PicoBlaze-Library offers several PicoBlaze devices and code routines to extend a common PicoBlaze environment to a little System on a Chip (SoC or SoFPGA).

- [PicoBlaze-Examples](#) A SoFPGA reference implementation, based on the PoC-Library and the PicoBlaze-Library.

Quick Start Guide

This **Quick Start Guide** gives a fast and simple introduction into PoC. All topics can be found in the [Using PoC](#) section with much more details and examples.

Contents of this Page

- [Requirements and Dependencies](#)
- [Download](#)
- [Configuring PoC on a Local System](#)
- [Integration](#)
- [Run a Simulation](#)
- [Run a Synthesis](#)
- [Updating](#)

2.1 Requirements and Dependencies

The PoC-Library comes with some scripts to ease most of the common tasks, like running testbenches or generating IP cores. PoC uses Python 3 as a platform independent scripting environment. All Python scripts are wrapped in Bash or PowerShell scripts, to hide some platform specifics of Darwin, Linux or Windows. See [Requirements](#) for further details.

PoC requires:

- A [supported synthesis tool chain](#), if you want to synthesize IP cores.
- A [supported simulator tool chain](#), if you want to simulate IP cores.
- The **Python 3** programming language and runtime, if you want to use PoC's infrastructure.
- A shell to execute shell scripts:
 - **Bash** on Linux and OS X

– PowerShell on Windows

PoC optionally requires:

- **Git** command line tools or
- **Git User Interface**, if you want to check out the latest ‘master’ or ‘release’ branch.

PoC depends on third part libraries:

- **Cocotb**  A coroutine based cosimulation library for writing VHDL and Verilog testbenches in Python.
- **OSVVM**  Open Source VHDL Verification Methodology.
- **UVVM**  Universal VHDL Verification Methodology.
- **VUnit**  An unit testing framework for VHDL.

All dependencies are available as GitHub repositories and are linked to PoC as Git submodules into the **PoC-Rootlib** directory. See *Third Party Libraries* for more details on these libraries.

2.2 Download

The PoC-Library can be downloaded as a [zip-file](#) (latest ‘master’ branch), cloned with `git clone` or embedded with `git submodule add` from GitHub. GitHub offers HTTPS and SSH as transfer protocols. See the [Download](#) page for further details. The installation directory is referred to as PoCRoot.

Protocol	Git Clone Command
HTTPS	<code>git clone --recursive https://github.com/VLSI-EDA/PoC.git PoC</code>
SSH	<code>git clone --recursive ssh://git@github.com:VLSI-EDA/PoC.git PoC</code>

2.3 Configuring PoC on a Local System

To explore PoC’s full potential, it’s required to configure some paths and synthesis or simulation tool chains. The following commands start a guided configuration process. Please follow the instructions on screen. It’s possible to relaunch the process at any time, for example to register new tools or to update tool versions. See [Configuration](#) for more details. Run the following command line instructions to configure PoC on your local system:

```
cd PoCRoot
.\poc.ps1 configure
```

Use the keyboard buttons: to accept, to decline, to skip/pass a step and to accept a default value displayed in brackets.

2.4 Integration

The PoC-Library is meant to be integrated into other HDL projects. Therefore it’s recommended to create a library folder and add the PoC-Library as a Git submodule. After the repository linking is done, some short configuration steps are required to setup paths, tool chains and the target platform. The following command line instructions show a short example on how to integrate PoC.

1. Adding the Library as a Git submodule

The following command line instructions will create the folder `lib\PoC\` and clone the PoC-Library as a Git [submodule](#) into that folder. `ProjectRoot` is the directory of the hosting Git. A detailed list of steps can be found at [Integration](#).

```
cd ProjectRoot
mkdir lib | cd
git submodule add https://github.com/VLSI-EDA/PoC.git PoC
cd PoC
git remote rename origin github
cd ../../
git add .gitmodules lib\PoC
git commit -m "Added new git submodule PoC in 'lib\PoC' (PoC-Library)."
```

2. Configuring PoC

The PoC-Library should be configured to explore its full potential. See [Configuration](#) for more details. The following command lines will start the configuration process:

```
cd ProjectRoot
.\lib\PoC\poc.psl configure
```

3. Creating PoC's `my_config.vhdl` and `my_project.vhdl` Files

The PoC-Library needs two VHDL files for its configuration. These files are used to determine the most suitable implementation depending on the provided target information. Copy the following two template files into your project's source folder. Rename these files to `*.vhdl` and configure the VHDL constants in the files:

```
cd ProjectRoot
cp lib\PoC\src\common\my_config.vhdl.template src\common\my_config.vhdl
cp lib\PoC\src\common\my_project.vhdl.template src\common\my_project.vhdl
```

`my_config.vhdl` defines two global constants, which need to be adjusted:

```
constant MY_BOARD          : string := "CHANGE THIS"; -- e.g. Custom, ML505, ↵
↵ KC705, Atlys
constant MY_DEVICE         : string := "CHANGE THIS"; -- e.g. None, XC5VLX50T-
↵ 1FF1136, EP2SGX90FF1508C3
```

`my_project.vhdl` also defines two global constants, which need to be adjusted:

```
constant MY_PROJECT_DIR    : string := "CHANGE THIS"; -- e.g. d:/vhdl/myproject/,
↵ /home/me/projects/myproject/"
constant MY_OPERATING_SYSTEM : string := "CHANGE THIS"; -- e.g. WINDOWS, LINUX
```

Further informations are provided at [Creating my_config/my_project.vhdl](#).

4. Adding PoC's Common Packages to a Synthesis or Simulation Project

PoC is shipped with a set of common packages, which are used by most of its modules. These packages are stored in the `PoCRoot\src\common` directory. PoC also provides a VHDL context in `common.vhdl`, which can be used to reference all packages at once.

5. Adding PoC's Simulation Packages to a Simulation Project

Simulation projects additionally require PoC's simulation helper packages, which are located in the `PoCRoot\src\sim` directory. Because some VHDL version are incompatible among each other, PoC uses version suffixes like `*.v93.vhdl` or `*.v08.vhdl` in the file name to denote the supported VHDL version of a file.

6. Compiling Shipped IP Cores

Some IP Cores are shipped are pre-configured vendor IP Cores. If such IP cores shall be used in a HDL project, it's recommended to use PoC to create, compile and if needed patch these IP cores. See [Synthesis](#) for more details.

2.5 Run a Simulation

The following quick example uses the GHDL Simulator to analyze, elaborate and simulate a testbench for the module `arith_prng` (Pseudo Random Number Generator - PRNG). The VHDL file `arith_prng.vhdl` is located at `PoCRoot\src\arith` and virtually a member in the `PoC.arith` namespace. So the module can be identified by a unique name: `PoC.arith.prng`, which is passed to the frontend script.

Example:

```
cd PoCRoot
.\poc.ps1 ghdl PoC.arith.prng
```

The CLI command `ghdl` chooses *GHDL Simulator* as the simulator and passes the fully qualified PoC entity name `PoC.arith.prng` as a parameter to the tool. All required source file are gathered and compiled to an executable. Afterwards this executable is launched in CLI mode and its outputs are displayed in console:

```

PS G:\git\PoC> .\poc.ps1 ghdl PoC.arith.prng
=====
The PoC-Library - Service Tool
=====
Initializing PoC-Library Service Tool for simulations
Preparing simulation environment...
Testbench: PoC.arith.prng
Running analysis for every vhd1 file...
Running elaboration...
Running simulation...
ghdl run messages for 'test.arith_prng_tb'
=====
POC TESTBENCH REPORT
=====
Tests          2
-1: Default test
 0: Test setup for BITS=8; SEED=0x12

Overall
Assertions    256
  failed      0
Processes     3
  active      0
Runtime       2.6 us
=====
SIMULATION RESULT = PASSED
=====
Overall Simulation Report
=====
Name | Time | Status
-----|-----|-----
arith_prng | 0:03 | PASSED
=====
Time: 0:03 Count: 1 Passed: 1 No Asserts: 0 Failed: 0 Errors: 0
=====
PS G:\git\PoC>

```

Each testbench uses PoC's simulation helper packages to count asserts and to track active stimuli and checker processes. After a completed simulation run, an report is written to STDOUT or the simulator's console. Note the

line `SIMULATION RESULT = PASSED`. For each simulated PoC entity, a line in the overall report is created. It lists the runtime per testbench and the simulation status (`... ERROR, FAILED, NO ASSERTS` or `PASSED`). See [Simulation](#) for more details.

2.6 Run a Synthesis

The following quick example uses the Xilinx Systemic Tool (XST) to synthesize a netlist for IP core `arith_prng` (Pseudo Random Number Generator - PRNG). The VHDL file `arith_prng.vhdl` is located at `PoCRoot\src\arith` and virtually a member in the `PoC.arith` namespace. So the module can be identified by an unique name: `PoC.arith.prng`, which is passed to the frontend script.

Example:

```
cd PoCRoot
.\poc.ps1 xst PoC.arith.prng --board=KC705
```

The CLI command `xst` chooses *Xilinx Synthesis Tool* as the synthesizer and passes the fully qualified PoC entity name `PoC.arith.prng` as a parameter to the tool. Additionally, the development board name is required to load the correct `my_config.vhdl` file. All required source file are gathered and synthesized to a netlist.

```
D:\git\poc [paebells/master] = *5.0 -0 ! *3 ~9 -0 !> .\poc.ps1 xst PoC.arith.prng --board=KC705
Loading Xilinx ISE environment 'C:\Xilinx\14.7\ISE_DS\settings64.bat'
*****
The PoC-Library - Service Tool
*****
Initializing PoC-Library Service Tool for synthesis
IP core: PoC.arith.prng
Preparing synthesis environment...
Executing post-processing tasks...
Running Xilinx Synthesis Tool...
xst messages for 'arith_prng.xst'

* HDL Parsing *
*****
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/utills.vhdl" Line 1006: Function scale does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 716: Function vendor does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 759: Function device does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 814: Function device family does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 883: Function device number does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 892: Function device subtype does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 1008: Function lut_fanin does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 1035: Function transceiver_type does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/config.vhdl" Line 1121: Function getfsmencoding_gray does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/strings.vhdl" Line 172: Function to_ipstyle does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/strings.vhdl" Line 548: Function to_digit does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/strings.vhdl" Line 632: Function to_natural does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 294: Function to_haud does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 739: Function to_real does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 751: Function to_real does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 762: Function to_real does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 772: Function to_real does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 784: Function to_int does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 795: Function to_int does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 806: Function to_int does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/physical.vhdl" Line 817: Function to_int does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/components.vhdl" Line 117: Function ffdre does not always return a value.
WARNING:HDLCompiler:443 - "D:/git/poc/src/common/components.vhdl" Line 151: Function ffdre does not always return a value.
* HDL Elaboration *
*****
* HDL Synthesis *
*****
* Advanced HDL Synthesis *
*****
* Low Level Synthesis *
*****
* Partition Report *
*****
* Design Summary *
*****
Executing post-processing tasks...
Unloading Xilinx ISE environment...
D:\git\poc [paebells/master] = *5.0 -0 ! *3 ~9 -0 !>
```

2.7 Updating

The PoC-Library can be updated by using `git fetch` and `git merge`.

```
cd PoCRoot
# update the local repository
git fetch --prune
# review the commit tree and messages, using the 'treea' alias
git treea
# if all changes are OK, do a fast-forward merge
git merge
```

See also:

Running one or more testbenches The installation can be checked by running one or more of PoC's testbenches.

Running one or more netlist generation flows The installation can also be checked by running one or more of PoC's synthesis flows.

A first step might be to use and explore PoC and its infrastructure in an own project. Moreover, we encourage to read our online help which covers all aspects from quickstart example up to detailed IP core documentation. While using PoC, you might discover issues or missing feature. Please report them as *listed below*. If you have an interesting project, please send us feedback or get listed on our *Who uses PoC?* page.

If you are more familiar with PoC and its components, you might start asking yourself how components internally work. Please read our more advanced topics in the online help, read our inline source code comments or start a discussion on *Gitter* to ask us directly.

Now you should be very familiar with our work and you might be interested in developing own components and contribute them to the main repository. See the *next section* for detailed instructions on the Git fork, commit, push and pull-request flow.

PoC ships some *third-party libraries*. If you are interested in getting your library or components shipped as part of PoC or as a third-party components, please contact us.

3.1 Report a Bug

Please report issues of any kind in our Git provider's issue tracker. This allows us to categorize issues into groups and assign developers to them. You can track the issue's state and see how it's getting solved. All enhancements and feature requests are tracked on GitHub at [GitHub Issues](#).

3.2 Feature Request

Please report missing features of any kind. We are always looking forward to provide a full feature set. Please use our Git provider's issue tracker to report enhancements and feature requests, so you can track the request's status and implementation. All enhancements and feature requests are tracked on GitHub at [GitHub Issues](#).

3.3 Talk to us on Gitter

You can chat with us on [Gitter](#) in our Gitter Room [VLSI-EDA/PoC](#). You can use Gitter for free with your existing GitHub or Twitter account.

3.4 Contributors License Agreement

We require all contributors to sign a Contributor License Agreement (CLA). If you don't know whatfore a CLA is needed and how it prevents legal issues on both sides, read [this short blog post](#). PoC uses the *Apache Contributor License Agreement* to match the *Apache License 2.0*.

So to get started, [sign the Contributor License Agreement \(CLA\)](#) at [CLAHub.com](#). You can authenticate yourself with an existing GitHub account.

3.5 Contribute to PoC

Contributing source code via Git is very easy. We don't provide direct write access to our repositories. Git offers the fork and pull-request philosophy, which means: You clone a repository, provide your changes in your own repository and notify us about outstanding changes via a pull-requests. We will then review your proposed changes and integrate them into our repository.

Steps 1 to 5 are done only once for setting up a forked repository.

3.5.1 1. Fork the PoC Repository

Git repositories can be cloned on a Git provider's server. This procedure is called *forking*. This allows Git providers to track the repository's network, check if repositories are related to each other and notify if pull-requests are available.

Fork our repository `VLSI-EDA/PoC` on GitHub into your or your's Git organisation's account. In the following the forked repository is referenced as `<username>/PoC`.

3.5.2 2. Clone the new Fork

Clone this new fork to your machine. See [Downloading via Git clone](#) for more details on how to clone PoC. If you have already cloned PoC, then you can setup the new fork as an additional *remote*. You should set `VLSI-EDA/PoC` as fetch target and the new fork `<username>/PoC` as push target.

Shell Commands for Cloning:

```
cd GitRoot
git clone --recursive "ssh://git@github.com:<username>/PoC.git" PoC
cd PoC
git remote rename origin github
git remote add upstream "ssh://git@github.com:VLSI-EDA/PoC.git"
git fetch --prune --tags
```

Shell Commands for Editing an existing Clone:

```
cd PoCRoot
git remote rename github upstream
git remote add github "ssh://git@github.com:<username>/PoC.git"
git fetch --prune --tags
```

These commands work for Git submodules too.

3.5.3 3. Checkout a Branch

Checkout the `master` or `release` branch and maybe stash outstanding changes.

```
cd PoCRoot
git checkout release
```

3.5.4 4. Setup PoC for Developers

Run PoC's *configuration routines* and setup the developer tools.

```
cd PoCRoot
.\PoC.ps1 configure git
```

3.5.5 5. Create your own master Branch

Each developer has his own master branch. So create one and check it out.

```
cd PoCRoot
git branch <username>/master
git checkout <username>/master
git push github <username>/master
```

If PoC's branches are moving forward, you can update your own master branch by merging changes into your branch.

3.5.6 6. Create your Feature Branch

Each new feature or bugfix is developed on a feature branch. Examples for branch names:

Branch name	Description
bugfix-utils	Fixes a bug in <code>utils.vhdl</code> .
docs-spelling	Fixes the documentation.
spi-controller	A new SPI controller implementation.

```
cd PoCRoot
git branch <username>/<feature>
git checkout <username>/<feature>
git push github <username>/<feature>
```

3.5.7 7. Commit and Push Changes

Commit your proposed changes onto your feature branch and push all changes to GitHub.

```
cd PoCRoot
# git add ....
git commit -m "Fixed a bug in function bounds() in utils.vhdl."
git push github <username>/<feature>
```

3.5.8 8. Create a Pull-Request

Go to your forked repository and click on “Compare and Pull-Request” or go to our PoC repository and create a new [pull request](#).

If this is your first Pull-Request, you need to sign our Contributors License Agreement (CLA).

3.5.9 9. Keep your master up-to-date

Todo: undocumented

3.6 Give us Feedback

Please send us feedback about the PoC documentation, our IP cores or your user story on how you use PoC.

3.7 List of Contributors

Contributor ¹	Contact E-Mail
Genßler, Paul	paul.genssler@tu-dresden.de
Köhler, Steffen	steffen.koehler@tu-dresden.de
Lehmann, Patrick ²	patrick.lehmann@tu-dresden.de; paebbels@gmail.com
Preußner, Thomas B. ²	thomas.preusser@tu-dresden.de; thomas.preusser@utexas.edu
Reichel, Peter	peter.reichel@eas.iis.fraunhofer.de; peter@peterreichel.info
Schirok, Jan	janschirok@gmx.net
Voß, Jens	jens.voss@mailbox.tu-dresden.de
Zabel, Martin ²	martin.zabel@tu-dresden.de

Note: This is a local copy of the [Apache License Version 2.0](#).

¹ In alphabetical order.

² Maintainer.

Version 2.0, January 2004

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

4.1 1. Definitions.

“License” shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

“Licensor” shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

“Legal Entity” shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, **“control”** means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

“You” (or **“Your”**) shall mean an individual or Legal Entity exercising permissions granted by this License.

“Source” form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

“Object” form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

“Work” shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

“Derivative Works” shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

“Contribution” shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, **“submitted”** means any form of electronic, verbal,

or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as **“Not a Contribution.”**

“Contributor” shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

4.2 2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

4.3 3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4.4 4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a **“NOTICE”** text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

4.5 5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

4.6 6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

4.7 7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

4.8 8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

4.9 9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

Appendix: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets “[]” replaced with your own identifying information. (Don’t include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same “printed page” as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

Part II

Main Documentation

PoC can be used in several ways, if all *Requirements* are fulfilled. Chose one of the following integration kinds:

- **Stand-Alone IP Core Library:** Download PoC as archive file (*.zip) from GitHub as latest branch copy or as tagged release file. IP cores can be copied into one or more destination projects or the projects link to the selected IP core source files.

Advantages:

- Simple and fast setup, configuring PoC is optional.
- Needs less disk space than a Git repository.
- After a configuration, PoC's additional features: simulation, synthesis, etc. can be used.

Disadvantages:

- Manual updating via download and file overwrites.
- Updated IP cores need to be copied again into the destination project.
- Using different PoC versions in different projects is not possible.
- No possibility to contribute bugfixes and extensions via Git pull requests.

Next steps: 1. See *Downloads* for how to download a stand-alone version (*.zip-file) of the PoC-Library. 2. See *Configuration* for how to configure PoC on a local system.

- **Stand-Alone IP Core Library cloned from Git:** Download PoC via `git clone` from GitHub as latest branch copy. IP cores can be copied into one or more destination projects or the projects link to the selected IP core source files.

Advantages:

- Simple and fast setup, configuring PoC is optional.
- Access to the newest commits on a branch: New IP cores, new features, bugfixes.
- Fast and simple updates via `git pull`.
- After a configuration, PoC's additional features: simulation, synthesis, etc. can be used.
- Contribute bugfixes and extensions via Git pull requests.

Disadvantages:

- Updated IP cores need to be copied again into the destination project.

- Using different PoC versions in different projects is not possible

Next steps: 1. See [Downloads](#) for how to clone a stand-alone version of the PoC-Library. 2. See [Configuration](#) for how to configure PoC on a local system.

- **Embedded IP Core Library as Git Submodule:** Integrate PoC as a Git submodule into the destination projects Git repository.

Advantages:

- Simple and fast setup, configuring PoC is optional, but recommended.
- Access to the newest commits on a branch: New IP cores, new features, bugfixes.
- Fast and simple updates via `git pull`.
- After a configuration, PoC's additional features: simulation, synthesis, etc. can be used.
- Moreover, some PoC infrastructure features can be used in the hosting repository and project as well.
- Contribute bugfixes and extensions via Git pull requests.
- Version linking between hosting Git and PoC.

Next steps: 1. See [Integration](#) for how to integrate PoC as a Git submodule into an existing Git. 2. See [Configuration](#) for how to configure PoC on a local system.

5.1 Requirements

Contents of this Page

- *Common requirements:*
- *Linux specific requirements:*
 - *Optional Tools on Linux:*
- *Mac OS specific requirements:*
 - *Optional Tools on Mac OS:*
- *Windows specific requirements:*
 - *Optional Tools on Windows:*

The PoC-Library comes with some scripts to ease most of the common tasks, like running testbenches or generating IP cores. We choose to use Python 3 as a platform independent scripting environment. All Python scripts are wrapped in Bash or PowerShell scripts, to hide some platform specifics of Darwin, Linux or Windows.

5.1.1 Common requirements:

Programming Languages and Runtime Environments:

- Python 3 (≥ 3.5):
 - colorama
 - py-flags

All Python requirements are listed in [requirements.txt](#) and can be installed via: `sudo python3.5 -m pip install -r requirements.txt`

Synthesis tool chains:

- Altera Quartus II ≥ 13.0 or

- Altera Quartus Prime ≥ 15.1 or
- Intel Quartus Prime ≥ 16.1 or
- Lattice Diamond ≥ 3.6 or
- Xilinx ISE 14.7¹ or
- Xilinx Vivado ≥ 2016.3 ²

Simulation tool chains

- Aldec Active-HDL (or Student Edition) or
- Aldec Active-HDL Lattice Edition or
- Mentor Graphics ModelSim PE (or Student Edition) or
- Mentor Graphics ModelSim SE or
- Mentor Graphics ModelSim Altera Edition or
- Mentor Graphics QuestaSim or
- Xilinx ISE Simulator 14.7 or
- Xilinx Vivado Simulator ≥ 2016.3 ³ or
- GHDL ≥ 0.34 dev and GTKWave $\geq 3.3.70$

5.1.2 Linux specific requirements:

Debian and Ubuntu specific:

- bash is configured as `/bin/sh` ([read more](#)) `dpkg-reconfigure dash`

Optional Tools on Linux:

Git The command line tools to manage Git repositories. It's possible to extend the shell prompt with Git information.

SmartGit A Git client to handle complex Git flows in a GUI.

Generic Colouriser (grc) ≥ 1.9 Colorizes outputs of foreign scripts and programs. GRC is hosted on [GitHub](#). The latest *.deb installation packages can be downloaded [here](#).

5.1.3 Mac OS specific requirements:

Bash ≥ 4.3 Mac OS is shipped with Bash 3.2. Use Homebrew to install an up-to-date Bash `brew install bash`

coreutils Mac OS' readlink program has a different behavior than the Linux version. The coreutils package installs a GNU readlink clone called greadlink. `brew install coreutils`

¹ Xilinx discontinued ISE since Oct. 2013. The last release was 14.7.

² Due to numerous bugs in the Xilinx Vivado Synthesis (incl. 2016.1), PoC can offer only a restricted Vivado support. See PoC's Vivado branch for a set of workarounds. The list of issues is documented on the [Known Issues](#) page.

³ Due to numerous bugs in the Xilinx Simulator (incl. 2016.1), PoC can offer only a restricted Vivado support. The list of issues is documented on the [Known Issues](#) page.

Optional Tools on Mac OS:

Git The command line tools to manage Git repositories. It's possible to extend the shell prompt with Git information.

SmartGit or SourceTree A Git client to handle complex Git flows in a GUI.

Generic Colouriser (grc) ≥ 1.9 Colorizes outputs of foreign scripts and programs. GRC is hosted on [GitHub](#)
`brew install Grc`

5.1.4 Windows specific requirements:

PowerShell

- **Allow local script execution** ([read more](#)) `PS> Set-ExecutionPolicy RemoteSigned`
- **PowerShell ≥ 5.0 (recommended)** PowerShell 5.0 is shipped since Windows 10. It is a part of the [Windows Management Framework 5.0](#) (WMF). Windows 7 and 8/8.1 can be updated to WMF 5.0. The package does not include **PSReadLine**, which is included in the Windows 10 PowerShell environment. Install PSReadLine manually: `PS> Install-Module PSReadline`.
- **PowerShell 4.0** PowerShell is shipped with Windows since Vista. If the required version not already included in Windows, it can be downloaded from Microsoft.com: [WMF 4.0](#)

Optional Tools on Windows:

PowerShell ≥ 4.0

- **PSReadLine** replaces the command line editing experience in PowerShell for versions 3 and up.
- **PowerShell Community Extensions (PSCX) ≥ 3.2** The latest PSCX can be downloaded from [PowerShellGallery](#) `PS> Install-Module Pscx` Note: PSCX $\geq 3.2.1$ is required for PowerShell ≥ 5.0 .

Git (MSys-Git) The command line tools to manage Git repositories.

SmartGit or SourceTree A Git client to handle complex Git flows in a GUI.

posh-git PowerShell integration for Git `PS> Install-Module posh-git`

5.2 Downloading PoC

Contents of this Page

- [Downloading from GitHub](#)
- [Downloading via `git clone`](#)
 - [On Linux](#)
 - [On OS X](#)
 - [On Windows](#)
- [Downloading via `git submodule add`](#)
 - [On Linux](#)
 - [On OS X](#)
 - [On Windows](#)

5.2.1 Downloading from GitHub

The PoC-Library can be downloaded as a zip-file from GitHub. See the following table, to choose your desired git branch.

Branch	Download Link
master	zip-file 
release	zip-file 

5.2.2 Downloading via git clone

The PoC-Library can be downloaded (cloned) with `git clone` from GitHub. GitHub offers the transfer protocols HTTPS and SSH. You should use SSH if you have a GitHub account and have already uploaded an OpenSSH public key to GitHub, otherwise use HTTPS if you have no account or you want to use login credentials.

The created folder `<GitRoot>\PoC` is used as `<PoCRoot>` in later instructions or on other pages in this documentation.

Protocol	GitHub Repository URL
HTTPS	https://github.com/VLSI-EDA/PoC.git
SSH	ssh://git@github.com:VLSI-EDA/PoC.git

On Linux

Command line instructions to clone the PoC-Library onto a Linux machine with HTTPS protocol:

```
cd GitRoot
git clone --recursive "https://github.com/VLSI-EDA/PoC.git" PoC
cd PoC
git remote rename origin github
```

Command line instructions to clone the PoC-Library onto a Linux machine machine with SSH protocol:

```
cd GitRoot
git clone --recursive "ssh://git@github.com:VLSI-EDA/PoC.git" PoC
cd PoC
git remote rename origin github
```

On OS X

Please see the Linux instructions.

On Windows

Note: All Windows command line instructions are intended for **Windows PowerShell**, if not marked otherwise. So executing the following instructions in Windows Command Prompt (**cmd.exe**) won't function or result in errors! See the [Requirements section](#) on where to download or update PowerShell.

Command line instructions to clone the PoC-Library onto a Windows machine with HTTPS protocol:

```
cd GitRoot
git clone --recursive "https://github.com/VLSI-EDA/PoC.git" PoC
cd PoC
git remote rename origin github
```

Command line instructions to clone the PoC-Library onto a Windows machine with SSH protocol:

```
cd GitRoot
git clone --recursive "ssh://git@github.com:VLSI-EDA/PoC.git" PoC
cd PoC
git remote rename origin github
```

Note: The option `--recursive` performs a recursive clone operation for all linked `git submodules`. An additional `git submodule init` and `git submodule update` call is not needed anymore.

5.2.3 Downloading via `git submodule add`

The PoC-Library is meant to be integrated into other HDL projects (preferably Git versioned projects). Therefore it's recommended to create a library folder and add the PoC-Library as a `git submodule`.

The following command line instructions will create a library folder `:file:'lib'` and clone PoC as a git submodule into the subfolder `:file:'<ProjectRoot>libPoC'`.

On Linux

Command line instructions to clone the PoC-Library onto a Linux machine with HTTPS protocol:

```
cd ProjectRoot
mkdir lib
git submodule add "https://github.com/VLSI-EDA/PoC.git" lib/PoC
cd lib/PoC
git remote rename origin github
cd ../../
git add .gitmodules lib/PoC
git commit -m "Added new git submodule PoC in 'lib/PoC' (PoC-Library)."
```

Command line instructions to clone the PoC-Library onto a Linux machine machine with SSH protocol:

```
cd ProjectRoot
mkdir lib
git submodule add "ssh://git@github.com:VLSI-EDA/PoC.git" lib/PoC
cd lib/PoC
git remote rename origin github
cd ../../
git add .gitmodules lib/PoC
git commit -m "Added new git submodule PoC in 'lib/PoC' (PoC-Library)."
```

On OS X

Please see the Linux instructions.

On Windows

Note: All Windows command line instructions are intended for **Windows PowerShell**, if not marked otherwise. So executing the following instructions in Windows Command Prompt (**cmd.exe**) won't function or result in errors! See the [Requirements section](#) on where to download or update PowerShell.

Command line instructions to clone the PoC-Library onto a Windows machine with HTTPS protocol:

```
cd <ProjectRoot>
mkdir lib | cd
git submodule add "https://github.com/VLSI-EDA/PoC.git" PoC
cd PoC
git remote rename origin github
cd ../../..
git add .gitmodules lib\PoC
git commit -m "Added new git submodule PoC in 'lib\PoC' (PoC-Library)."
```

Command line instructions to clone the PoC-Library onto a Windows machine with SSH protocol:

```
cd <ProjectRoot>
mkdir lib | cd
git submodule add "ssh://git@github.com:VLSI-EDA/PoC.git" PoC
cd PoC
git remote rename origin github
cd ../../..
git add .gitmodules lib\PoC
git commit -m "Added new git submodule PoC in 'lib\PoC' (PoC-Library)."
```

5.3 Integrating PoC into Projects

Contents of this page

- *As a Git submodule*
 - *On Linux*
 - *On OS X*
 - *On Windows*

5.3.1 As a Git submodule

The following command line instructions will integrate PoC into a existing Git repository and register PoC as a Git submodule. Therefore a directory `lib\PoC\` is created and the PoC-Library is cloned as a Git submodule into that directory.

On Linux

```
cd ProjectRoot
mkdir lib
cd lib
git submodule add https://github.com/VLSI-EDA/PoC.git PoC
cd PoC
git remote rename origin github
cd ../../..
git add .gitmodules lib\PoC
git commit -m "Added new git submodule PoC in 'lib/PoC' (PoC-Library)."
```

On OS X

Please see the Linux instructions.

On Windows

Note: All Windows command line instructions are intended for **Windows PowerShell**, if not marked otherwise. So executing the following instructions in Windows Command Prompt (**cmd.exe**) won't function or result in errors! See the *Requirements section* on where to download or update PowerShell.

```
cd ProjectRoot
mkdir lib | cd
git submodule add https://github.com/VLSI-EDA/PoC.git PoC
cd PoC
git remote rename origin github
cd ..\..
git add .gitmodules lib\PoC
git commit -m "Added new git submodule PoC in 'lib\PoC' (PoC-Library)."
```

See also:

Configuring PoC on a Local System

Create PoC's VHDL Configuration Files

5.4 Configuring PoC's Infrastructure

To explore PoC's full potential, it's required to configure some paths and synthesis or simulation tool chains. It's possible to relaunch the process at any time, for example to register new tools or to update tool versions.

Contents of this page

- *Overview*
- *The PoC-Library*
- *Git*
- *Aldec*
 - *Active-HDL*
- *Altera*
 - *Quartus*
 - *ModelSim Altera Edition*
- *Lattice*
 - *Diamond*
 - *Active-HDL Lattice Edition*
- *Mentor Graphics*
 - *QuestaSim*
- *Xilinx*
 - *ISE*

- *Vivado*
- *GHDL*
- *GTKWave*
- *Hook Files*

5.4.1 Overview

The setup process is started by invoking PoC's frontend script with the command `configure`. Please follow the instructions on screen. Use the keyboard buttons: to accept, to decline, to skip/pass a step and to accept a default value displayed in brackets.

Optionally, a vendor or tool chain name can be passed to the configuration process to launch only its configuration routines.

On Linux:

```
cd ProjectRoot
./lib/PoC/poc.sh configure
# with tool chain name
./lib/PoC/poc.sh configure Xilinx.Vivado
```

On OS X

Please see the Linux instructions.

On Windows

Note: All Windows command line instructions are intended for **Windows PowerShell**, if not marked otherwise. So executing the following instructions in Windows Command Prompt (**cmd.exe**) won't function or result in errors! See the [Requirements section](#) on where to download or update PowerShell.

```
cd ProjectRoot
.\lib\PoC\poc.ps1 configure
# with tool chain name
.\lib\PoC\poc.ps1 configure Xilinx.Vivado
```

Introduction screen:

```
PS D:\git\PoC> .\poc.ps1 configure
=====
                        The PoC-Library - Service Tool
=====
Explanation of abbreviations:
  Y - yes          P          - pass (jump to next question)
  N - no          Ctrl + C - abort (no changes are saved)
Upper case or value in '[...]' means default value
-----

Configuring PoC
PoC version: v1.0.1 (found in git)
Installation directory: D:\git\PoC (found in environment variable)
```

5.4.2 The PoC-Library

PoC itself has a fully automated configuration routine. It detects if PoC is under Git control. If so, it extracts the current version number from the latest Git tag. The installation directory is inferred from `$PoCRootDirectory` setup by `PoC.ps1` or `poc.sh`.

```
Configuring PoC
PoC version: v1.0.1 (found in git)
Installation directory: D:\git\PoC (found in environment variable)
```

5.4.3 Git

Note: Setting up Git and Git developer settings, is an advanced feature recommended for all developers interested in providing Git pull requests or patches.

```
Configuring Git
Git installation directory [C:\Program Files\Git]:
Install Git mechanisms for PoC developers? [y/N/p]: y
Install Git filters? [Y/n/p]:
Installing Git filters...
Install Git hooks? [Y/n/p]:
Installing Git hooks...
Setting 'pre-commit' hook for PoC...
```

5.4.4 Aldec

Configure the installation directory for all Aldec tools.

```
Configuring Aldec
Are Aldec products installed on your system? [Y/n/p]: Y
Aldec installation directory [C:\Aldec]:
```

Active-HDL

```
Configuring Aldec Active-HDL
Is Aldec Active-HDL installed on your system? [Y/n/p]: Y
Aldec Active-HDL version [10.3]:
Aldec Active-HDL installation directory [C:\Aldec\Active-HDL]: C:\Aldec\Active-
↪HDL-Student-Edition
```

5.4.5 Altera

Configure the installation directory for all Altera tools.

```
Configuring Altera
Are Altera products installed on your system? [Y/n/p]: Y
Altera installation directory [C:\Altera]:
```

Quartus

```
Configuring Altera Quartus
Is Altera Quartus-II or Quartus Prime installed on your system? [Y/n/p]: Y
Altera Quartus version [15.1]: 16.0
Altera Quartus installation directory [C:\Altera\16.0\quartus]:
```

ModelSim Altera Edition

```
Configuring ModelSim Altera Edition
  Is ModelSim Altera Edition installed on your system? [Y/n/p]: Y
  ModelSim Altera Edition installation directory [C:\Altera\15.0\modelsim_ae]: ↵
↵C:\Altera\16.0\modelsim_ase
```

5.4.6 Lattice

Configure the installation directory for all Lattice Semiconductor tools.

```
Configuring Lattice
  Are Lattice products installed on your system? [Y/n/p]: Y
  Lattice installation directory [D:\Lattice]:
```

Diamond

```
Configuring Lattice Diamond
  Is Lattice Diamond installed on your system? [Y/n/p]: >
  Lattice Diamond version [3.7]:
  Lattice Diamond installation directory [D:\Lattice\Diamond\3.7_x64]:
```

Active-HDL Lattice Edition

```
Configuring Active-HDL Lattice Edition
  Is Aldec Active-HDL installed on your system? [Y/n/p]: Y
  Active-HDL Lattice Edition version [10.2]:
  Active-HDL Lattice Edition installation directory [D:\Lattice\Diamond\3.7_
↵x64\active-hdl]:
```

5.4.7 Mentor Graphics

Configure the installation directory for all mentor Graphics tools.

```
Configuring Mentor
  Are Mentor products installed on your system? [Y/n/p]: Y
  Mentor installation directory [C:\Mentor]:
```

QuestaSim

```
Configuring Mentor QuestaSim
  Is Mentor QuestaSim installed on your system? [Y/n/p]: Y
  Mentor QuestaSim version [10.4d]: 10.4c
  Mentor QuestaSim installation directory [C:\Mentor\QuestaSim\10.4c]: ↵
↵C:\Mentor\QuestaSim64\10.4c
```

5.4.8 Xilinx

Configure the installation directory for all Xilinx tools.

```
Configuring Xilinx
```

```
Are Xilinx products installed on your system? [Y/n/p]: Y
Xilinx installation directory [C:\Xilinx]:
```

ISE

If an Xilinx ISE environment is available and shall be configured in PoC, then answer the following questions:

```
Configuring Xilinx ISE
```

```
Is Xilinx ISE installed on your system? [Y/n/p]: Y
Xilinx ISE installation directory [C:\Xilinx\14.7\ISE_DS]:
```

Vivado

If an Xilinx ISE environment is available and shall be configured in PoC, then answer the following questions:

```
Configuring Xilinx Vivado
```

```
Is Xilinx Vivado installed on your system? [Y/n/p]: Y
Xilinx Vivado version [2016.2]:
Xilinx Vivado installation directory [C:\Xilinx\Vivado\2016.2]:
```

5.4.9 GHDL

```
Configuring GHDL
```

```
Is GHDL installed on your system? [Y/n/p]: Y
GHDL installation directory [C:\Tools\GHDL\0.34dev]:
```

5.4.10 GTKWave

```
Configuring GTKWave
```

```
Is GTKWave installed on your system? [Y/n/p]: Y
GTKWave installation directory [C:\Tools\GTKWave\3.3.71]:
```

5.4.11 Hook Files

PoC's wrapper scripts can be customized through pre- and post-hook file. See *Wrapper Script Hook Files* for more details.

5.5 Creating my_config/my_project.vhdl

The PoC-Library needs two VHDL files for its configuration. These files are used to determine the most suitable implementation depending on the provided platform information. These files are also used to select appropriate work arounds.

5.5.1 Create my_config.vhdl

The **my_config.vhdl** file can easily be created from the template file `my_config.vhdl.template` provided by PoC in `PoCRoot\src\common`. (View source on [GitHub](#).) Copy this file into the project's source directory and rename it to `my_config.vhdl`.

This file should be included in version control systems and shared with other systems. `my_config.vhdl` defines three global constants, which need to be adjusted:

```
constant MY_BOARD      : string := "CHANGE THIS"; -- e.g. Custom, ML505, KC705, Atlys
constant MY_DEVICE     : string := "CHANGE THIS"; -- e.g. None, XC5VLX50T-1FF1136,
↳EP2SGX90FF1508C3
constant MY_VERBOSE    : boolean := FALSE;         -- activate report statements in
↳VHDL subprograms
```

The easiest way is to define a board name and set `MY_DEVICE` to `None`. So the device name is inferred from the board information stored in `PoCRoot\src\common\config.vhdl`. If the requested board is not known to PoC or it's custom made, then set `MY_BOARD` to `Custom` and `MY_DEVICE` to the full FPGA device string.

Example 1: A “Stratix II GX Audio Video Development Kit” board:

```
constant MY_BOARD      : string := "S2GXAV"; -- Stratix II GX Audio Video Development
↳Kit
constant MY_DEVICE     : string := "None";    -- infer from MY_BOARD
```

Example 2: A custom made Spartan-6 LX45 board:

```
constant MY_BOARD      : string := "Custom";
constant MY_DEVICE     : string := "XC6SLX45-3CSG324";
```

5.5.2 Create `my_project.vhdl`

The `my_project.vhdl` file can also be created from a template file `my_project.vhdl.template` provided by PoC in `PoCRoot\src\common`.

The file should to be copied into a projects source directory and renamed into `my_project.vhdl`. This file **must not** be included into version control systems – it's private to a computer. `my_project.vhdl` defines two global constants, which need to be adjusted:

```
constant MY_PROJECT_DIR      : string := "CHANGE THIS"; -- e.g. "d:/vhdl/myproject/"
↳", "/home/me/projects/myproject/"
constant MY_OPERATING_SYSTEM : string := "CHANGE THIS"; -- e.g. "WINDOWS", "LINUX"
```

Example 1: A Windows System:

```
constant MY_PROJECT_DIR      : string := "D:/git/GitHub/PoC/";
constant MY_OPERATING_SYSTEM : string := "WINDOWS";
```

Example 2: A Debian System:

```
constant MY_PROJECT_DIR      : string := "/home/paebbels/git/GitHub/PoC/";
constant MY_OPERATING_SYSTEM : string := "LINUX";
```

See also:

Running one or more testbenches The installation can be checked by running one or more of PoC's testbenches.

Running one or more netlist generation flows The installation can also be checked by running one or more of PoC's synthesis flows.

5.6 Adding IP Cores to a Project

5.6.1 Manually Addind IP Cores

Adding IP Cores to Altera Quartus

Todo: No documentation available.

Adding IP Cores to Lattice Diamond

Todo: No documentation available.

Adding IP Cores to Xilinx ISE

Todo: No documentation available.

Adding IP Cores to Xilinx Vivado

Todo: No documentation available.

5.7 Simulation

Contents of this Page

- *Overview*
- *Quick Example*
- *Vendor Specific Testbenches*
- *Running a Single Testbench*
 - *Aldec Active-HDL*
 - *Cocotb with QuestaSim backend*
 - *GHDL (plus GTKwave)*
 - *Mentor Graphics QuestaSim*
 - *Xilinx ISE Simulator*
 - *Xilinx Vivado Simulator*
- *Running a Group of Testbenches*
- *Continuous Integration (CI)*

5.7.1 Overview

The Python Infrastructure shipped with the PoC-Library can launch manual, half-automated and fully automated testbenches. The testbench can be run in command line or GUI mode. If available, the used simulator is launched with pre-configured waveform files. This can be done by invoking one of PoC's frontend script:

- **poc.sh:** `poc.sh <common options> <simulator> <module> <simulator options>`
Use this frontend script on Darwin, Linux and Unix platforms.

- **poc.psl:** `poc.psl <common options> <simulator> <module> <simulator options>` Use this frontend script Windows platforms.

Attention: All Windows command line instructions are intended for Windows PowerShell, if not marked otherwise. So executing the following instructions in Windows Command Prompt (`cmd.exe`) won't function or result in errors!

See also:

PoC Configuration See the Configuration page on how to configure PoC and your installed simulator tool chains. This is required to invoke the simulators.

Supported Simulators See the Intruction page for a list of supported simulators.

5.7.2 Quick Example

The following quick example uses the GHDL Simulator to analyze, elaborate and simulate a testbench for the module `arith_prng` (Pseudo Random Number Generator - PRNG). The VHDL file `arith_prng.vhdl` is located at `PoCRoot\src\arith` and virtually a member in the *PoC.arith* namespace. So the module can be identified by an unique name: `PoC.arith.prng`, which is passed to the frontend script.

Example 1:

```
cd PoCRoot
.\poc.psl ghdl PoC.arith.prng
```

The CLI command `ghdl` chooses *GHDL Simulator* as the simulator and passes the fully qualified PoC entity name `PoC.arith.prng` as a parameter to the tool. All required source file are gathered and compiled to an executable. Afterwards this executable is launched in CLI mode and it's outputs are displayed in console:

```

PS G:\git\PoC> .\poc.psl ghdl PoC.arith.prng

=====
The PoC-Library - Service Tool
=====
Initializing PoC-Library Service Tool for simulations
Preparing simulation environment...
Testbench: PoC.arith.prng
Running analysis for every vhd file...
Running elaboration...
Running simulation...
ghdl run messages for 'test.arith_prng_tb'
=====
POC TESTBENCH REPORT
=====
Tests      2
-1: Default test
  0: Test setup for BITS=8; SEED=0x12

Overall
Assertions 256
  failed    0
Processes  3
  active    0
Runtime    2.6 us
=====
SIMULATION RESULT = PASSED
=====

Overall Simulation Report
=====
Name      | Time | Status
-----
arith     | 0:03 | PASSED
prng
=====
Time: 0:03 Count: 1 Passed: 1 No Asserts: 0 Failed: 0 Errors: 0
=====
PS G:\git\PoC>

```

Each testbench uses PoC's simulation helper packages to count asserts and to track active stimuli and checker processes. After a completed simulation run, an report is written to STDOUT or the simulator's console. Note the

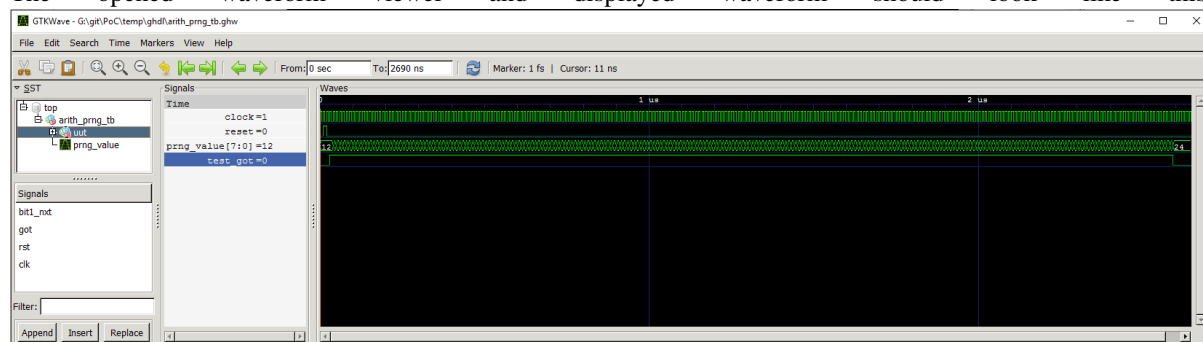
line `SIMULATION RESULT = PASSED`. For each simulated PoC entity, a line in the overall report is created. It lists the runtime per testbench and the simulation status (. . . `ERROR`, `FAILED`, `NO ASSERTS` or `PASSED`).

Example 2:

Passing an additional option `--gui` to the service tool, opens the testbench in GUI-mode. If a waveform configuration file is present (e.g. a `*.gtkw` file for GTKWave), then it is preloaded into the simulator's waveform viewer.

```
cd PoCRoot
.\poc.ps1 ghdl PoC.arith.prng --gui
```

The opened waveform viewer and displayed waveform should look like this:



5.7.3 Vendor Specific Testbenches

PoC is shipped with a set of well known FPGA development boards. This set is extended by a list of generic boards, named after each supported FPGA vendor. These generic boards can be used in simulations to select a representative FPGA of a supported device vendor. If no board or device name is passed to a testbench run, the `GENERIC` board is chosen.

Board Name	Target Board	Target Device
GENERIC	GENERIC	GENERIC
Altera	DE4	Stratix-IV 230
Lattice	ECP5Versa	ECP5-45UM
Xilinx	KC705	Kintex-7 325T

A vendor specific testbench can be launched by passing either `--board=xxx` or `--device=yyy` as an additional parameter to the PoC scripts.

```
# Example 1 - A Lattice board
.\poc.ps1 ghdl PoC.arith.prng --board=Lattice
# Example 2 - A Altera Stratix IV board
.\poc.ps1 ghdl PoC.arith.prng --board=DE4
# Example 3 - A Xilinx Kintex-7 325T device
.\poc.ps1 ghdl PoC.arith.prng --device=XC7K325T-2FFG900
```

Note: Running vendor specific testbenches may require pre-compiled vendor libraries. Some simulators are shipped with diverse pre-compiled libraries, others include scripts or user guides to pre-compile them on the target system.

PoC is shipped with a set of pre-compile scripts to offer a unified interface and common storage for all supported vendor's pre-compile procedures. See [Pre-Compiling Vendor Libraries](#).

5.7.4 Running a Single Testbench

A testbench run is supervised by PoC's `PoCRoot\py\PoC.py` service tool, which offers a consistent interface to all simulators. Unfortunately, every platform has its specialties, so a wrapper script is needed as abstraction from the host's operating system. Depending on the chosen tool chain, the wrapper script will source or invoke the vendor tool's environment scripts to pre-load the needed environment variables, paths or license file settings.

The order of options to the frontend script is as following: `<common options> <simulator> <module> <simulator options>`

The frontend offers several common options:

Common Option		Description
-q	-quiet	Quiet-mode (print nothing)
-v	-verbose	Print more messages
-d	-debug	Debug mode (print everything)
	-dryrun	Run in dry-run mode

One of the following supported simulators can be chosen, if installed and configured in PoC:

Simulator	Description
asim	Active-HDL Simulator
cocotb	Cocotb simulation using QuestaSim Simulator
ghdl	GHDL Simulator
isim	Xilinx ISE Simulator
vsim	QuestaSim Simulator or ModelSim
xsim	Xilinx Vivado Simulator

A testbench run can be interrupted by sending a keyboard interrupt to Python. On most operating systems this is done by pressing `Ctrl + C`. If PoC runs multiple testbenches at once, all finished testbenches are reported with their testbench result. The aborted testbench will be listed as errored.

Aldec Active-HDL

The command to invoke a simulation using Active-HDL is `asim` followed by a list of PoC entities. The following options are supported for Active-HDL:

Simulator Option		Description
	-board=<BOARD>	Specify a target board.
	-device=<DEVICE>	Specify a target device.
	-std=[87/93/02/08]	Select a VHDL standard. Default: 08

Note: GUI mode for Active-HDL is not yet supported.

Example:

```
cd PoCRoot
.\poc.ps1 asim PoC.arith.prng --std=93
```

Cocotb with QuestaSim backend

The command to invoke a Cocotb simulation using QuestaSim is `cocotb` followed by a list of PoC entities. The following options are supported for Cocotb:

Simulator Option		Description
	<code>-board=<BOARD></code>	Specify a target board.
	<code>-device=<DEVICE></code>	Specify a target device.
<code>-g</code>	<code>-gui</code>	Start the simulation in the QuestaSim GUI.

Note: Cocotb is currently only on Linux with QuestaSim supported. We are working to support the Windows platform and the GHDL backend.

Example:

```
cd PoCRoot
.\poc.ps1 cocotb PoC.cache.par
```

GHDL (plus GTKwave)

The command to invoke a simulation using GHDL is `ghdl` followed by a list of PoC entities. The following options are supported for GHDL:

Simulator Option		Description
	<code>-board=<BOARD></code>	Specify a target board.
	<code>-device=<DEVICE></code>	Specify a target device.
<code>-g</code>	<code>-gui</code>	Start GTKwave, if installed. Open *.gtkw, if available.
	<code>-std=[87 93 02 08]</code>	Select a VHDL standard. Default: 08

Example:

```
cd PoCRoot
.\poc.ps1 ghdl PoC.arith.prng --board=Atlys -g
```

Mentor Graphics QuestaSim

The command to invoke a simulation using QuestaSim or ModelSim is `vsim` followed by a list of PoC entities. The following options are supported for QuestaSim:

Simulator Option		Description
	<code>-board=<BOARD></code>	Specify a target board.
	<code>-device=<DEVICE></code>	Specify a target device.
<code>-g</code>	<code>-gui</code>	Start the simulation in the QuestaSim GUI.
	<code>-std=[87 93 02 08]</code>	Select a VHDL standard. Default: 08

Example:

```
cd PoCRoot
.\poc.ps1 vsim PoC.arith.prng --board=DE4 --gui
```

If QuestaSim is started in GUI mode (`--gui`), PoC will provide several Tcl files (*.do) in the simulator's working directory to recompile, restart or rerun the current simulation. The rerun command is based on the saved IP core's run script, which may default to `run -all`.

Tcl Script	Performed Tasks
recompile.do	recompile and restart
relaunch.do	recompile, restart and rerun
saveWaveform.do	save the current waveform viewer settings

Xilinx ISE Simulator

The command to invoke a simulation using ISE Simulator (isim) is `isim` followed by a list of PoC entities. The following options are supported for ISE Simulator:

Simulator Option		Description
	<code>--board=<BOARD></code>	Specify a target board.
	<code>--device=<DEVICE></code>	Specify a target device.
<code>-g</code>	<code>--gui</code>	Start the simulation in the ISE Simulator GUI (iSim).

Example:

```
cd PoCRoot
.\poc.ps1 isim PoC.arith.prng --board=Atlys -g
```

Xilinx Vivado Simulator

The command to invoke a simulation using Vivado Simulator (xsim) is `xsim` followed by a list of PoC entities. The following options are supported for Vivado Simulator:

Simulator Option		Description
	<code>--board=<BOARD></code>	Specify a target board.
	<code>--device=<DEVICE></code>	Specify a target device.
<code>-g</code>	<code>--gui</code>	Start Vivado in simulation mode.
	<code>--std=[93 08]</code>	Select a VHDL standard. Default: 93

Example:

```
cd PoCRoot
.\poc.ps1 xsim PoC.arith.prng --board=Atlys -g
```

5.7.5 Running a Group of Testbenches

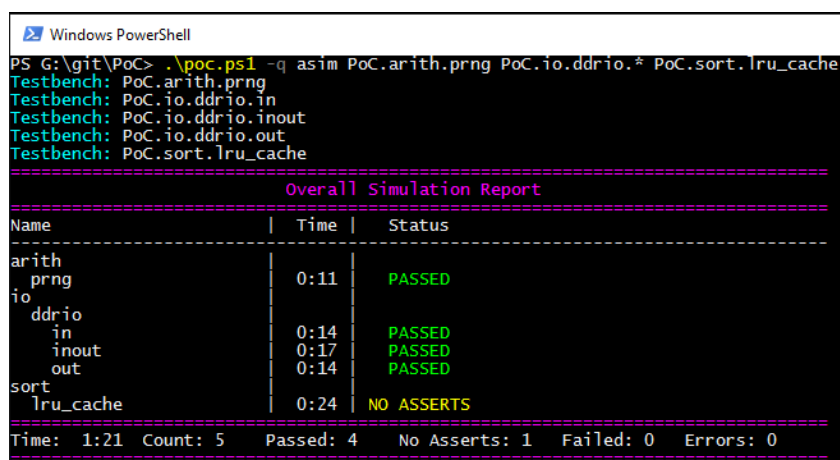
Each simulator can be invoked with a space separated list of PoC entities or a wildcard at the end of the fully qualified entity name.

Supported wildcard patterns are * and ?. Question mark refers to all entities in a PoC (sub-)namespace. Asterisk refers to all PoC entries in the current namespace and all sub-namespaces.

Examples for testbenches groups:

PoC entity list	Description
PoC.arith.prng	A single PoC entity: arith_prng
PoC.*	All entities in the whole library
PoC.io.ddrio.?	All entities in PoC.io.ddrio: ddrio_in, ddrio_inout, ddrio_out
PoC.fifo.* PoC.dstruct.*	All FIFO, cache and data-structure testbenches.

```
cd PoCRoot
.\poc.ps1 -q asim PoC.arith.prng PoC.io.ddrio.* PoC.sort.lru_cache
```



```

PS G:\git\PoC> .\poc.ps1 -q asim PoC.arith.prng PoC.io.ddrio.* PoC.sort.lru_cache
Testbench: PoC.arith.prng
Testbench: PoC.io.ddrio.in
Testbench: PoC.io.ddrio.inout
Testbench: PoC.io.ddrio.out
Testbench: PoC.sort.lru_cache

=====
Overall Simulation Report
=====
Name | Time | Status
-----|-----|-----
arith |      | 
prng | 0:11 | PASSED
io |      | 
ddrio | 0:14 | PASSED
in | 0:17 | PASSED
inout | 0:14 | PASSED
out |      | 
sort | 0:24 | NO ASSERTS
lru_cache |      | 
Time: 1:21 Count: 5 Passed: 4 No Asserts: 1 Failed: 0 Errors: 0
=====

```

Resulting output: PS G:\git\PoC>

5.7.6 Continuous Integration (CI)

All PoC testbenches are executed on every GitHub upload (push) via Travis-CI. The testsuite runs all testbenches for the virtual board `GENERIC` with an FPGA device called `GENERIC`. We can't run vendor dependent testbenches, because we can't upload the vendor simulation libraries to Travis-CI.

To reproduce the Travis-CI results on a local machine, run the following command. The `-q` option, launches the frontend in quiet mode to reduce the command line messages:

```
cd PoCRoot
.\poc.ps1 -q ghdl PoC.*
```

```

Windows PowerShell
Testbench: PoC.sort.sortnet.OddEvenSort
Testbench: PoC.sort.sortnet.OddEvenMergeSort
Testbench: PoC.sort.sortnet.Stream_Adapter
Testbench: PoC.sort.sortnet.Stream_Adapter2
Testbench: PoC.sort.lru_cache

=====
Overall Simulation Report
=====
Name | Time | Status
-----|-----|-----
arith
  addw          3:04 PASSED
  convert_bin2bcd 0:01 PASSED
  counter_bcd    0:02 PASSED
  div           0:03 PASSED
  firststone     0:02 PASSED
  prefix_and     0:01 PASSED
  prefix_or      0:01 PASSED
  prng           0:01 PASSED
  scaler        0:02 PASSED
dstruct
  deque         0:02 PASSED
  stack         0:02 PASSED
fifo
  cc_got        0:02 PASSED
  cc_got_tempput 0:02 PASSED
  ic_assembly   0:02 PASSED
  ic_got        0:02 PASSED
io
  ddrio
    in          0:01 PASSED
    inout       0:01 PASSED
    out         0:01 PASSED
  uart
    rx          0:02 PASSED
  Debounce     0:02 PASSED
mem
  lut
    Sine        0:01 NO ASSERTS
  ocram
    sdp         0:01 PASSED
misc
  gearbox
    down_cc     0:02 NO ASSERTS
    down_dc     0:02 FAILED
    up_cc       0:02 NO ASSERTS
    up_dc       0:01 FAILED
  stat
    Average     0:02 PASSED
    Histogram   0:04 NO ASSERTS
    Minimum     0:01 PASSED
    Maximum     0:01 PASSED
  sync
    Bits        0:01 PASSED
    Reset       0:01 NO ASSERTS
    Strobe      0:02 NO ASSERTS
    Vector      0:02 NO ASSERTS
    Command     0:02 NO ASSERTS
sort
  sortnet
    BitonicSort 0:56 PASSED
    OddEvenSort 2:54 PASSED
    OddEvenMergeSort 0:46 PASSED
    Stream_Adapter 0:03 PASSED
    Stream_Adapter2 0:04 PASSED
    lru_cache   0:02 NO ASSERTS
=====
Time: 9:07 Count: 41 Passed: 30 No Asserts: 9 Failed: 2 Errors: 0
=====
PS G:\git\PoC>

```

If the vendor libraries are available and pre-compiled, then it's also possible to run a CI flow for a specific vendor. This is an Altera example for the Terrasic DE4 board:

```

cd PoCRoot
.\poc.ps1 -q vsim PoC.* --board=DE4

```

See also:

PoC Configuration See the Configuration page on how to configure PoC and your installed simulator tool chains. This is required to invoke the simulators.

Latest Travis-CI Report Browse the list of branches at Travis-CI.org.

5.8 Synthesis

Contents of this Page

- *Overview*
- *Quick Example*
- *Running a single Synthesis*
 - *Altera / Intel Quartus*
 - *Lattice Diamond*
 - *Xilinx ISE Synthesis Tool (XST)*
 - *Xilinx ISE Core Generator*
 - *Xilinx Vivado Synthesis*

5.8.1 Overview

The Python infrastructure shipped with the PoC-Library can launch manual, half-automated and fully automated synthesis runs. This can be done by invoking one of PoC's frontend script:

- **poc.sh:** `poc.sh <common options> <compiler> <module> <compiler options>` Use this frontend script on Darwin, Linux and Unix platforms.
- **poc.ps1:** `poc.ps1 <common options> <compiler> <module> <compiler options>` Use this frontend script Windows platforms.

Attention: All Windows command line instructions are intended for Windows PowerShell, if not marked otherwise. So executing the following instructions in Windows Command Prompt (`cmd.exe`) won't function or result in errors!

See also:

PoC Configuration See the Configuration page on how to configure PoC and your installed synthesis tool chains. This is required to invoke the compilers.

Supported Compiler See the Intruction page for a list of supported compilers.

See also:

List of Supported FPGA Devices See this list to find a supported and well known target device.

List of Supported Development Boards See this list to find a supported and well known development board.

5.8.2 Quick Example

The following quick example uses the Xilinx Systesis Tool (XST) to synthesize a netlist for IP core `arith_prng` (Pseudo Random Number Generator - PRNG). The VHDL file `arith_prng.vhdl` is located at `PoCRoot\src\arith` and virtually a member in the *PoC.arith* namespace. So the module can be identified by an unique name: `PoC.arith.prng`, which is passed to the frontend script.

Example 1:

```
cd PoCRoot
.\poc.ps1 xst PoC.arith.prng --board=KC705
```


Simulator Option		Description
	--board=<Board>	Specify a target board.
	--device=<Device>	Specify a target device.

Example:

```
cd PoCRoot
.\poc.ps1 quartus PoC.arith.prng --board=DE4
```

Lattice Diamond

The command to invoke a synthesis using Lattice Diamond is `lse` followed by a list of PoC entities. The following options are supported for the Lattice Synthesis Engine (LSE):

Simulator Option		Description
	--board=<Board>	Specify a target board.
	--device=<Device>	Specify a target device.

Example:

```
cd PoCRoot
.\poc.ps1 lse PoC.arith.prng --board=ECP5Versa
```

Xilinx ISE Synthesis Tool (XST)

The command to invoke a synthesis using Xilinx ISE Synthesis is `xst` followed by a list of PoC entities. The following options are supported for the Xilinx Synthesis Tool (XST):

Simulator Option		Description
	--board=<Board>	Specify a target board.
	--device=<Device>	Specify a target device.

Example:

```
cd PoCRoot
.\poc.ps1 xst PoC.arith.prng --board=KC705
```

Xilinx ISE Core Generator

The command to invoke an IP core generation using Xilinx Core Generator is `coregen` followed by a list of PoC entities. The following options are supported for Core Generator (CG):

Simulator Option		Description
	--board=<Board>	Specify a target board.
	--device=<Device>	Specify a target device.

Example:

```
cd PoCRoot
.\poc.ps1 coregen PoC.xil.mig.Atlys_1x128 --board=Atlys
```

Xilinx Vivado Synthesis

The command to invoke a synthesis using Xilinx Vivado Synthesis is vivado followed by a list of PoC entities. The following options are supported for Vivado Synthesis (Synth):

Simulator Option		Description
	--board=<Board>	Specify a target board.
	--device=<Device>	Specify a target device.

Example:

```
cd PoCRoot
.\poc.ps1 vivado PoC.arith.prng --board=KC705
```

5.9 Project Management

5.9.1 Overview

5.9.2 Solutions

5.9.3 Projects

5.10 Pre-Compiling Vendor Libraries

Contents of this Page

- *Overview*
- *Supported Simulators*
- *FPGA Vendor's Primitive Libraries*
 - *Altera*
 - *Lattice*
 - *Xilinx ISE*
 - *Xilinx Vivado*
- *Third-Party Libraries*
 - *OSVVM*
 - *UVVM*
- *Simulator Adapters*
 - *Cocotb*

5.10.1 Overview

Running vendor specific testbenches may require pre-compiled vendor libraries. Some vendors ship their simulators with diverse pre-compiled libraries, but these don't include primitive libraries from hardware vendors. More over, many auxillary libraries are outdated. Hardware vendors ship their tool chains with pre-compile scripts or user guides to pre-compile the primitive libraries for a list of supported simulators on a target system.

PoC is shipped with a set of pre-compile scripts to offer a unified interface and common storage for all supported vendor's pre-compile procedures. The scripts are located in `\tools\precompile\` and the output is stored in `\temp\precompiled\<Simulator>\<Library>`.

5.10.2 Supported Simulators

The current set of pre-compile scripts support these simulators:

Vendor	Simulator and Edition	Altera	Lattice	Xilinx (ISE)	Xilinx (Vivado)
20. Gingold	GHDL with --std=93c GHDL with --std=08	yes yes	yes yes	yes yes	yes yes
Aldec	Active-HDL (or Student Ed.) Active-HDL Lattice Ed. Reviera-PRO	planned planned planned	planned shipped planned	planned planned planned	planned planned planned
Mentor	ModelSim PE (or Student Ed.) ModelSim SE ModelSim Altera Ed. QuestaSim	yes yes shipped yes	yes yes yes yes	yes yes yes yes	yes yes yes yes
Xilinx	ISE Simulator Vivado Simulator			shipped not supported	not supported shipped

5.10.3 FPGA Vendor's Primitive Libraries

Altera

Note: The Altera Quartus tool chain needs to be configured in PoC. See [Configuring PoC's Infrastructure](#) for further details.

On Linux

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-altera.sh --all
# Example 2 - Compile only for GHDL and VHDL-2008
./tools/precompile/compile-altera.sh --ghdl --vhdl2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<code>--help</code>	Print embedded help page(s).
-c	<code>--clean</code>	Clean-up directories.
-a	<code>--all</code>	Compile for all simulators.
	<code>--ghdl</code>	Compile for GHDL.
	<code>--questa</code>	Compile for QuestaSim.
	<code>--vhdl93</code>	GHDL only: Compile only for VHDL-93.
	<code>--vhdl2008</code>	GHDL only: Compile only for VHDL-2008.

On Windows

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-altera.ps1 -All
# Example 2 - Compile only for GHDL and VHDL-2008
.\tools\precompile\compile-altera.ps1 -GHDL -VHDL2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<code>-Help</code>	Print embedded help page(s).
-c	<code>-Clean</code>	Clean-up directories.
-a	<code>-All</code>	Compile for all simulators.
	<code>-GHDL</code>	Compile for GHDL.
	<code>-Questa</code>	Compile for QuestaSim.
	<code>-VHDL93</code>	GHDL only: Compile only for VHDL-93.
	<code>-VHDL2008</code>	GHDL only: Compile only for VHDL-2008.

Lattice

Note: The Lattice Diamond tool chain needs to be configured in PoC. See [Configuring PoC's Infrastructure](#) for further details.

On Linux

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-lattice.sh --all
# Example 2 - Compile only for GHDL and VHDL-2008
./tools/precompile/compile-lattice.sh --ghdl --vhdl2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<code>--help</code>	Print embedded help page(s).
-c	<code>--clean</code>	Clean-up directories.
-a	<code>--all</code>	Compile for all simulators.
	<code>--ghdl</code>	Compile for GHDL.
	<code>--questa</code>	Compile for QuestaSim.
	<code>--vhdl93</code>	GHDL only: Compile only for VHDL-93.
	<code>--vhdl2008</code>	GHDL only: Compile only for VHDL-2008.

On Windows

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-lattice.ps1 -All
# Example 2 - Compile only for GHDL and VHDL-2008
.\tools\precompile\compile-lattice.ps1 -GHDL -VHDL2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>-Help</i>	Print embedded help page(s).
-c	<i>-Clean</i>	Clean-up directories.
-a	<i>-All</i>	Compile for all simulators.
	<i>-GHDL</i>	Compile for GHDL.
	<i>-Questa</i>	Compile for QuestaSim.
	<i>-VHDL93</i>	GHDL only: Compile only for VHDL-93.
	<i>-VHDL2008</i>	GHDL only: Compile only for VHDL-2008.

Xilinx ISE

Note: The Xilinx ISE tool chain needs to be configured in PoC. See *Configuring PoC's Infrastructure* for further details.

On Linux

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-xilinx-ise.sh --all
# Example 2 - Compile only for GHDL and VHDL-2008
./tools/precompile/compile-xilinx-ise.sh --ghdl --vhd12008
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>--help</i>	Print embedded help page(s).
-c	<i>--clean</i>	Clean-up directories.
-a	<i>--all</i>	Compile for all simulators.
	<i>--ghdl</i>	Compile for GHDL.
	<i>--questa</i>	Compile for QuestaSim.
	<i>--vhd193</i>	GHDL only: Compile only for VHDL-93.
	<i>--vhd12008</i>	GHDL only: Compile only for VHDL-2008.

On Windows

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-xilinx-ise.ps1 -All
# Example 2 - Compile only for GHDL and VHDL-2008
.\tools\precompile\compile-xilinx-ise.ps1 -GHDL -VHDL2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>-Help</i>	Print embedded help page(s).
-c	<i>-Clean</i>	Clean-up directories.
-a	<i>-All</i>	Compile for all simulators.
	<i>-GHDL</i>	Compile for GHDL.
	<i>-Questa</i>	Compile for QuestaSim.
	<i>-VHDL93</i>	GHDL only: Compile only for VHDL-93.
	<i>-VHDL2008</i>	GHDL only: Compile only for VHDL-2008.

Xilinx Vivado

Note: The Xilinx Vivado tool chain needs to be configured in PoC. See [Configuring PoC's Infrastructure](#) for further details.

On Linux

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-xilinx-vivado.sh --all
# Example 2 - Compile only for GHDL and VHDL-2008
./tools/precompile/compile-xilinx-vivado.sh --ghdl --vhdl2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>--help</i>	Print embedded help page(s).
-c	<i>--clean</i>	Clean-up directories.
-a	<i>--all</i>	Compile for all simulators.
	<i>--ghdl</i>	Compile for GHDL.
	<i>--questa</i>	Compile for QuestaSim.
	<i>--vhdl93</i>	GHDL only: Compile only for VHDL-93.
	<i>--vhdl2008</i>	GHDL only: Compile only for VHDL-2008.

On Windows

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-xilinx-vivado.ps1 -All
# Example 2 - Compile only for GHDL and VHDL-2008
.\tools\precompile\compile-xilinx-vivado.ps1 -GHDL -VHDL2008
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>-Help</i>	Print embedded help page(s).
-c	<i>-Clean</i>	Clean-up directories.
-a	<i>-All</i>	Compile for all simulators.
	<i>-GHDL</i>	Compile for GHDL.
	<i>-Questa</i>	Compile for QuestaSim.
	<i>-VHDL93</i>	GHDL only: Compile only for VHDL-93.
	<i>-VHDL2008</i>	GHDL only: Compile only for VHDL-2008.

5.10.4 Third-Party Libraries

OSVVM

On Linux

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-osvvm.sh --all
# Example 2 - Compile only for GHDL
./tools/precompile/compile-osvvm.sh --ghdl
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>--help</i>	Print embedded help page(s).
-c	<i>--clean</i>	Clean-up directories.
-a	<i>--all</i>	Compile for all simulators.
	<i>--ghdl</i>	Compile for GHDL.
	<i>--questa</i>	Compile for QuestaSim.

On Windows

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-osvvm.ps1 -All
# Example 2 - Compile only for GHDL
.\tools\precompile\compile-osvvm.ps1 -GHDL
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>-Help</i>	Print embedded help page(s).
-c	<i>-Clean</i>	Clean-up directories.
-a	<i>-All</i>	Compile for all simulators.
	<i>-GHDL</i>	Compile for GHDL.
	<i>-Questa</i>	Compile for QuestaSim.

UVVM

On Linux

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-uvvm.sh --all
# Example 2 - Compile only for GHDL
./tools/precompile/compile-uvvm.sh --ghdl
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>--help</i>	Print embedded help page(s).
-c	<i>--clean</i>	Clean-up directories.
-a	<i>--all</i>	Compile for all simulators.
	<i>--ghdl</i>	Compile for GHDL.
	<i>--questa</i>	Compile for QuestaSim.

On Windows

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-uvvm.ps1 -All
# Example 2 - Compile only for GHDL
.\tools\precompile\compile-uvvm.ps1 -GHDL
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>-Help</i>	Print embedded help page(s).
-c	<i>-Clean</i>	Clean-up directories.
-a	<i>-All</i>	Compile for all simulators.
	<i>-GHDL</i>	Compile for GHDL.
	<i>-Questa</i>	Compile for QuestaSim.

5.10.5 Simulator Adapters

Cocotb

On Linux

Attention: This is an experimental compile script.

```
# Example 1 - Compile for all Simulators
./tools/precompile/compile-cocotb.sh --all
# Example 2 - Compile only for GHDL
./tools/precompile/compile-cocotb.sh --ghdl
```

List of command line arguments:

Common Option		Parameter Description
-h	<i>--help</i>	Print embedded help page(s).
-c	<i>--clean</i>	Clean-up directories.
-a	<i>--all</i>	Compile for all simulators.
	<i>--ghdl</i>	Compile for GHDL.
	<i>--questa</i>	Compile for QuestaSim.

On Windows

Attention: This is an experimental compile script.

```
# Example 1 - Compile for all Simulators
.\tools\precompile\compile-cocotb.ps1 -All
# Example 2 - Compile only for GHDL
.\tools\precompile\compile-cocotb.ps1 -GHDL
```

List of command line arguments:

Common Option		Parameter Description
-h	-Help	Print embedded help page(s).
-c	-Clean	Clean-up directories.
-a	-All	Compile for all simulators.
	-GHDL	Compile for GHDL.
	-Questa	Compile for QuestaSim.

5.11 Miscellaneous

The directory `PoCRoot\tools\` contains several tools and addons to ease the work with the PoC-Library and VHDL.

5.11.1 GNU Emacs

Todo: No documentation available.

5.11.2 Git

- `git-alias.setup.ps1/git-alias.setup.sh` registers new global aliases in Git
 - `git tree` - Prints the colored commit tree into the console
 - `git treea` - Prints the colored commit tree into the console

```
git config --global alias.tree 'log --decorate --pretty=oneline --abbrev-
↪commit --date-order --graph'
git config --global alias.tree 'log --decorate --pretty=oneline --abbrev-
↪commit --date-order --graph --all'
```

Browse the [Git](#) directory.

5.11.3 Notepad++

The PoC-Library is shipped with syntax highlighting rules for [Notepad++](#). The following additional file types are supported:

- PoC Configuration Files (*.ini)
- PoC *.Files Files* (.files)
- PoC *.Rules Files* (.rules)
- Xilinx User Constraint Files (*.ucf): Syntax Highlighting - Xilinx UCF

Browse the [Notepad++](#) directory.

PoC defines a set of on-chip interfaces described in the next sections.

6.1 Command-Status-Error (PoC.CSE) Interface

Todo: Define the PoC.CSE (Command-Status-Error) interface used in ...

6.2 PoC.FIFO Interface

Todo: Define the PoC.FIFO interface (writer and reader) used in `PoC.fifo.*` ...

6.3 PoC.Mem Interface

PoC.Mem is a single-cycle, pipelined memory interface used by various memory controllers and related components like caches. Memory accesses are always word aligned, and during writes a mask defines which bytes are actually written to the memory (if supported by the memory controller).

6.3.1 Configuration

Each entity may have an individual configuration, especially if it has two PoC.Mem interfaces or if it adapts between PoC.Mem and another interface.

The typical configuration parameters are:

Parameter	Description
ADDR_BITS or A_BITS	Number of address bits. Each address identifies exactly one memory word.
DATA_BITS or D_BITS	Size of a memory word in bits. DATA_BITS must be divisible by 8.

A memory word consists of DATA_BITS/8 bytes.

Individual bytes are only addressed during writes by the write mask. The write mask has one mask-bit for each byte in a memory word.

For example, a 1 KiByte memory with a 32-bit datapath has the following configuration:

- 4 bytes per memory word,
- ADDR_BITS=8 because $\log_2(1 \text{ KiByte}/4 \text{ bytes}) = 8$, and
- DATA_BITS=32 which is the datapath size in bits.

6.3.2 Interface signals

The following signal names are typically prefixed in the port list of a concrete entity to separate the PoC.Mem interface from other interfaces of the entity. Moreover, clock and reset may be shared with other interfaces of the entity.

The PoC.Mem interface consists of the following signals:

Signal	Description
clk	The clock. All other signals are synchronous to the rising edge of this clock.
rst	High-active synchronous reset.
rdy	High-active ready for request.
req	High-active request.
write	'1' if write request, '0' if read request
addr	The (word) address.
wdata	The data to be written to the memory.
wmask (optional)	Write-mask, for each byte: '0' = write byte, '1' = mask byte from write. Signal/port is omitted if write mask is not supported.
rstb	High-active read-strobe.
rdata	The read-data returned from the memory.

The interface is actually splitted into two parts:

- the request part: signals `rdy`, `req`, `write`, `addr`, `wdata` and `wmask`, and
- the read-reply part: signals `rstb` and `rdata`.

6.3.3 Operation

The request and the read-reply part operate indepent of each other to support pipelined reading from memory. The pipeline depth is defined by the actual memory controller. If a user application does support only a specific number of outstanding reads, then the application must limit the number of issued reads on its own.

Requests

If `req` is low, then no request is issued to the memory in the current clock cycle. The state of the signals `write`, `addr`, `wdata` and `wmask` doesn't care.

If `req` is high, then a request is issued to the memory in the current clock cycle as given by `write`, `addr`, `wdata` and `wmask`. The request will be accepted by the memory, if `rdy` is high in the same clock cycle, otherwise the request will be ignored. `wdata` and `wmask` doesn't care if a read request is issued.

`rdy` does not depend on `req` in the current clock cycle. `rdy` may go low in the following clock cycle after a request has been issued or a synchronous reset has been applied.

Read Replies

If `rstb` is high in the current clock cycle, then `rdata` delivers the requested read data (read reply). Otherwise, if `rstb` is low, then `rdata` is unknown. The user application has to immediately handle the incoming read data, because it cannot signal ready or acknowledge.

After issuing a read request, the memory responds with a read reply in the following clock cycle (i.e. synchronous read) or any later clock cycle depending on the pipeline depth. For each read request, a read reply is generated. Read requests are not reordered.

6.4 PoC.Stream Interface

Todo: Define the `PoC.Stream` interface used in `PoC.net.*` and `PoC.bus.stream.*` ...

Namespace for Packages:

7.1 Common Packages

These are common packages. . . .

7.1.1 components

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.2 context

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.3 config

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.4 fileio

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.5 math

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.6 strings

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.7 utils

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.1.8 vectors

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2 Simulation Packages

7.2.1 sim_types

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam

voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.2 sim_global (VHDL-93)

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.3 sim_global (VHDL-2008)

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.4 sim_unprotected (VHDL-93)

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.5 sim_protected (VHDL-2008)

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.6 simulation (VHDL-93)

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.7 simulation (VHDL-2008)

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur

sadipscing elit, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.2.8 sim_waveform

Lorem ipsum dolor sit amet, consetetur sadipscing elit, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elit, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

Namespaces for Entities:

7.3 PoC.alt

Todo: This namespace is reserved for Altera specific entities.

7.4 PoC.arith

These are arithmetic entities....

Package

PoC.arith Package

Entities

- *PoC.arith.addw*
- *PoC.arith.carrychain_inc*
- *PoC.arith.convert_bin2bcd*
- *PoC.arith.counter_bcd*
- *PoC.arith.counter_free*
- *PoC.arith.counter_gray*
- *PoC.arith.counter_ring*
- *PoC.arith.div*
- *PoC.arith.firstone*
- *PoC.arith.muls_wide*
- *PoC.arith.prefix_and*
- *PoC.arith.prefix_or*
- *PoC.arith.prng*
- *PoC.arith.same*
- *PoC.arith.scaler*
- *PoC.arith.shifter_barrel*
- *PoC.arith.sqrt*

7.4.1 PoC.arith Package

This package holds all component declarations for this namespace.

Exported Enumerations

- `tArch`
- `tBlocking`
- `tSkipping`

Exported Functions

- `arith_div_latency`

Exported Components

- *PoC.arith.addw*
- `PoC.arith.carrychain_inc_xilinx`
- *PoC.arith.counter_bcd*
- *PoC.arith.counter_gray*
- *PoC.arith.div*
- *PoC.arith.firstone*
- `PoC.arith.inc_ovcy_xilinx`
- *PoC.arith.muls_wide*
- `PoC.arith.prefix_and_xilinx`
- `PoC.arith.prefix_or_xilinx`
- *PoC.arith.prng*
- *PoC.arith.same*
- *PoC.arith.sqrt*

Source file: `arith.pkg.vhdl`

7.4.2 PoC.arith.addw

Implements wide addition providing several options all based on an adaptation of a carry-select approach.

References:

- Hong Diep Nguyen and Bogdan Pasca and Thomas B. Preusser: FPGA-Specific Arithmetic Optimizations of Short-Latency Adders, FPL 2011. -> ARCH: AAM, CAI, CCA -> SKIPPING: CCC
- Marcin Rogawski, Kris Gaj and Ekawat Homsirikamol: A Novel Modular Adder for One Thousand Bits and More Using Fast Carry Chains of Modern FPGAs, FPL 2014. -> ARCH: PAI -> SKIPPING: PPN_KS, PPN_BK

Entity Declaration:

```
1 entity arith_addw is
2   generic (
3     N : positive;           -- Operand Width
4     K : positive;           -- Block Count
5
6     ARCH      : tArch      := AAM;      -- Architecture
7     BLOCKING   : tBlocking := DFLT;      -- Blocking Scheme
8     SKIPPING   : tSkipping := CCC;       -- Carry Skip Scheme
9     P_INCLUSIVE : boolean   := false     -- Use Inclusive Propagate, i.e. c^1
10  );
11  port (
12    a, b : in std_logic_vector(N-1 downto 0);
13    cin  : in std_logic;
14
15    s      : out std_logic_vector(N-1 downto 0);
16    cout   : out std_logic
17  );
18 end entity;
```

Source file: arith/arith_addw.vhdl

7.4.3 PoC.arith.carrychain_inc

This is a generic carry-chain abstraction for increment by one operations.

$$Y \leq X + (0 \dots 0) \& Cin$$
Entity Declaration:

```
1 entity arith_carrychain_inc is
2   generic (
3     BITS      : positive
4  );
5  port (
6    X : in std_logic_vector(BITS - 1 downto 0);
7    CIn : in std_logic := '1';
8    Y : out std_logic_vector(BITS - 1 downto 0)
9  );
10 end entity;
```

Source file: arith/arith_carrychain_inc.vhdl

7.4.4 PoC.arith.convert_bin2bcd

Todo: No documentation available.

Entity Declaration:

```
1 entity arith_convert_bin2bcd is
2   generic (
3     BITS      : positive := 8;
```

(continues on next page)

(continued from previous page)

```

4     DIGITS      : positive := 3;
5     RADIX       : positive := 2
6 );
7 port (
8     Clock       : in  std_logic;
9     Reset       : in  std_logic;
10
11     Start       : in  std_logic;
12     Busy        : out std_logic;
13
14     Binary       : in  std_logic_vector (BITS - 1 downto 0);
15     IsSigned     : in  std_logic := '0';
16     BCDDigits    : out T_BCD_VECTOR (DIGITS - 1 downto 0);
17     Sign         : out std_logic
18 );
19 end entity;
```

Source file: `arith/arith_convert_bin2bcd.vhdl`

7.4.5 PoC.arith.counter_bcd

Counter with output in binary coded decimal (BCD). The number of BCD digits is configurable by `DIGITS`.

All control signals (reset `rst`, increment `inc`) are high-active and synchronous to clock `clk`. The output `val` is the current counter state. Groups of 4 bit represent one BCD digit. The lowest significant digit is specified by `val(3 downto 0)`.

Todo:

- implement a `dec` input for decrementing
- implement a `load` input to load a value

Entity Declaration:

```

1 entity arith_counter_bcd is
2     generic (
3         DIGITS : positive -- Number of BCD digits
4     );
5     port (
6         clk : in  std_logic;
7         rst : in  std_logic; -- Reset to 0
8         inc : in  std_logic; -- Increment
9         val : out T_BCD_VECTOR (DIGITS-1 downto 0) -- Value output
10    );
11 end entity;
```

Source file: `arith/arith_counter_bcd.vhdl`

7.4.6 PoC.arith.counter_free

Implements a free-running counter that generates a strobe signal every `DIVIDER`-th cycle the increment input was asserted. There is deliberately no output or specification of the counter value so as to allow an implementation to optimize as much as possible.

The implementation guarantees a strobe output directly from a register. It is asserted exactly for one clock after `DIVIDER` cycles of an asserted increment input have been observed.

Entity Declaration:

```
1 entity arith_counter_free is
2   generic (
3     DIVIDER : positive
4   );
5   port (
6     -- Global Control
7     clk : in std_logic;
8     rst : in std_logic;
9
10    inc : in std_logic;
11    stb : out std_logic           -- End-of-Period Strobe
12  );
13 end entity arith_counter_free;
```

Source file: arith/arith_counter_free.vhdl

7.4.7 PoC.arith.counter_gray

Todo: No documentation available.

Entity Declaration:

```
1 entity arith_counter_gray is
2   generic (
3     BITS : positive;           -- Bit width of the counter
4     INIT : natural             := 0      -- Initial/reset counter value
5   );
6   port (
7     clk : in std_logic;
8     rst : in std_logic;        -- Reset to INIT value
9     inc : in std_logic;        -- Increment
10    dec : in std_logic         := '0';   -- Decrement
11    val : out std_logic_vector (BITS-1 downto 0); -- Value output
12    cry : out std_logic        -- Carry output
13  );
14 end entity arith_counter_gray;
```

Source file: arith/arith_counter_gray.vhdl

7.4.8 PoC.arith.counter_ring

This module implements an up/down ring-counter with loadable initial value (seed) on reset. The counter can be configured to a Johnson counter by enabling `INVERT_FEEDBACK`. The number of counter bits is configurable with `BITS`.

Entity Declaration:

```
1 entity arith_counter_ring is
2   generic (
3     BITS : positive;
4     INVERT_FEEDBACK : boolean := FALSE           -- FALSE_
5   )
6   --> ring counter; TRUE -> johnson counter
```

(continues on next page)

(continued from previous page)

```

5   );
6   port (
7       Clock    : in  std_logic;           -- Clock
8       Reset    : in  std_logic;           -- Reset
9       seed     : in  std_logic_vector(BITS - 1 downto 0) := (others => '0'); --
↳initial counter vector / load value
10      inc      : in  std_logic              := '0';           --
↳increment counter
11      dec      : in  std_logic              := '0';           --
↳decrement counter
12      value    : out std_logic_vector(BITS - 1 downto 0)      --
↳counter value
13   );
14 end entity;

```

Source file: arith/arith_counter_ring.vhdl

7.4.9 PoC.arith.div

Implementation of a Non-Performing restoring divider with a configurable radix. The multi-cycle division is controlled by ‘start’ / ‘rdy’. A new division is started by asserting ‘start’. The result $Q = A/D$ is available when ‘rdy’ returns to ‘1’. A division by zero is identified by output Z. The Q and R outputs are undefined in this case.

Entity Declaration:

```

1  entity arith_div is
2      generic (
3          A_BITS      : positive;           -- Dividend Width
4          D_BITS      : positive;           -- Divisor Width
5          RAPOW       : positive := 1;      -- Power of Compute Radix (2**RAPOW)
6          PIPELINED   : boolean  := false  -- Computation Pipeline
7      );
8      port (
9          -- Global Reset/Clock
10         clk : in  std_logic;
11         rst : in  std_logic;
12
13         -- Ready / Start
14         start : in  std_logic;
15         ready : out std_logic;
16
17         -- Arguments / Result (2's complement)
18         A : in  std_logic_vector(A_BITS-1 downto 0); -- Dividend
19         D : in  std_logic_vector(D_BITS-1 downto 0); -- Divisor
20         Q : out std_logic_vector(A_BITS-1 downto 0); -- Quotient
21         R : out std_logic_vector(D_BITS-1 downto 0); -- Remainder
22         Z : out std_logic  -- Division by Zero
23     );
24 end entity arith_div;

```

Source file: arith/arith_div.vhdl

7.4.10 PoC.arith.firstone

Computes from an input word, a word of the same size that has, at most, one bit set. The output contains a set bit at the position of the rightmost set bit of the input if and only if such a set bit exists in the input.

A typical use case for this computation would be an arbitration over requests with a fixed and strictly ordered priority. The terminology of the interface assumes this use case and provides some useful extras:

- Set `tin` (no input token) to disallow grants altogether.
- Read `tout` (unused token) to see whether or any grant was issued.
- Read `bin` to obtain the binary index of the rightmost detected one bit. The index starts at zero (0) in the rightmost bit position.

This implementation uses carry chains for wider implementations.

Entity Declaration:

```
1 entity arith_firstone is
2   generic (
3     N : positive                                -- Length of Token Chain
4   );
5   port (
6     tin  : in  std_logic := '1';                -- Enable: Fed Token
7     rqst : in  std_logic_vector(N-1 downto 0);   -- Request: Token Requests
8     grnt : out std_logic_vector(N-1 downto 0);    -- Grant: Token Output
9     tout : out std_logic;                        -- Inactive: Unused Token
10    bin  : out std_logic_vector(log2ceil(N)-1 downto 0) -- Binary Grant Index
11  );
12 end entity arith_firstone;
```

Source file: `arith/arith_firstone.vhdl`

7.4.11 PoC.arith.muls_wide

Signed wide multiplication spanning multiple DSP or MULT blocks. Small partial products are calculated through LUTs. For detailed documentation see below.

Entity Declaration:

Source file: `arith/arith_muls_wide.vhdl`

7.4.12 PoC.arith.prefix_and

Prefix AND computation: `y(i) <= '1'` when `x(i downto 0) = (i downto 0 => '1')` else `'0'`; This implementation uses carry chains for wider implementations.

Entity Declaration:

```
1 entity arith_prefix_and is
2   generic (
3     N : positive
4   );
5   port (
6     x : in  std_logic_vector(N-1 downto 0);
7     y : out std_logic_vector(N-1 downto 0)
8   );
9 end entity;
```

Source file: `arith/arith_prefix_and.vhdl`

7.4.13 PoC.arith.prefix_or

Prefix OR computation: $y(i) \leq '0'$ when $x(i \text{ downto } 0) = (i \text{ downto } 0 \Rightarrow '0')$ else $'1'$; This implementation uses carry chains for wider implementations.

Entity Declaration:

```

1 entity arith_prefix_or is
2   generic (
3     N : positive
4   );
5   port (
6     x : in std_logic_vector(N-1 downto 0);
7     y : out std_logic_vector(N-1 downto 0)
8   );
9 end entity;
```

Source file: arith/arith_prefix_or.vhdl

7.4.14 PoC.arith.prng

This module implements a Pseudo-Random Number Generator (PRNG) with configurable bit count (BITS). This module uses an internal list of FPGA optimized polynomials from 3 to 168 bits. The polynomials have at most 5 tap positions, so that long shift registers can be inferred instead of single flip-flops.

The generated number sequence includes the value all-zeros, but not all-ones.

Entity Declaration:

```

1 entity arith_prng is
2   generic (
3     BITS : positive      := 32;
4     SEED : std_logic_vector := "0"
5   );
6   port (
7     clk : in std_logic;
8     rst : in std_logic;      -- reset value to initial seed
9     got : in std_logic;      -- the current value has been_
10    -- got, and a new value should be calculated
11    val : out std_logic_vector(BITS - 1 downto 0) -- the pseudo-random number
12  );
13 end entity;
```

Source file: arith/arith_prng.vhdl

7.4.15 PoC.arith.same

This circuit may, for instance, be used to detect the first sign change and, thus, the range of a two's complement number.

These components may be chained by using the output of the predecessor as guard input. This chaining allows to have intermediate results available while still ensuring the use of a fast carry chain on supporting FPGA architectures. When chaining, make sure to overlap both vector slices by one bit position as to avoid an undetected sign change between the slices.

Entity Declaration:

```
1 entity arith_same is
2   generic (
3     N : positive                                -- Input width
4   );
5   port (
6     g : in std_logic := '1';                  -- Guard Input (!g => !y)
7     x : in std_logic_vector(N-1 downto 0);    -- Input Vector
8     y : out std_logic                          -- All-same Output
9   );
10 end entity;
```

Source file: arith/arith_same.vhdl

7.4.16 PoC.arith.scaler

A flexible scaler for fixed-point values. The scaler is implemented for a set of multiplier and divider values. Each individual scaling operation can arbitrarily select one value from each these sets.

The computation calculates: $\text{unsigned}(\text{arg}) * \text{MULS}(\text{msel}) / \text{DIVS}(\text{dsel})$ rounded to the nearest (tie upwards) fixed-point result of the same precision as *arg*.

The computation is started by asserting *start* to high for one cycle. If a computation is running, it will be restarted. The completion of a calculation is signaled via *done*. *done* is high when no computation is in progress. The result of the last scaling operation is stable and can be read from *res*. The weight of the LSB of *res* is the same as the LSB of *arg*. Make sure to tap a sufficient number of result bits in accordance to the highest scaling ratio to be used in order to avoid a truncation overflow.

Entity Declaration:

```
1 entity arith_scaler is
2   generic (
3     MULS : T_POSVEC := (0 => 1); -- The set of multipliers to choose from in_
↪ scaling operations.
4     DIVS : T_POSVEC := (0 => 1)  -- The set of divisors to choose from in scaling_
↪ operations.
5   );
6   port (
7     clk : in std_logic;
8     rst : in std_logic;
9
10    start : in std_logic; -- Start of Computation
11    arg : in std_logic_vector; -- Fixed-point value to be scaled
12    msel : in std_logic_vector(log2ceil(MULS'length)-1 downto 0) := (others => '0
↪ ');
13    dsel : in std_logic_vector(log2ceil(DIVS'length)-1 downto 0) := (others => '0
↪ ');
14
15    done : out std_logic; -- Completion
16    res : out std_logic_vector -- Result
17  );
18 end entity arith_scaler;
```

Source file: arith/arith_scaler.vhdl

7.4.17 PoC.arith.shifter_barrel

This Barrel-Shifter supports:

- shifting and rotating
- right and left operations
- arithmetic and logic mode (only valid for shift operations)

This is equivalent to the CPU instructions: SLL, SLA, SRL, SRA, RL, RR

Entity Declaration:

```

1 entity arith_shifter_barrel is
2   generic (
3     BITS          : positive      := 32
4   );
5   port (
6     Input          : in  std_logic_vector(BITS - 1 downto 0);
7     ShiftAmount    : in  std_logic_vector(log2ceilnz(BITS) - 1 downto 0);
8     ShiftRotate    : in  std_logic;
9     LeftRight      : in  std_logic;
10    ArithmeticLogic : in  std_logic;
11    Output          : out std_logic_vector(BITS - 1 downto 0)
12  );
13 end entity;
```

Source file: arith/arith_shifter_barrel.vhdl

7.4.18 PoC.arith.sqrt

Iterative Square Root Extractor.

Its computation requires $(N+1)/2$ steps for an argument bit width of N .

Entity Declaration:

```

1 entity arith_sqrt is
2   generic (
3     N : positive -- := 8                                -- Bit Width of Argument
4   );
5   port (
6     -- Global Control
7     rst : in std_logic;                                -- Reset (synchronous)
8     clk : in std_logic;                                -- Clock
9
10    -- Inputs
11    arg : in std_logic_vector(N-1 downto 0); -- Radicand
12    start : in std_logic;                    -- Start Strobe
13
14    -- Outputs
15    sqrt : out std_logic_vector((N-1)/2 downto 0); -- Result
16    rdy : out std_logic;                        -- Ready / Done
17  );
18 end entity arith_sqrt;
```

Source file: arith/arith_sqrt.vhdl

7.5 PoC.bus

These are bus entities....

Sub-namespaces

- *PoC.bus.stream*
- *PoC.bus.wb*

Entities

- *PoC.bus.Arbitrator*

7.5.1 PoC.bus.stream

PoC.Stream modules ...

PoC.bus.stream Package

Source file: `stream.pkg.vhdl`

PoC.bus.stream.Buffer

This module implements a generic buffer (FIFO) for the *PoC.Stream* protocol. It is generic in `DATA_BITS` and in `META_BITS` as well as in FIFO depths for data and meta information.

Entity Declaration:

```

1  entity stream_Buffer is
2      generic (
3          FRAMES           : positive                                := 2;
4          DATA_BITS       : positive                                := 8;
5          DATA_FIFO_DEPTH : positive                                := 8;
6          META_BITS        : T_POSVEC                               := 8;
7          META_FIFO_DEPTH  : T_POSVEC                               := 16;
8      );
9      port (
10         Clock           : in  std_logic;
11         Reset           : in  std_logic;
12         -- IN Port
13         In_Valid        : in  std_logic;
14         In_Data          : in  std_logic_vector (DATA_BITS - 1 downto 0);
15         In_SOF           : in  std_logic;
16         In_EOF           : in  std_logic;
17         In_Ack           : out std_logic;
18         In_Meta_rst      : out std_logic;
19         In_Meta_nxt      : out std_logic_vector (META_BITS'length - 1 downto 0);
20         In_Meta_Data     : in  std_logic_vector (isum(META_BITS) - 1 downto 0);
21         -- OUT Port
22         Out_Valid        : out std_logic;
23         Out_Data         : out std_logic_vector (DATA_BITS - 1 downto 0);
24         Out_SOF          : out std_logic;

```

(continues on next page)

(continued from previous page)

```

25     Out_EOF           : out std_logic;
26     Out_Ack           : in  std_logic;
27     Out_Meta_rst      : in  std_logic;
28     Out_Meta_nxt      : in  std_logic_vector(META_BITS'length - 1 downto 0);
29     Out_Meta_Data     : out std_logic_vector(isum(META_BITS) - 1 downto 0)
30 );
31 end entity;

```

Source file: [bus/stream/stream_Buffer.vhdl](#)

PoC.bus.stream.DeMux

Todo: No documentation available.

Entity Declaration:

```

1  entity stream_DeMux is
2      generic (
3          PORTS           : positive           := 2;
4          DATA_BITS      : positive           := 8;
5          META_BITS       : natural            := 8;
6          META_REV_BITS   : natural            := 2
7      );
8      port (
9          Clock            : in  std_logic;
10         Reset             : in  std_logic;
11         -- Control interface
12         DeMuxControl      : in  std_logic_vector(PORTS - 1 downto 0);
13         -- IN Port
14         In_Valid          : in  std_logic;
15         In_Data            : in  std_logic_vector(DATA_BITS - 1 downto 0);
16         In_Meta           : in  std_logic_vector(META_BITS - 1 downto 0);
17         In_Meta_rev       : out std_logic_vector(META_REV_BITS - 1 downto 0);
18         In_SOF            : in  std_logic;
19         In_EOF            : in  std_logic;
20         In_Ack            : out std_logic;
21         -- OUT Ports
22         Out_Valid         : out std_logic_vector(PORTS - 1 downto 0);
23         Out_Data          : out T_SLM(PORTS - 1 downto 0, DATA_BITS - 1 downto 0);
24         Out_Meta          : out T_SLM(PORTS - 1 downto 0, META_BITS - 1 downto 0);
25         Out_Meta_rev      : in  T_SLM(PORTS - 1 downto 0, META_REV_BITS - 1 downto 0);
26         Out_SOF           : out std_logic_vector(PORTS - 1 downto 0);
27         Out_EOF           : out std_logic_vector(PORTS - 1 downto 0);
28         Out_Ack           : in  std_logic_vector(PORTS - 1 downto 0)
29     );
30 end entity;

```

Source file: [bus/stream/stream_DeMux.vhdl](#)

PoC.bus.stream.Mux

Todo: No documentation available.

Entity Declaration:

```
1 entity stream_Mux is
2   generic (
3     PORTS          : positive          := 2;
4     DATA_BITS     : positive          := 8;
5     META_BITS      : natural           := 8;
6     META_REV_BITS  : natural           := 2--;
7     -- WEIGHTS      : T_INTVEC          := (1, 1)
8   );
9   port (
10    Clock           : in  std_logic;
11    Reset           : in  std_logic;
12    -- IN Ports
13    In_Valid        : in  std_logic_vector(PORTS - 1 downto 0);
14    In_Data          : in  T_SLM(PORTS - 1 downto 0, DATA_BITS - 1 downto 0);
15    In_Meta         : in  T_SLM(PORTS - 1 downto 0, META_BITS - 1 downto 0);
16    In_Meta_rev     : out T_SLM(PORTS - 1 downto 0, META_REV_BITS - 1 downto 0);
17    In_SOF           : in  std_logic_vector(PORTS - 1 downto 0);
18    In_EOF           : in  std_logic_vector(PORTS - 1 downto 0);
19    In_Ack           : out std_logic_vector(PORTS - 1 downto 0);
20    -- OUT Port
21    Out_Valid        : out std_logic;
22    Out_Data          : out std_logic_vector(DATA_BITS - 1 downto 0);
23    Out_Meta         : out std_logic_vector(META_BITS - 1 downto 0);
24    Out_Meta_rev     : in  std_logic_vector(META_REV_BITS - 1 downto 0);
25    Out_SOF           : out std_logic;
26    Out_EOF           : out std_logic;
27    Out_Ack           : in  std_logic
28  );
29 end entity;
```

Source file: [bus/stream/stream_Mux.vhdl](#)

PoC.bus.stream.Mirror

Todo: No documentation available.

Entity Declaration:

```
1 entity stream_Mirror is
2   generic (
3     PORTS          : positive          := 2;
4     DATA_BITS     : positive          := 8;
5     META_BITS      : T_POSVEC          := (0 => 8);
6     META_LENGTH    : T_POSVEC          := (0 => 16)
7   );
8   port (
9     Clock           : in  std_logic;
10    Reset           : in  std_logic;
11    -- IN Port
12    In_Valid        : in  std_logic;
13    In_Data          : in  std_logic_vector(DATA_BITS - 1 downto 0);
14    In_SOF           : in  std_logic;
15    In_EOF           : in  std_logic;
16    In_Ack           : out std_logic;
17    In_Meta_rst     : out std_logic;
```

(continues on next page)

(continued from previous page)

```

18   In_Meta_nxt      : out std_logic_vector(META_BITS'length - 1 downto 0);
19   In_Meta_Data    : in  std_logic_vector(isum(META_BITS) - 1 downto 0);
20   -- OUT Port
21   Out_Valid       : out std_logic_vector(PORTS - 1 downto 0);
22   Out_Data        : out T_SLM(PORTS - 1 downto 0, DATA_BITS - 1 downto 0);
23   Out_SOF         : out std_logic_vector(PORTS - 1 downto 0);
24   Out_EOF         : out std_logic_vector(PORTS - 1 downto 0);
25   Out_Ack         : in  std_logic_vector(PORTS - 1 downto 0);
26   Out_Meta_rst    : in  std_logic_vector(PORTS - 1 downto 0);
27   Out_Meta_nxt    : in  T_SLM(PORTS - 1 downto 0, META_BITS'length - 1 downto
↪0);
28   Out_Meta_Data   : out T_SLM(PORTS - 1 downto 0, isum(META_BITS) - 1 downto 0)
29   );
30 end entity;
```

Source file: `bus/stream/stream_Mirror.vhdl`

PoC.bus.stream.Sink

Todo: No documentation available.

Entity Declaration:

Source file: `bus/stream/stream_Sink.vhdl`

PoC.bus.stream.Source

Todo: No documentation available.

Entity Declaration:

```

1  entity stream_Source is
2    generic (
3      TESTCASES      : T_SIM_STREAM_FRAMEGROUP_VECTOR_8
4    );
5    port (
6      Clock          : in  std_logic;
7      Reset          : in  std_logic;
8      -- Control interface
9      Enable         : in  std_logic;
10     -- OUT Port
11     Out_Valid       : out std_logic;
12     Out_Data        : out T_SLV_8;
13     Out_SOF         : out std_logic;
14     Out_EOF         : out std_logic;
15     Out_Ack         : in  std_logic
16   );
17 end entity;
```

Source file: `bus/stream/stream_Source.vhdl`

PoC.bus.stream.FrameGenerator

Todo: No documentation available.

Entity Declaration:

```
1  entity stream_FrameGenerator is
2      generic (
3          DATA_BITS          : positive          := 8;
4          WORD_BITS           : positive          := 16;
5          APPEND               : T_FRAMEGEN_APPEND := FRAMEGEN_APP_NONE;
6          FRAMEGROUPS         : T_FRAMEGEN_FRAMEGROUP_VECTOR_8 := (0 => C_FRAMEGEN_
7      ↪ FRAMEGROUP_EMPTY)
8      );
9      port (
10         Clock                : in  std_logic;
11         Reset                 : in  std_logic;
12         -- CSE interface
13         Command               : in  T_FRAMEGEN_COMMAND;
14         Status                 : out T_FRAMEGEN_STATUS;
15         -- Control interface
16         Pause                 : in  T_SLV_16;
17         FrameGroupIndex       : in  T_SLV_8;
18         FrameIndex            : in  T_SLV_8;
19         Sequences              : in  T_SLV_16;
20         FrameLength           : in  T_SLV_16;
21         -- OUT Port
22         Out_Valid             : out std_logic;
23         Out_Data              : out std_logic_vector (DATA_BITS - 1 downto 0);
24         Out_SOF               : out std_logic;
25         Out_EOF               : out std_logic;
26         Out_Ack               : in  std_logic
27     );
28 end entity;
```

Source file: [bus/stream/stream_FrameGenerator.vhdl](#)

7.5.2 PoC.bus.wb

WishBone modules ...

Entities:

PoC.bus.wb Package

Source file: [wb.pkg.vhdl](#)

PoC.bus.wb.ocram

This slave supports Wishbone Registered Feedback bus cycles (aka. burst transfers / advanced synchronous cycle termination). The mode “Incrementing burst cycle” (CTI = 010) with “Linear burst” (BTE = 00) is supported.

If your master does support Wishbone Classis bus cycles only, then connect `wb_cti_i` = “000” and `wb_bte_i` = “00”.

Connect the ocram of your choice to the `ram_*` port signals. (Every RAM with single cycle read latency is supported.)

Configuration:

PIPE_STAGES = 1 The RAM output is directly connected to the bus. Thus, the read access latency (one cycle) is short. But, the RAM's read timing delay must be respected.

PIPE_STAGES = 2 The RAM output is registered again. Thus, the read access latency is two cycles.

Entity Declaration:

Source file: `bus/wb/wb_ocram.vhdl`

PoC.bus.wb.fifo_adapter

Small FIFOs are included in this module, if larger or asynchronous transmit / receive FIFOs are required, then they must be connected externally.

old comments: UART BAUD rate generator `bclk_r` = bit clock is rising `bclk_x8_r` = bit clock times 8 is rising

Entity Declaration:

Source file: `bus/wb/wb_fifo_adapter.vhdl`

PoC.bus.wb.uart_wrapper

Wrapper module for *PoC.io.uart.rx* and *PoC.io.uart.tx* to support the Wishbone interface. Synchronized reset is used.

Entity Declaration:

Source file: `bus/wb/wb_uart_wrapper.vhdl`

7.5.3 PoC.bus.Arbitrator

This module implements a generic arbitrator. It currently supports the following arbitration strategies:

- Round Robin (RR)

Entity Declaration:

```

1  entity bus_Arbitrator is
2      generic (
3          STRATEGY          : string          := "RR";          -- RR, LOT
4          PORTS              : positive       := 1;
5          WEIGHTS            : T_INTVEC       := (0 => 1);
6          OUTPUT_REG        : boolean        := TRUE
7      );
8      port (
9          Clock              : in  std_logic;
10         Reset              : in  std_logic;
11
12         Arbitrate           : in  std_logic;
13         Request_Vector      : in  std_logic_vector (PORTS - 1 downto 0);
14

```

(continues on next page)

(continued from previous page)

```

15   Arbitrated           : out std_logic;
16   Grant_Vector         : out std_logic_vector (PORTS - 1 downto 0);
17   Grant_Index          : out std_logic_vector (log2ceilnz(PORTS) - 1 downto
    ↪ 0);
18   );
19 end entity;
```

Source file: [bus/bus_Arbiter.vhdl](#)

7.6 PoC.cache

The namespace *PoC.cache* offers different cache implementations.

Entities

- *PoC.cache.cpu*: Cache with cache controller to be used within a CPU.
- *PoC.cache.mem*: Cache with *PoC.Mem Interface* interface on the “CPU” side.
- *PoC.cache.par*: Cache with parallel tag-unit and data memory (using inferred memory).
- *PoC.cache.par2*: Cache with parallel tag-unit and data memory (using *PoC.mem.ocram.sp*).
- *PoC.cache.tagunit_par*: Tag-Unit with parallel tag comparison. Configurable as:
 - Full-associative cache,
 - Direct-mapped cache, or
 - Set-associative cache.
- *PoC.cache.tagunit_seq*: Tag-Unit with sequential tag comparison. Configurable as:
 - Full-associative cache,
 - Direct-mapped cache, or
 - Set-associative cache.

7.6.1 PoC.cache.cpu

This unit provides a cache (*PoC.cache.par2*) together with a cache controller which reads / writes cache lines from / to memory. The memory is accessed using a *PoC.Mem Interface* interfaces, the related ports and parameters are prefixed with `mem_`.

The CPU side (prefix `cpu_`) has a modified *PoC.Mem* interface, so that this unit can be easily integrated into processor pipelines. For example, let’s have a pipeline where a load/store instruction is executed in 3 stages (after fetching, decoding, ...):

1. Execute (EX) for address calculation,
2. Load/Store 1 (LS1) for the cache access,
3. Load/Store 2 (LS2) where the cache returns the read data.

The read data is always returned one cycle after the cache access completes, so there is conceptually a pipeline register within this unit. The stage LS2 can be merged with a write-back stage if the clock period allows so.

The stage LS1 and thus EX and LS2 must stall, until the cache access is completed, i.e., the EX/LS1 pipeline register must hold the cache request until it is acknowledged by the cache. This is signaled by `cpu_got` as described in Section Operation below. The pipeline moves forward (is enabled) when:

```
pipeline_enable <= (not cpu_req) or cpu_got;
```

If the pipeline can stall due to other reasons, care must be taken to not unintentionally executing the cache access twice or missing the read data.

Of course, the EX/LS1 pipeline register can be omitted and the CPU side directly fed by the address calculator. But be aware of the high setup time of this unit and high propagate time for `cpu_got`.

This unit supports only one outstanding CPU request. More outstanding requests are provided by [PoC.cache.mem](#).

Configuration

Parameter	Description
REPLACE- MENT_POLICY	Replacement policy of embedded cache. For supported values see <code>PoC.cache_replacement_policy</code> .
CACHE_LINES	Number of cache lines.
ASSOCIATIV- ITY	Associativity of embedded cache.
CPU_ADDR_BITS	Number of address bits on the CPU side. Each address identifies one memory word as seen from the CPU. Calculated from other parameters as described below.
CPU_DATA_BITS	Width of the data bus (in bits) on the CPU side. CPU_DATA_BITS must be divisible by 8.
MEM_ADDR_BITS	Number of address bits on the memory side. Each address identifies one word in the memory.
MEM_DATA_BITS	Width of a memory word and of a cache line in bits. MEM_DATA_BITS must be divisible by CPU_DATA_BITS.

If the CPU data-bus width is smaller than the memory data-bus width, then the CPU needs additional address bits to identify one CPU data word inside a memory word. Thus, the CPU address-bus width is calculated from:

$$\text{CPU_ADDR_BITS} = \log_2 \text{ceil}(\text{MEM_DATA_BITS} / \text{CPU_DATA_BITS}) + \text{MEM_ADDR_BITS}$$

The write policy is: write-through, no-write-allocate.

Operation

Alignment of Cache / Memory Accesses

Memory accesses are always aligned to a word boundary. Each memory word (and each cache line) consists of MEM_DATA_BITS bits. For example if MEM_DATA_BITS=128:

- memory address 0 selects the bits 0..127 in memory,
- memory address 1 selects the bits 128..256 in memory, and so on.

Cache accesses are always aligned to a CPU word boundary. Each CPU word consists of CPU_DATA_BITS bits. For example if CPU_DATA_BITS=32:

- CPU address 0 selects the bits 0.. 31 in memory word 0,
- CPU address 1 selects the bits 32.. 63 in memory word 0,
- CPU address 2 selects the bits 64.. 95 in memory word 0,
- CPU address 3 selects the bits 96..127 in memory word 0,
- CPU address 4 selects the bits 0.. 31 in memory word 1,
- CPU address 5 selects the bits 32.. 63 in memory word 1, and so on.

Shared and Memory Side Interface

A synchronous reset must be applied even on a FPGA.

The memory side interface is documented in detail [here](#).

CPU Side Interface

The CPU (pipeline stage LS1, see above) issues a request by setting `cpu_req`, `cpu_write`, `cpu_addr`, `cpu_wdata` and `cpu_wmask` as in the *PoC.Mem Interface* interface. The cache acknowledges the request by setting `cpu_got` to '1'. If the request is not acknowledged (`cpu_got = '0'`) in the current clock cycle, then the request must be repeated in the following clock cycle(s) until it is acknowledged, i.e., the pipeline must stall.

A cache access is completed when it is acknowledged. A new request can be issued in the following clock cycle.

Of course, `cpu_got` may be asserted in the same clock cycle where the request was issued if a read hit occurs. This allows a throughput of one (read) request per clock cycle, but the drawback is, that `cpu_got` has a high propagation delay. Thus, this output should only control a simple pipeline enable logic.

When `cpu_got` is asserted for a read access, then the read data will be available in the following clock cycle.

Due to the write-through policy, a write will always take several clock cycles and acknowledged when the data has been issued to the memory.

Warning: If the design is synthesized with Xilinx ISE / XST, then the synthesis option “Keep Hierarchy” must be set to SOFT or TRUE.

Entity Declaration:

```

1  entity cache_cpu is
2      generic (
3          REPLACEMENT_POLICY : string := "LRU";
4          CACHE_LINES         : positive;
5          ASSOCIATIVITY       : positive;
6          CPU_DATA_BITS       : positive;
7          MEM_ADDR_BITS       : positive;
8          MEM_DATA_BITS       : positive
9      );
10     port (
11         clk : in std_logic; -- clock
12         rst : in std_logic; -- reset
13
14         -- "CPU" side
15         cpu_req   : in std_logic;
16         cpu_write : in std_logic;
17         cpu_addr  : in unsigned(log2ceil(MEM_DATA_BITS/CPU_DATA_BITS)+MEM_ADDR_BITS-1_
18         ↪downto 0);
19         cpu_wdata : in std_logic_vector(CPU_DATA_BITS-1 downto 0);
20         cpu_wmask : in std_logic_vector(CPU_DATA_BITS/8-1 downto 0);
21         cpu_got   : out std_logic;
22         cpu_rdata : out std_logic_vector(CPU_DATA_BITS-1 downto 0);
23
24         -- Memory side
25         mem_req   : out std_logic;
26         mem_write : out std_logic;
27         mem_addr  : out unsigned(MEM_ADDR_BITS-1 downto 0);
28         mem_wdata : out std_logic_vector(MEM_DATA_BITS-1 downto 0);

```

(continues on next page)

(continued from previous page)

```

28     mem_wmask : out std_logic_vector(MEM_DATA_BITS/8-1 downto 0);
29     mem_rdy   : in  std_logic;
30     mem_rstb  : in  std_logic;
31     mem_rdata : in  std_logic_vector(MEM_DATA_BITS-1 downto 0)
32 );
33 end entity;
```

See also:*PoC.cache.mem*Source file: `cache/cache_cpu.vhdl`

7.6.2 PoC.cache.mem

This unit provides a cache (*PoC.cache.par2*) together with a cache controller which reads / writes cache lines from / to memory. It has two *PoC.Mem Interface* interfaces:

- one for the “CPU” side (ports with prefix `cpu_`), and
- one for the memory side (ports with prefix `mem_`).

Thus, this unit can be placed into an already available memory path between the CPU and the memory (controller). If you want to plugin a cache into a CPU pipeline, see *PoC.cache.cpu*.

Configuration

Parameter	Description
REPLACE-MENT_POLICY	Replacement policy of embedded cache. For supported values see <code>PoC.cache_replacement_policy</code> .
CACHE_LINES	Number of cache lines.
ASSOCIATIV-ITY	Associativity of embedded cache.
CPU_ADDR_BITS	Number of address bits on the CPU side. Each address identifies one memory word as seen from the CPU. Calculated from other parameters as described below.
CPU_DATA_BITS	Width of the data bus (in bits) on the CPU side. <code>CPU_DATA_BITS</code> must be divisible by 8.
MEM_ADDR_BITS	Number of address bits on the memory side. Each address identifies one word in the memory.
MEM_DATA_BITS	Width of a memory word and of a cache line in bits. <code>MEM_DATA_BITS</code> must be divisible by <code>CPU_DATA_BITS</code> .
OUTSTAND-ING_REQ	Number of outstanding requests, see notes below.

If the CPU data-bus width is smaller than the memory data-bus width, then the CPU needs additional address bits to identify one CPU data word inside a memory word. Thus, the CPU address-bus width is calculated from:

$$\text{CPU_ADDR_BITS} = \log_2 \text{ceil}(\text{MEM_DATA_BITS} / \text{CPU_DATA_BITS}) + \text{MEM_ADDR_BITS}$$

The write policy is: write-through, no-write-allocate.

The maximum throughput is one request per clock cycle, except for `OUTSTANDING_REQ = 1`.

If `OUTSTANDING_REQ` is:

- 1: then 1 request is buffered by a single register. To give a short critical path (clock-to-output delay) for `cpu_rdy`, the throughput is degraded to one request per 2 clock cycles at maximum.
- 2: then 2 requests are buffered by *PoC.fifo.glue*. This setting has the lowest area requirements without degrading the performance.

- >2: then the requests are buffered by *PoC.fifo.cc_got*. The number of outstanding requests is rounded up to the next suitable value. This setting is useful in applications with out-of-order execution (of other operations). The CPU requests to the cache are always processed in-order.

Operation

Memory accesses are always aligned to a word boundary. Each memory word (and each cache line) consists of MEM_DATA_BITS bits. For example if MEM_DATA_BITS=128:

- memory address 0 selects the bits 0..127 in memory,
- memory address 1 selects the bits 128..256 in memory, and so on.

Cache accesses are always aligned to a CPU word boundary. Each CPU word consists of CPU_DATA_BITS bits. For example if CPU_DATA_BITS=32:

- CPU address 0 selects the bits 0.. 31 in memory word 0,
- CPU address 1 selects the bits 32.. 63 in memory word 0,
- CPU address 2 selects the bits 64.. 95 in memory word 0,
- CPU address 3 selects the bits 96..127 in memory word 0,
- CPU address 4 selects the bits 0.. 31 in memory word 1,
- CPU address 5 selects the bits 32.. 63 in memory word 1, and so on.

A synchronous reset must be applied even on a FPGA.

The interface is documented in detail [here](#).

Warning: If the design is synthesized with Xilinx ISE / XST, then the synthesis option “Keep Hierarchy” must be set to SOFT or TRUE.

Entity Declaration:

```

1  entity cache_mem is
2      generic (
3          REPLACEMENT_POLICY : string := "LRU";
4          CACHE_LINES         : positive;
5          ASSOCIATIVITY        : positive;
6          CPU_DATA_BITS        : positive;
7          MEM_ADDR_BITS        : positive;
8          MEM_DATA_BITS        : positive;
9          OUTSTANDING_REQ      : positive := 2
10     );
11     port (
12         clk : in std_logic; -- clock
13         rst : in std_logic; -- reset
14
15         -- "CPU" side
16         cpu_req  : in std_logic;
17         cpu_write : in std_logic;
18         cpu_addr  : in unsigned(log2ceil(MEM_DATA_BITS/CPU_DATA_BITS)+MEM_ADDR_BITS-1,
19         ↪downto 0);
20         cpu_wdata : in std_logic_vector(CPU_DATA_BITS-1 downto 0);
21         cpu_wmask : in std_logic_vector(CPU_DATA_BITS/8-1 downto 0) := (others => '0
22         ↪');
23         cpu_rdy   : out std_logic;
24         cpu_rstb  : out std_logic;
25         cpu_rdata : out std_logic_vector(CPU_DATA_BITS-1 downto 0);

```

(continues on next page)

(continued from previous page)

```

24
25  -- Memory side
26  mem_req   : out std_logic;
27  mem_write : out std_logic;
28  mem_addr  : out unsigned(MEM_ADDR_BITS-1 downto 0);
29  mem_wdata : out std_logic_vector(MEM_DATA_BITS-1 downto 0);
30  mem_wmask : out std_logic_vector(MEM_DATA_BITS/8-1 downto 0);
31  mem_rdy   : in  std_logic;
32  mem_rstb  : in  std_logic;
33  mem_rdata : in  std_logic_vector(MEM_DATA_BITS-1 downto 0)
34  );
35 end entity;

```

See also:[PoC.cache.cpu](#)Source file: [cache/cache_mem.vhdl](#)

7.6.3 PoC.cache.par

Implements a cache with parallel tag-unit and data memory.

Note: This component infers a single-port memory with read-first behavior, that is, upon writes the old-data is returned on the read output. Such memory (e.g. LUT-RAM) is not available on all devices. Thus, synthesis may infer a lot of flip-flops plus multiplexers instead, which is very inefficient. It is recommended to use [PoC.cache.par2](#) instead which has a slightly different interface.

All inputs are synchronous to the rising-edge of the clock *clock*.

Command truth table:

Request	ReadWrite	Invalidate	Replace	Command
0	0	0	0	None
1	0	0	0	Read cache line
1	1	0	0	Update cache line
1	0	1	0	Read cache line and discard it
1	1	1	0	Write cache line and discard it
0		0	1	Replace cache line.

All commands use *Address* to lookup (request) or replace a cache line. *Address* and *OldAddress* do not include the word/byte select part. Each command is completed within one clock cycle, but outputs are delayed as described below.

Upon requests, the outputs *CacheMiss* and *CacheHit* indicate (high-active) whether the *Address* is stored within the cache, or not. Both outputs have a latency of one clock cycle.

Upon writing a cache line, the new content is given by *CacheLineIn*. Upon reading a cache line, the current content is outputted on *CacheLineOut* with a latency of one clock cycle.

Upon replacing a cache line, the new content is given by *CacheLineIn*. The old content is outputted on *CacheLineOut* and the old tag on *OldAddress*, both with a latency of one clock cycle.

Warning: If the design is synthesized with Xilinx ISE / XST, then the synthesis option “Keep Hierarchy” must be set to SOFT or TRUE.

Entity Declaration:

```

1  entity cache_par is
2      generic (
3          REPLACEMENT_POLICY : string := "LRU";
4          CACHE_LINES        : positive := 32; --1024;
5          ASSOCIATIVITY       : positive := 32; --4;
6          ADDRESS_BITS        : positive := 8;  --32-6;
7          DATA_BITS          : positive := 8   --64*8
8      );
9      port (
10         Clock : in std_logic;
11         Reset  : in std_logic;
12
13         Request      : in std_logic;
14         ReadWrite    : in std_logic;
15         Invalidate   : in std_logic;
16         Replace      : in std_logic;
17         Address      : in std_logic_vector(ADDRESS_BITS - 1 downto 0);
18
19         CacheLineIn  : in std_logic_vector(DATA_BITS - 1 downto 0);
20         CacheLineOut : out std_logic_vector(DATA_BITS - 1 downto 0);
21         CacheHit     : out std_logic := '0';
22         CacheMiss    : out std_logic := '0';
23         OldAddress   : out std_logic_vector(ADDRESS_BITS - 1 downto 0)
24     );
25 end entity;

```

Source file: [cache/cache_par.vhdl](#)

7.6.4 PoC.cache.par2

Cache with parallel tag-unit and data memory. For the data memory, *PoC.mem.ocram.sp* is used.

Configuration

Parameter	Description
REPLACE-MENT_POLICY	Replacement policy. For supported policies see PoC.cache_replacement_policy.
CACHE_LINES	Number of cache lines.
ASSOCIATIVITY	Associativity of the cache.
ADDR_BITS	Number of address bits. Each address identifies exactly one cache line in memory.
DATA_BITS	Size of a cache line in bits. DATA_BITS must be divisible by 8.

Command truth table

Request	ReadWrite	Invalidate	Replace	Command
0	0	0	0	None
1	0	0	0	Read cache line
1	1	0	0	Update cache line
1	0	1	0	Read cache line and discard it
1	1	1	0	Write cache line and discard it
0	0	0	1	Read cache line before replace.
0	1	0	1	Replace cache line.

Operation

All inputs are synchronous to the rising-edge of the clock *clock*.

All commands use *Address* to lookup (request) or replace a cache line. *Address* and *OldAddress* do not include the word/byte select part. Each command is completed within one clock cycle, but outputs are delayed as described below.

Upon requests, the outputs *CacheMiss* and *CacheHit* indicate (high-active) whether the *Address* is stored within the cache, or not. Both outputs have a latency of one clock cycle (pipelined) if *HIT_MISS_REG* is true, otherwise the result is outputted immediately (combinational).

Upon writing a cache line, the new content is given by *CacheLineIn*. Only the bytes which are not masked, i.e. the corresponding bit in *WriteMask* is '0', are actually written.

Upon reading a cache line, the current content is outputted on *CacheLineOut* with a latency of one clock cycle.

Replacing a cache line requires two steps, both with *Replace* = '1':

1. Read old contents of cache line by setting *ReadWrite* to '0'. The old content is outputted on *CacheLineOut* and the old tag on *OldAddress*, both with a latency of one clock cycle.
2. Write new cache line by setting *ReadWrite* to '1'. The new content is given by *CacheLineIn*. All bytes shall be written, i.e. *WriteMask* = 0. The new cache line content will be outputted again on *CacheLineOut* in the next clock cycle (latency = 1).

Warning: If the design is synthesized with Xilinx ISE / XST, then the synthesis option "Keep Hierarchy" must be set to SOFT or TRUE.

Entity Declaration:

```

1  entity cache_par2 is
2      generic (
3          REPLACEMENT_POLICY : string    := "LRU";
4          CACHE_LINES         : positive := 32;
5          ASSOCIATIVITY       : positive := 32;
6          ADDR_BITS           : positive := 8;
7          DATA_BITS          : positive := 8;
8          HIT_MISS_REG        : boolean   := true  -- must be true for Cocotb.
9      );
10     port (
11         Clock : in std_logic;
12         Reset  : in std_logic;
13
14         Request      : in std_logic;
15         ReadWrite    : in std_logic;
16         WriteMask     : in std_logic_vector(DATA_BITS/8 - 1 downto 0) := (others => '0');
17         Invalidate    : in std_logic;
18         Replace       : in std_logic;
19         Address       : in std_logic_vector(ADDR_BITS-1 downto 0);
20
21         CacheLineIn  : in std_logic_vector(DATA_BITS - 1 downto 0);
22         CacheLineOut : out std_logic_vector(DATA_BITS - 1 downto 0);
23         CacheHit      : out std_logic := '0';
24         CacheMiss     : out std_logic := '0';
25         OldAddress    : out std_logic_vector(ADDR_BITS-1 downto 0)
26     );
27 end entity;
```

Source file: `cache/cache_par2.vhdl`

7.6.5 PoC.cache.replacement_policy

Supported policies:

Abbr.	Policies	supported
RR	round robin	not yet
RAND	random	not yet
CLOCK	clock algorithm	not yet
LRU	least recently used	YES
LFU	least frequently used	not yet

Command thruth table:

TagAccess	ReadWrite	Invalidate	Replace	Command
0			0	None
1	0	0	0	TagHit and reading a cache line
1	1	0	0	TagHit and writing a cache line
1	0	1	0	TagHit and invalidate a cache line (while reading)
1	1	1	0	TagHit and invalidate a cache line (while writing)
0		0	1	Replace cache line

In a set-associative cache, each cache-set has its own instance of this component.

The input `HitWay` specifies the accessed way in a fully-associative or set-associative cache.

The output `ReplaceWay` identifies the way which will be replaced as next by a replace command. In a set-associative cache, this is the way in a specific cache set (see above).

Entity Declaration:

```
1  entity cache_replacement_policy is
2    generic (
3      REPLACEMENT_POLICY : string    := "LRU";
4      CACHE_WAYS          : positive := 32
5    );
6    port (
7      Clock : in std_logic;
8      Reset : in std_logic;
9
10     -- replacement interface
11     Replace      : in std_logic;
12     ReplaceWay   : out std_logic_vector(log2ceilnz(CACHE_WAYS) - 1 downto 0);
13
14     -- cacheline usage update interface
15     TagAccess    : in std_logic;
16     ReadWrite    : in std_logic;
17     Invalidate    : in std_logic;
18     HitWay       : in std_logic_vector(log2ceilnz(CACHE_WAYS) - 1 downto 0)
19   );
20 end entity;
```

Source file: `cache/cache_replacement_policy.vhdl`

7.6.6 PoC.cache.tagunit_par

Tag-unit with fully-parallel compare of tag.

Configuration

Parameter	Description
REPLACE- MENT_POLICY	Replacement policy. For supported policies see PoC.cache_replacement_policy.
CACHE_LINES	Number of cache lines.
ASSOCIATIVITY	Associativity of the cache.
ADDRESS_BITS	Number of address bits. Each address identifies exactly one cache line in memory.

Command truth table

Request	ReadWrite	Invalidate	Replace	Command
0	0	0	0	None
1	0	0	0	Read cache line
1	1	0	0	Update cache line
1	0	1	0	Read cache line and discard it
1	1	1	0	Write cache line and discard it
0		0	1	Replace cache line.

Operation

All inputs are synchronous to the rising-edge of the clock *clock*.

All commands use *Address* to lookup (request) or replace a cache line. Each command is completed within one clock cycle.

Upon requests, the outputs *CacheMiss* and *CacheHit* indicate (high-active) immediately (combinational) whether the *Address* is stored within the cache, or not. But, the cache-line usage is updated at the rising-edge of the clock. If hit, *LineIndex* specifies the cache line where to find the content.

The output *ReplaceLineIndex* indicates which cache line will be replaced as next by a replace command. The output *OldAddress* specifies the old tag stored at this index. The replace command will store the *Address* and update the cache-line usage at the rising-edge of the clock.

For a direct-mapped cache, the number of *CACHE_LINES* must be a power of 2. For a set-associative cache, the expression $CACHE_LINES / ASSOCIATIVITY$ must be a power of 2.

Note: The port *NewAddress* has been removed. Use *Address* instead as described above.

If *Address* is fed from a register and an Altera FPGA is used, then Quartus Map converts the tag memory from a memory with asynchronous read to a memory with synchronous read by adding a pass-through logic. Quartus Map reports warning 276020 which is intended.

Warning: If the design is synthesized with Xilinx ISE / XST, then the synthesis option “Keep Hierarchy” must be set to SOFT or TRUE.

Entity Declaration:

```

1 entity cache_tagunit_par is
2   generic (
3     REPLACEMENT_POLICY : string := "LRU";

```

(continues on next page)

(continued from previous page)

```

4  CACHE_LINES      : positive := 32;
5  ASSOCIATIVITY    : positive := 32;
6  ADDRESS_BITS     : positive := 8;
7  );
8  port (
9      Clock : in std_logic;
10     Reset : in std_logic;
11
12     Replace      : in std_logic;
13     ReplaceLineIndex : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto 0);
14     OldAddress    : out std_logic_vector(ADDRESS_BITS - 1 downto 0);
15
16     Request      : in std_logic;
17     ReadWrite    : in std_logic;
18     Invalidate   : in std_logic;
19     Address      : in std_logic_vector(ADDRESS_BITS - 1 downto 0);
20     LineIndex    : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto 0);
21     TagHit       : out std_logic;
22     TagMiss      : out std_logic;
23 );
24 end entity;

```

Source file: `cache/cache_tagunit_par.vhdl`

7.6.7 PoC.cache.tagunit_seq

Todo: No documentation available.

Entity Declaration:

```

1  entity cache_tagunit_seq is
2      generic (
3          REPLACEMENT_POLICY : string      := "LRU";
4          CACHE_LINES        : positive    := 32;
5          ASSOCIATIVITY      : positive    := 32;
6          TAG_BITS           : positive    := 128;
7          CHUNK_BITS         : positive    := 8;
8          TAG_BYTE_ORDER     : T_BYTE_ORDER := LITTLE_ENDIAN;
9          USE_INITIAL_TAGS   : boolean     := false;
10         INITIAL_TAGS       : T_SLM       := (0 downto 0 => (127 downto 0 => '0'))
11     );
12     port (
13         Clock : in std_logic;
14         Reset : in std_logic;
15
16         Replace      : in std_logic;
17         Replaced     : out std_logic;
18         Replace_NewTag_rst : out std_logic;
19         Replace_NewTag_rev : out std_logic;
20         Replace_NewTag_nxt : out std_logic;
21         Replace_NewTag_Data : in std_logic_vector(CHUNK_BITS - 1 downto 0);
22         Replace_NewIndex  : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto _
↳ 0);
23
24         Request      : in std_logic;
25         Request_ReadWrite : in std_logic;

```

(continues on next page)

(continued from previous page)

```

26     Request_Invalidate : in  std_logic;
27     Request_Tag_rst    : out std_logic;
28     Request_Tag_rev    : out std_logic;
29     Request_Tag_nxt    : out std_logic;
30     Request_Tag_Data   : in  std_logic_vector(CHUNK_BITS - 1 downto 0);
31     Request_Index      : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto
↳ 0);
32     Request_TagHit     : out std_logic;
33     Request_TagMiss    : out std_logic
34 );
35 end entity;

```

Source file: `cache/cache_tagunit_seq.vhdl`

7.7 PoC.comm

These are communication entities. ...

7.7.1 PoC.comm Package

Source file: `comm.pkg.vhdl`

7.7.2 PoC.comm.crc

Computes the Cyclic Redundancy Check (CRC) for a data packet as remainder of the polynomial division of the message by the given generator polynomial (GEN).

The computation is unrolled so as to process an arbitrary number of message bits per step. The generated CRC is independent from the chosen processing width.

Entity Declaration:

```

1  entity comm_crc is
2      generic (
3          GEN      : bit_vector;           -- Generator Polynomial
4          BITS     : positive;             -- Number of Bits to be
↳ processed in parallel
5
6          STARTUP_RMD : std_logic_vector := "0";
7          OUTPUT_REGS : boolean          := true
8      );
9      port (
10         clk : in  std_logic;              -- Clock
11
12         set : in  std_logic;              -- Parallel Preload of
↳ Remainder
13         init : in std_logic_vector(abs(mssb_idx(GEN)-GEN'right)-1 downto 0); --
14         step : in std_logic;              -- Process Input Data (MSB
↳ first)
15         din : in  std_logic_vector(BITS-1 downto 0);      --
16
17         rmd : out std_logic_vector(abs(mssb_idx(GEN)-GEN'right)-1 downto 0); --
↳ Remainder
18         zero : out std_logic              -- Remainder is Zero

```

(continues on next page)

(continued from previous page)

```
19 );  
20 end entity comm_crc;
```

Source file: `comm/comm_crc.vhdl`

7.7.3 PoC.comm.scramble

The LFSR computation is unrolled to generate an arbitrary number of mask bits in parallel. The mask are output in little endian. The generated bit sequence is independent from the chosen output width.

Entity Declaration:

```
1  entity comm_scramble is  
2      generic (  
3          GEN    : bit_vector;      -- Generator Polynomial (little endian)  
4          BITS   : positive         -- Width of Mask Bits to be computed in parallel in_  
--each step  
5      );  
6      port (  
7          clk    : in  std_logic;    -- Clock  
8  
9          set    : in  std_logic;     -- Set LFSR to value provided on din  
10         din    : in  std_logic_vector(GEN'length-2 downto 0) := (others => '0');  
11  
12         step   : in  std_logic;     -- Compute a Mask Output  
13         mask   : out std_logic_vector(BITS-1 downto 0)  
14     );  
15 end entity comm_scramble;
```

Source file: `comm/comm_scramble.vhdl`

7.8 PoC.dstruct

The namespace *PoC.dstruct* offers different data structure implementations.

Package

The package *PoC.dstruct* holds all component declarations for this namespace.

Entities

- *PoC.dstruct.deque* implements a deque (two-sided FIFO).
- *PoC.dstruct.stack* implements a regular stack.

7.8.1 PoC.dstruct.deque

Implements a deque (double-ended queue). This data structure allows two acting entities to queue data elements for the consumption by the other while still being able to unqueue untaken ones in LIFO fashion.

Entity Declaration:

```

1 entity dstruct_deque is
2   generic (
3     D_BITS      : positive;           -- Data Width
4     MIN_DEPTH   : positive           -- Minimum Deque Depth
5   );
6   port (
7     -- Shared Ports
8     clk, rst : in std_logic;
9
10    -- Port A
11    dinA      : in  std_logic_vector(D_BITS-1 downto 0); -- DataA Input
12    putA      : in  std_logic;
13    gotA      : in  std_logic;
14    doutA     : out std_logic_vector(D_BITS-1 downto 0); -- DataA Output
15    validA    : out std_logic;
16    fullA     : out std_logic;
17
18    -- Port B
19    dinB      : in  std_logic_vector(D_BITS-1 downto 0); -- DataB Input
20    putB      : in  std_logic;
21    gotB      : in  std_logic;
22    doutB     : out std_logic_vector(D_BITS-1 downto 0);
23    validB    : out std_logic;
24    fullB     : out std_logic
25  );
26 end entity dstruct_deque;

```

Source file: dstruct/dstruct_deque.vhdl

7.8.2 PoC.dstruct.stack

Implements a stack, a LIFO storage abstraction.

Entity Declaration:

```

1 entity dstruct_stack is
2   generic (
3     D_BITS      : positive;           -- Data Width
4     MIN_DEPTH   : positive           -- Minimum Stack Depth
5   );
6   port (
7     -- INPUTS
8     clk, rst : in std_logic;
9
10    -- Write Ports
11    din      : in  std_logic_vector(D_BITS-1 downto 0); -- Data Input
12    put      : in  std_logic; -- 0 -> pop, 1 -> push
13    full     : out std_logic;
14
15    -- Read Ports
16    got      : in  std_logic;
17    dout     : out std_logic_vector(D_BITS-1 downto 0);
18    valid    : out std_logic
19  );
20 end entity dstruct_stack;

```

Source file: dstruct/dstruct_stack.vhdl

7.9 PoC.fifo

The namespace *PoC.fifo* offers different FIFO (first-in, first-out) implementations.

Package

The package *PoC.fifo* holds all component declarations for this namespace.

Entities

PoC offers FIFOs with a *got*-interface. This means, the current read-pointer value is available on the output. Asserting the *got*-input, acknowledge the processing of the current output signals and moves the read-pointer to the next value, if available.

All FIFOs implement a bidirectional flow control (*put/full* and *valid/got*). Each FIFO also offers a *EmptyState* (write-side) and *FullState* (read-side) to indicate the current fill-state.

The prefixes *cc_* (common clock), *dc_* (dependent clock) and *ic_* (independent clock) refer to the write- and read-side clock relationship.

- *PoC.fifo.cc_got* implements a regular FIFO (one common clock, got-interface)
- *PoC.fifo.cc_got_tempgot* implements a regular FIFO (one common clock, got-interface), extended by a transactional *tempgot*-interface (read-side).
- *PoC.fifo.cc_got_tempput* implements a regular FIFO (one common clock, got-interface), extended by a transactional *tempput*-interface (write-side).
- *IP:fifo_dc_got* implements a cross-clock FIFO (two related clocks, got-interface)
- *PoC.fifo.ic_got* implements a cross-clock FIFO (two independent clocks, got-interface)
- *PoC.fifo.glue* implements a two-stage FIFO (one common clock, got-interface)
- *PoC.fifo.shift* implements a regular FIFO (one common clock, got-interface, optimized for FPGAs with shifter primitives)

7.9.1 PoC.fifo Package

This package holds all component declarations for this namespace.

Source file: [fifo.pkg.vhdl](#)

7.9.2 PoC.fifo.cc_got

This module implements a regular FIFO with common clock (*cc*), pipelined interface. Common clock means read and write port use the same clock. The FIFO size can be configured in word width (*D_BITS*) and minimum word count *MIN_DEPTH*. The specified depth is rounded up to the next suitable value.

DATA_REG (=true) is a hint, that distributed memory or registers should be used as data storage. The actual memory type depends on the device architecture. See implementation for details.

STATE_*_BITS* defines the granularity of the fill state indicator **state_. If a fill state is not of interest, set **STATE_*_BITS* = 0. *fstate_rd* is associated with the read clock domain and outputs the guaranteed number of words available in the FIFO. *estate_wr* is associated with the write clock domain and outputs the number of words that is guaranteed to be accepted by the FIFO without a capacity overflow. Note that both these indicators cannot replace the *full* or *valid* outputs as they may be implemented as giving pessimistic bounds that are minimally off the true fill state.

fstate_rd and *estate_wr* are combinatorial outputs and include an address comparator (subtractor) in their path.

Examples:

- FSTATE_RD_BITS = 1:

fstate_rd	filled (at least)
0	0/2 full
1	1/2 full (half full)

- FSTATE_RD_BITS = 2:

fstate_rd	filled (at least)
0	0/4 full
1	1/4 full
2	2/4 full (half full)
3	3/4 full

Entity Declaration:

```

1 entity fifo_cc_got is
2   generic (
3     D_BITS          : positive;           -- Data Width
4     MIN_DEPTH       : positive;           -- Minimum FIFO Depth
5     DATA_REG       : boolean := false;   -- Store Data Content in Registers
6     STATE_REG       : boolean := false;   -- Registered Full/Empty Indicators
7     OUTPUT_REG      : boolean := false;   -- Registered FIFO Output
8     ESTATE_WR_BITS  : natural := 0;       -- Empty State Bits
9     FSTATE_RD_BITS  : natural := 0;       -- Full State Bits
10  );
11  port (
12    -- Global Reset and Clock
13    rst, clk : in std_logic;
14
15    -- Writing Interface
16    put       : in std_logic;               -- Write Request
17    din       : in std_logic_vector(D_BITS-1 downto 0); -- Input Data
18    full      : out std_logic;
19    estate_wr : out std_logic_vector(imax(0, ESTATE_WR_BITS-1) downto 0);
20
21    -- Reading Interface
22    got       : in std_logic;               -- Read Completed
23    dout      : out std_logic_vector(D_BITS-1 downto 0); -- Output Data
24    valid     : out std_logic;
25    fstate_rd : out std_logic_vector(imax(0, FSTATE_RD_BITS-1) downto 0)
26  );
27 end entity fifo_cc_got;

```

See also:

IP:[fifo_dc_got](#) For a FIFO with dependent clocks.

[PoC.fifo.ic_got](#) For a FIFO with independent clocks (cross-clock FIFO).

[PoC.fifo.glue](#) For a minimal FIFO / pipeline decoupling.

Source file: [fifo/fifo_cc_got.vhdl](#)

7.9.3 PoC.fifo.cc_got_tempgot

The specified depth (`MIN_DEPTH`) is rounded up to the next suitable value.

As uncommitted reads occupy FIFO space that is not yet available for writing, an instance of this FIFO can, indeed, report full and not vld at the same time. While a commit would eventually make space available for writing (not ful), a rollback would re-iterate data for reading (vld).

commit and rollback are inclusive and apply to all reads (got) since the previous commit or rollback up to and including a potentially simultaneous read.

The FIFO state upon a simultaneous assertion of commit and rollback is *undefined*!

*STATE_*_BITS defines the granularity of the fill state indicator *state_*. fstate_rd is associated with the read clock domain and outputs the guaranteed number of words available in the FIFO. estate_wr is associated with the write clock domain and outputs the number of words that is guaranteed to be accepted by the FIFO without a capacity overflow. Note that both these indicators cannot replace the full or valid outputs as they may be implemented as giving pessimistic bounds that are minimally off the true fill state.

If a fill state is not of interest, set *STATE_*_BITS = 0.

fstate_rd and estate_wr are combinatorial outputs and include an address comparator (subtractor) in their path.

Examples:

- FSTATE_RD_BITS = 1:
 - fstate_rd == 0 => 0/2 full
 - fstate_rd == 1 => 1/2 full (half full)
- FSTATE_RD_BITS = 2:
 - fstate_rd == 0 => 0/4 full
 - fstate_rd == 1 => 1/4 full
 - fstate_rd == 2 => 2/4 full
 - fstate_rd == 3 => 3/4 full

Entity Declaration:

```

1  entity fifo_cc_got_tempgot is
2      generic (
3          D_BITS          : positive;          -- Data Width
4          MIN_DEPTH       : positive;          -- Minimum FIFO Depth
5          DATA_REG       : boolean := false;  -- Store Data Content in Registers
6          STATE_REG       : boolean := false;  -- Registered Full/Empty Indicators
7          OUTPUT_REG      : boolean := false;  -- Registered FIFO Output
8          ESTATE_WR_BITS  : natural := 0;      -- Empty State Bits
9          FSTATE_RD_BITS  : natural := 0;      -- Full State Bits
10     );
11     port (
12         -- Global Reset and Clock
13         rst, clk : in  std_logic;
14
15         -- Writing Interface
16         put      : in  std_logic;              -- Write Request
17         din      : in  std_logic_vector(D_BITS-1 downto 0); -- Input Data
18         full     : out std_logic;
19         estate_wr : out std_logic_vector(imax(0, ESTATE_WR_BITS-1) downto 0);
20
21         -- Reading Interface
22         got      : in  std_logic;              -- Read Completed
23         dout     : out std_logic_vector(D_BITS-1 downto 0); -- Output Data
24         valid    : out std_logic;
25         fstate_rd : out std_logic_vector(imax(0, FSTATE_RD_BITS-1) downto 0);

```

(continues on next page)

(continued from previous page)

```

26
27     commit      : in std_logic;
28     rollback    : in std_logic
29 );
30 end entity fifo_cc_got_tempgot;

```

Source file: `fifo/fifo_cc_got_tempgot.vhdl`

7.9.4 PoC.fifo.cc_got_tempput

The specified depth (`MIN_DEPTH`) is rounded up to the next suitable value.

As uncommitted writes populate FIFO space that is not yet available for reading, an instance of this FIFO can, indeed, report `full` and `not vld` at the same time. While a `commit` would eventually make data available for reading (`vld`), a `rollback` would free the space for subsequent writing (`not ful`).

`commit` and `rollback` are inclusive and apply to all writes (`put`) since the previous ‘commit’ or ‘rollback’ up to and including a potentially simultaneous write.

The FIFO state upon a simultaneous assertion of `commit` and `rollback` is *undefined*.

`*STATE*_BITS` defines the granularity of the fill state indicator `*state_*`. `fstate_rd` is associated with the read clock domain and outputs the guaranteed number of words available in the FIFO. `estate_wr` is associated with the write clock domain and outputs the number of words that is guaranteed to be accepted by the FIFO without a capacity overflow. Note that both these indicators cannot replace the `full` or `valid` outputs as they may be implemented as giving pessimistic bounds that are minimally off the true fill state.

If a fill state is not of interest, set `*STATE*_BITS = 0`.

`fstate_rd` and `estate_wr` are combinatorial outputs and include an address comparator (subtractor) in their path.

Examples:

- `FSTATE_RD_BITS = 1`:
 - `fstate_rd == 0` => 0/2 full
 - `fstate_rd == 1` => 1/2 full (half full)
- `FSTATE_RD_BITS = 2`:
 - `fstate_rd == 0` => 0/4 full
 - `fstate_rd == 1` => 1/4 full
 - `fstate_rd == 2` => 2/4 full
 - `fstate_rd == 3` => 3/4 full

Entity Declaration:

```

1  entity fifo_cc_got_tempput is
2      generic (
3          D_BITS          : positive;           -- Data Width
4          MIN_DEPTH       : positive;           -- Minimum FIFO Depth
5          DATA_REG       : boolean := false;   -- Store Data Content in Registers
6          STATE_REG       : boolean := false;   -- Registered Full/Empty Indicators
7          OUTPUT_REG      : boolean := false;   -- Registered FIFO Output
8          ESTATE_WR_BITS  : natural := 0;       -- Empty State Bits
9          FSTATE_RD_BITS  : natural := 0;       -- Full State Bits
10 );
11 port (

```

(continues on next page)

(continued from previous page)

```

12  -- Global Reset and Clock
13  rst, clk : in  std_logic;
14
15  -- Writing Interface
16  put      : in  std_logic;           -- Write Request
17  din      : in  std_logic_vector(D_BITS-1 downto 0); -- Input Data
18  full     : out std_logic;
19  estate_wr : out std_logic_vector(imax(0, ESTATE_WR_BITS-1) downto 0);
20
21  commit   : in  std_logic;
22  rollback : in  std_logic;
23
24  -- Reading Interface
25  got      : in  std_logic;           -- Read Completed
26  dout     : out std_logic_vector(D_BITS-1 downto 0); -- Output Data
27  valid    : out std_logic;
28  fstate_rd : out std_logic_vector(imax(0, FSTATE_RD_BITS-1) downto 0)
29  );
30  end entity fifo_cc_got_tempput;

```

Source file: `fifo/fifo_cc_got_tempput.vhdl`

7.9.5 PoC.fifo.glue

Its primary use is the decoupling of enable domains in a processing pipeline. Data storage is limited to two words only so as to allow both the `ful` and the `vld` indicators to be driven by registers.

Entity Declaration:

```

1  entity fifo_glue is
2    generic (
3      D_BITS : positive           -- Data Width
4    );
5    port (
6      -- Control
7      clk : in  std_logic;        -- Clock
8      rst : in  std_logic;        -- Synchronous Reset
9
10     -- Input
11     put : in  std_logic;         -- Put Value
12     di  : in  std_logic_vector(D_BITS-1 downto 0); -- Data Input
13     ful : out std_logic;         -- Full
14
15     -- Output
16     vld : out std_logic;         -- Data Available
17     do  : out std_logic_vector(D_BITS-1 downto 0); -- Data Output
18     got : in  std_logic         -- Data Consumed
19   );
20  end entity fifo_glue;

```

Source file: `fifo/fifo_glue.vhdl`

7.9.6 PoC.fifo.ic_assembly

This module assembles a FIFO stream from data blocks that may arrive slightly out of order. The arriving data is ordered according to their address. The streamed output starts with the data word written to address zero (0) and may proceed all the way to just before the first yet missing data. The association of data with addresses is used on

the input side for the sole purpose of reconstructing the correct order of the data. It is assumed to wrap so as to allow an infinite input sequence. Addresses are not actively exposed to the purely stream-based FIFO output.

The implemented functionality enables the reconstruction of streams that are tunnelled across address-based transports that are allowed to reorder the transmission of data blocks. This applies to many DMA implementations.

Entity Declaration:

```

1  entity fifo_ic_assembly is
2      generic (
3          D_BITS : positive;           -- Data Width
4          A_BITS : positive;           -- Address Bits
5          G_BITS : positive;           -- Generation Guard Bits
6      );
7      port (
8          -- Write Interface
9          clk_wr : in std_logic;
10         rst_wr : in std_logic;
11
12         -- Only write addresses in the range [base, base+2**(A_BITS-G_BITS)) are
13         -- acceptable. This is equivalent to the test
14         -- tmp(A_BITS-1 downto A_BITS-G_BITS) = 0 where tmp = addr - base.
15         -- Writes performed outside the allowable range will assert the failure
16         -- indicator, which will stick until the next reset.
17         -- No write is to be performed before base turns zero (0) for the first
18         -- time.
19         base : out std_logic_vector(A_BITS-1 downto 0);
20         failed : out std_logic;
21
22         addr : in std_logic_vector(A_BITS-1 downto 0);
23         din : in std_logic_vector(D_BITS-1 downto 0);
24         put : in std_logic;
25
26         -- Read Interface
27         clk_rd : in std_logic;
28         rst_rd : in std_logic;
29
30         dout : out std_logic_vector(D_BITS-1 downto 0);
31         vld : out std_logic;
32         got : in std_logic
33     );
34 end entity fifo_ic_assembly;

```

Source file: `fifo/fifo_ic_assembly.vhdl`

7.9.7 PoC.fifo.ic_got

Independent clocks means that read and write clock are unrelated.

This implementation uses dedicated block RAM for storing data.

First-word-fall-through (FWFT) mode is implemented, so data can be read out as soon as `valid` goes high. After the data has been captured, then the signal `got` must be asserted.

Synchronous reset is used. Both resets may overlap.

`DATA_REG` (`=true`) is a hint, that distributed memory or registers should be used as data storage. The actual memory type depends on the device architecture. See implementation for details.

`*STATE_*_BITS` defines the granularity of the fill state indicator `*state_*`. `fstate_rd` is associated with the read clock domain and outputs the guaranteed number of words available in the FIFO. `estate_wr` is associated with the write clock domain and outputs the number of words that is guaranteed to be accepted by the FIFO

without a capacity overflow. Note that both these indicators cannot replace the `full` or `valid` outputs as they may be implemented as giving pessimistic bounds that are minimally off the true fill state.

If a fill state is not of interest, set `*STATE_*_BITS = 0`.

`fstate_rd` and `estate_wr` are combinatorial outputs and include an address comparator (subtractor) in their path.

Examples: - `FSTATE_RD_BITS = 1`: `fstate_rd == 0` => 0/2 full

`fstate_rd == 1` => 1/2 full (half full)

- **`FSTATE_RD_BITS = 2`**: `fstate_rd == 0` => 0/4 full `fstate_rd == 1` => 1/4 full `fstate_rd == 2` => 2/4 full `fstate_rd == 3` => 3/4 full

Entity Declaration:

```

1 entity fifo_ic_got is
2   generic (
3     D_BITS          : positive;          -- Data Width
4     MIN_DEPTH       : positive;          -- Minimum FIFO Depth
5     DATA_REG       : boolean := false;  -- Store Data Content in Registers
6     OUTPUT_REG      : boolean := false;  -- Registered FIFO Output
7     ESTATE_WR_BITS  : natural := 0;      -- Empty State Bits
8     FSTATE_RD_BITS  : natural := 0;      -- Full State Bits
9   );
10  port (
11    -- Write Interface
12    clk_wr   : in  std_logic;
13    rst_wr   : in  std_logic;
14    put      : in  std_logic;
15    din      : in  std_logic_vector(D_BITS-1 downto 0);
16    full     : out std_logic;
17    estate_wr : out std_logic_vector(imax(ESTATE_WR_BITS-1, 0) downto 0);
18
19    -- Read Interface
20    clk_rd   : in  std_logic;
21    rst_rd   : in  std_logic;
22    got      : in  std_logic;
23    valid    : out std_logic;
24    dout     : out std_logic_vector(D_BITS-1 downto 0);
25    fstate_rd : out std_logic_vector(imax(FSTATE_RD_BITS-1, 0) downto 0)
26  );
27 end entity fifo_ic_got;
```

Source file: `fifo/fifo_ic_got.vhdl`

7.9.8 PoC.fifo.shift

This FIFO implementation is based on an internal shift register. This is especially useful for smaller FIFO sizes, which can be implemented in LUT storage on some devices (e.g. Xilinx' SRLs). Only a single read pointer is maintained, which determines the number of valid entries within the underlying shift register.

The specified depth (`MIN_DEPTH`) is rounded up to the next suitable value.

Entity Declaration:

```

1 entity fifo_shift is
2   generic (
```

(continues on next page)

(continued from previous page)

```

3      D_BITS      : positive;           -- Data Width
4      MIN_DEPTH   : positive           -- Minimum FIFO Size in Words
5  );
6  port (
7      -- Global Control
8      clk : in std_logic;
9      rst : in std_logic;
10
11     -- Writing Interface
12     put : in std_logic;                 -- Write Request
13     din : in std_logic_vector(D_BITS-1 downto 0); -- Input Data
14     ful : out std_logic;                -- Capacity Exhausted
15
16     -- Reading Interface
17     got : in std_logic;                 -- Read Done Strobe
18     dout : out std_logic_vector(D_BITS-1 downto 0); -- Output Data
19     vld : out std_logic;                -- Data Valid
20 );
21 end entity fifo_shift;

```

Source file: fifo/fifo_shift.vhdl

7.10 PoC.io

The namespace `PoC.io` offers different general purpose I/O (GPIO) implementations, as well as low-speed bus protocol controllers.

Sub-namespaces

- *PoC.io.ddrio* - Double-Data-Rate (DDR) input/output abstraction layer.
- *PoC.io.iic* - I²C bus controllers
- *PoC.io.jtag* - JTAG implementations
- *PoC.io.lcd* - LC-Display bus controllers
- *PoC.io.mdio* - Management Data I/O (MDIO) controllers for Ethernet PHYs
- *PoC.io.ow* - OneWire / iButton bus controllers
- *PoC.io.ps2* - Periphery bus of the Personal System/2 (PS/2)
- *PoC.io.uart* - Universal Asynchronous Receiver Transmitter (UART) controllers
- *PoC.io.vga* - VGA, DVI, HDMI controllers

Package

The package *PoC.io* holds all enum, function and component declarations for this namespace.

Entities

- *PoC.io.Debounce*
- *PoC.io.7SegmentMux_BCD*
- *PoC.io.7SegmentMux_HEX*
- *PoC.io.FanControl*
- *PoC.io.FrequencyCounter*
- *PoC.io.GlitchFilter*
- *PoC.io.PulseWidthModulation*

- *PoC.io.TimingCounter*

7.10.1 PoC.io.ddrio

These are DDR-I/O (Double Data Rate - Input/Output) entities. ...

Entities

- *PoC.io.ddrio.in*
- *PoC.io.ddrio.inout*
- *PoC.io.ddrio.out*

PoC.io.ddrio Package

Source file: `ddrio.pkg.vhdl`

PoC.io.ddrio.in

Instantiates chip-specific DDR (Double Data Rate) input registers.

Both data `DataIn_high/low` are synchronously outputted to the on-chip logic with the rising edge of `Clock`. `DataIn_high` is the value at the Pad sampled with the same rising edge. `DataIn_low` is the value sampled with the falling edge directly before this rising edge. Thus sampling starts with the falling edge of the clock as depicted in the following waveform.

After power-up, the output ports `DataIn_high` and `DataIn_low` both equal `INIT_VALUE`.

Pad must be connected to a PAD because FPGAs only have these registers in IOBs.

Entity Declaration:

```
1 entity ddrio_in is
2   generic (
3     BITS          : positive;
4     INIT_VALUE    : bit_vector := x"FFFFFFF"
5   );
6   port (
7     Clock          : in    std_logic;
8     ClockEnable    : in    std_logic;
9     DataIn_high    : out   std_logic_vector (BITS - 1 downto 0);
10    DataIn_low     : out   std_logic_vector (BITS - 1 downto 0);
11    Pad            : in    std_logic_vector (BITS - 1 downto 0)
12  );
13 end entity;
```

Source file: `io/ddrio/ddrio_in.vhdl`

PoC.io.ddrio.inout

Instantiates chip-specific DDR input and output registers.

Both data `DataOut_high/low` as well as `OutputEnable` are sampled with the `rising_edge(Clock)` from the on-chip logic. `DataOut_high` is brought out with this rising edge. `DataOut_low` is brought out with the falling edge.

`OutputEnable` (Tri-State) is high-active. It is automatically inverted if necessary. Output is disabled after power-up.

Both data `DataIn_high/low` are synchronously outputted to the on-chip logic with the rising edge of `Clock`. `DataIn_high` is the value at the Pad sampled with the same rising edge. `DataIn_low` is the value sampled with the falling edge directly before this rising edge. Thus sampling starts with the falling edge of the clock as depicted in the following waveform.

Pad must be connected to a PAD because FPGAs only have these registers in IOBs.

Entity Declaration:

```

1 entity ddrrio_inout is
2   generic (
3     BITS          : positive
4   );
5   port (
6     ClockOut       : in    std_logic;
7     ClockOutEnable : in    std_logic;
8     OutputEnable   : in    std_logic;
9     DataOut_high   : in    std_logic_vector(BITS - 1 downto 0);
10    DataOut_low    : in    std_logic_vector(BITS - 1 downto 0);
11
12    ClockIn        : in    std_logic;
13    ClockInEnable  : in    std_logic;
14    DataIn_high    : out   std_logic_vector(BITS - 1 downto 0);
15    DataIn_low     : out   std_logic_vector(BITS - 1 downto 0);
16
17    Pad            : inout std_logic_vector(BITS - 1 downto 0)
18  );
19 end entity;
```

Source file: `io/ddrio/ddrio_inout.vhdl`

PoC.io.ddrio.out

Instantiates chip-specific DDR output registers.

Both data `DataOut_high/low` as well as `OutputEnable` are sampled with the rising edge (`Clock`) from the on-chip logic. `DataOut_high` is brought out with this rising edge. `DataOut_low` is brought out with the falling edge.

`OutputEnable` (Tri-State) is high-active. It is automatically inverted if necessary. If an output enable is not required, you may save some logic by setting `NO_OUTPUT_ENABLE = true`.

If `NO_OUTPUT_ENABLE = false` then output is disabled after power-up. If `NO_OUTPUT_ENABLE = true` then output after power-up equals `INIT_VALUE`.

Pad must be connected to a PAD because FPGAs only have these registers in IOBs.

Entity Declaration:

```

1 entity ddrrio_out is
2   generic (
3     NO_OUTPUT_ENABLE : boolean := false;
4     BITS              : positive;
5     INIT_VALUE        : bit_vector := x"FFFFFFFF"
6   );
7   port (
8     Clock       : in    std_logic;
9     ClockEnable : in    std_logic := '1';
10    OutputEnable : in    std_logic := '1';
```

(continues on next page)

(continued from previous page)

```
11     DataOut_high   : in std_logic_vector (BITS - 1 downto 0);
12     DataOut_low    : in std_logic_vector (BITS - 1 downto 0);
13     Pad            : out std_logic_vector (BITS - 1 downto 0)
14 );
15 end entity;
```

Source file: [io/ddrio/ddrio_out.vhdl](#)

7.10.2 PoC.io.iic

These are I2C entities. . . .

PoC.io.iic Package

Source file: [iic.pkg.vhdl](#)

PoC.io.iic.BusController

The I2C BusController transmits bits over the I2C bus (SerialClock - SCL, SerialData - SDA) and also receives them. To send/receive words over the I2C bus, use the I2C Controller, which utilizes this controller. This controller is compatible to the System Management Bus (SMBus).

Entity Declaration:

Source file: [io/iic/iic_BusController.vhdl](#)

PoC.io.iic.Controller

The I2C Controller transmits words over the I2C bus (SerialClock - SCL, SerialData - SDA) and also receives them. This controller utilizes the I2C BusController to send/receive bits over the I2C bus. This controller is compatible to the System Management Bus (SMBus).

Entity Declaration:

Source file: [io/iic/iic_Controller.vhdl](#)

PoC.io.iic.Switch_PCA9548A

Todo: No documentation available. TODO

Entity Declaration:

Source file: [io/iic/iic_Switch_PCA9548A.vhdl](#)

7.10.3 PoC.io.jtag

These are JTAG entities. . . .

7.10.4 PoC.io.lcd

These are LCD entities. . .

PoC.io.lcd.Package

Source file: [lcd.pkg.vhdl](#)

PoC.io.lcd.LCDBuffer

Todo: No documentation available.

Entity Declaration:

Source file: [io/lcd/lcd_LCDBuffer.vhdl](#)

PoC.io.lcd.LCDBusController

Todo: No documentation available.

Entity Declaration:

Source file: [io/lcd/lcd_LCDBusController.vhdl](#)

PoC.io.lcd.LCDController_KS0066U

Todo: No documentation available.

Entity Declaration:

Source file: [io/lcd/lcd_LCDController_KS0066U.vhdl](#)

PoC.io.lcd.LCDSynchronizer

Todo: No documentation available.

Entity Declaration:

Source file: [io/lcd/lcd_LCDSynchronizer.vhdl](#)

7.10.5 PoC.io.mdio

These are MDIO entities. . . .

mdio_BusController

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

PoC.io.mdio.Controller

Todo: No documentation available.

Entity Declaration:

Source file: [io/mdio/mdio_Controller.vhdl](#)

PoC.io.mdio.IIC_Adapter

Todo: No documentation available.

Entity Declaration:

Source file: [io/mdio/mdio_IIC_Adapter.vhdl](#)

7.10.6 PoC.io.ow

These are OneWire entities. . . .

ow_BusController

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

ow_Controller

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.10.7 PoC.io.pio

These are Pmod entities. . . .

PoC.io.pio.in

Entity Declaration:

Source file: [io/pio/pio_in.vhdl](#)

PoC.io.pio.out

Entity Declaration:

Source file: [io/pio/pio_out.vhdl](#)

PoC.io.pio.fifo_in

Entity Declaration:

Source file: [io/pio/pio_fifo_in.vhdl](#)

PoC.io.pio.fifo_out

Entity Declaration:

Source file: [io/pio/pio_fifo_out.vhdl](#)

7.10.8 PoC.io.pmod

These are Pmod entities. . . .

Entities

- *PoC.io.pmod.KYPD*
- *PoC.io.pmod.SSD*
- *PoC.io.pmod.USBUART*

PoC.io.pmod Package

Source file: [pmod.pkg.vhdl](#)

PoC.io.pmod.KYPD

This module drives a 4-bit one-cold encoded column vector to read back a 4-bit rows vector. By scanning column-by-column it's possible to extract the current button state of the whole keypad. This wrapper converts the high-active signals from *PoC.io.KeypadScanner* to low-active signals for the pmod. An additional debounce circuit filters the button signals. The scan frequency and bounce time can be configured.

Entity Declaration:

```

1 entity pmod_KYPD is
2   generic (
3     CLOCK_FREQ      : FREQ      := 100 MHz;
4     SCAN_FREQ       : FREQ      := 1 kHz;
5     BOUNCE_TIME     : time      := 10 ms
6   );
7   port (
8     Clock           : in  std_logic;
9     Reset           : in  std_logic;
10    -- Matrix interface
11    Keys             : out T_PMOD_KYPD_KEYPAD;
12    -- KeyPad interface
13    Columns_n        : out std_logic_vector(3 downto 0);
14    Rows_n           : in  std_logic_vector(3 downto 0)
15  );
16 end entity;
```

Source file: `io/pmod/pmod_KYPD.vhdl`

PoC.io.pmod.SSD

This module drives a dual-digit 7-segment display (Pmod_SSD). The module expects two binary encoded 4-bit Digit<i>i</i> signals and drives a 2x6 bit Pmod connector (7 anode bits, 1 cathode bit).

Segment	Pos./	Index
AAA		000
F B		5 1
F B		5 1
GGG		666
E C		4 2
E C		4 2
DDD DOT		333 7

Entity Declaration:

```

1 entity pmod_SSD is
2   generic (
3     CLOCK_FREQ      : FREQ      := 100 MHz;
4     REFRESH_RATE    : FREQ      := 1 kHz
5   );
6   port (
7     Clock           : in  std_logic;
8
9     Digit0          : in  std_logic_vector(3 downto 0);
10    Digit1          : in  std_logic_vector(3 downto 0);
11
12    SSD              : out T_PMOD_SSD_PINS
```

(continues on next page)

(continued from previous page)

```

13     );
14 end entity;
```

Source file: `io/pmod/pmod_SSD.vhdl`

PoC.io.pmod.USBUART

This module abstracts a FTDI FT232R USB-UART bridge by instantiating a *PoC.io.uart.fifo*. The FT232R supports up to 3 MBaud. A synchronous FIFO interface with a 32 words buffer is provided. Hardware flow control (RTS_CTS) is enabled.

Entity Declaration:

```

1  entity pmod_USBUART is
2      generic (
3          CLOCK_FREQ      : FREQ      := 100 MHz;
4          BAUDRATE        : BAUD      := 115200 Bd
5      );
6      port (
7          Clock           : in  std_logic;
8          Reset           : in  std_logic;
9
10         TX_put          : in  std_logic;
11         TX_Data         : in  std_logic_vector(7 downto 0);
12         TX_Full         : out std_logic;
13
14         RX_Valid        : out std_logic;
15         RX_Data         : out std_logic_vector(7 downto 0);
16         RX_got          : in  std_logic;
17
18         UART_TX         : out std_logic;
19         UART_RX         : in  std_logic;
20         UART_RTS        : out std_logic;
21         UART_CTS        : in  std_logic
22     );
23 end entity;
```

Source file: `io/pmod/pmod_USBUART.vhdl`

7.10.9 PoC.io.ps2

These are PS/2 entities....

7.10.10 PoC.io.uart

These are UART (Universal Asynchronous Receiver Transmitter) entities....

Entities

- *PoC.io.uart.bclk*
- *PoC.io.uart.rx*
- *PoC.io.uart.tx*
- *PoC.io.uart.fifo*

PoC.io.uart Package

Source file: `uart.pkg.vhdl`

PoC.io.uart.bclk

Todo: No documentation available.

old comments: UART BAUD rate generator `bclk_r` = bit clock is rising `bclk_x8_r` = bit clock times 8 is rising

Entity Declaration:

```
1 entity uart_bclk is
2   generic (
3     CLOCK_FREQ      : FREQ      := 100 MHz;
4     BAUDRATE        : BAUD      := 115200 Bd
5   );
6   port (
7     clk             : in  std_logic;
8     rst             : in  std_logic;
9     bclk            : out std_logic;
10    bclk_x8         : out std_logic
11  );
12 end entity;
```

Source file: `io/uart/uart_bclk.vhdl`

PoC.io.uart.rx

UART Receiver: 1 Start + 8 Data + 1 Stop

Entity Declaration:

```
1 entity uart_rx is
2   generic (
3     SYNC_DEPTH : natural := 2 -- use zero for already clock-synchronous rx
4   );
5   port (
6     -- Global Control
7     clk : in  std_logic;
8     rst : in  std_logic;
9
10    -- Bit Clock and RX Line
11    bclk_x8 : in  std_logic; -- bit clock, eight strobes per bit length
12    rx      : in  std_logic;
13
14    -- Byte Stream Output
15    do : out std_logic_vector(7 downto 0);
16    stb : out std_logic
17  );
18 end entity;
```

Source file: `io/uart/uart_rx.vhdl`

PoC.io.uart.tx

UART Transmitter: 1 Start + 8 Data + 1 Stop

Entity Declaration:

```

1 entity uart_tx is
2   port (
3     -- Global Control
4     clk : in std_logic;
5     rst : in std_logic;
6
7     -- Bit Clock and TX Line
8     bclk : in std_logic; -- bit clock, one strobe each bit length
9     tx   : out std_logic;
10
11    -- Byte Stream Input
12    di : in std_logic_vector(7 downto 0);
13    put : in std_logic;
14    ful : out std_logic
15  );
16 end entity;
```

Source file: io/uart/uart_tx.vhdl

PoC.io.uart.fifo

Small FIFO s are included in this module, if larger or asynchronous transmit / receive FIFOs are required, then they must be connected externally.

old comments: UART BAUD rate generator bclk = bit clock is rising bclk_x8 = bit clock times 8 is rising

Entity Declaration:

```

1 entity uart_fifo is
2   generic (
3     -- Communication Parameters
4     CLOCK_FREQ      : FREQ;
5     BAUDRATE        : BAUD;
6     ADD_INPUT_SYNCHRONIZERS : boolean := TRUE;
7
8     -- Buffer Dimensioning
9     TX_MIN_DEPTH    : positive := 16;
10    TX_ESTATE_BITS   : natural  := 0;
11    RX_MIN_DEPTH     : positive := 16;
12    RX_FSTATE_BITS   : natural  := 0;
13
14    -- Flow Control
15    FLOWCONTROL       : T_IO_UART_FLOWCONTROL_KIND := UART_FLOWCONTROL_
16    ↪ NONE;
17    SWFC_XON_CHAR     : std_logic_vector(7 downto 0) := x"11"; -- ^Q
18    SWFC_XON_TRIGGER  : real := 0.0625;
19    SWFC_XOFF_CHAR    : std_logic_vector(7 downto 0) := x"13"; -- ^S
20    SWFC_XOFF_TRIGGER : real := 0.75;
21  );
22  port (
23    Clock      : in std_logic;
24    Reset      : in std_logic;
```

(continues on next page)

(continued from previous page)

```

24
25  -- FIFO interface
26  TX_put      : in  std_logic;
27  TX_Data     : in  std_logic_vector(7 downto 0);
28  TX_Full     : out std_logic;
29  TX_EmptyState : out std_logic_vector(imax(0, TX_ESTATE_BITS-1) downto 0);
30
31  RX_Valid    : out std_logic;
32  RX_Data     : out std_logic_vector(7 downto 0);
33  RX_got      : in  std_logic;
34  RX_FullState : out std_logic_vector(imax(0, RX_FSTATE_BITS-1) downto 0);
35  RX_Overflow  : out std_logic;
36
37  -- External pins
38  UART_TX     : out std_logic;
39  UART_RX     : in  std_logic;
40  UART_RTS    : out std_logic;
41  UART_CTS    : in  std_logic
42  );
43  end entity;

```

Source file: `io/uart/uart_fifo.vhdl`

7.10.11 PoC.io.vga

These are VGA entities. . . .

PoC.io.vga Package

Source file: `vga.pkg.vhdl`

PoC.io.vga.phy

Entity Declaration:

Source file: `io/vga/vga_phy.vhdl`

PoC.io.vga.phy_ch7301c

The clock frequency must be the same as used for the timing module, e.g., 25 MHZ for VGA 640x480. A phase-shifted clock must be provided: - clk0 : 0 degrees - clk90 : 90 degrees

pixel_data(23 downto 16) : red pixel_data(15 downto 8) : green pixel_data(7 downto 0) : blue

The reset_b-pin must be driven by other logic (such as the reset button).

The IIC_interface is not part of this modules, as an IIC-master controls several slaves. The following registers must be set, see tests/ml505/vga_test_ml505.vhdl for an example.

Register	Value	Description
0x49 PM	0xC0 0xD0	Enable DVI, RGB bypass off Enable DVI, RGB bypass on
0x33 TPCP	0x08 if clk_freq <= 65 MHz else 0x06	
0x34 TPD	0x16 if clk_freq <= 65 MHz else 0x26	
0x36 TPF	0x60 if clk_freq <= 65 MHz else 0xA0	
0x1F IDF	0x80 0x90	when using SMT (VS0, HS0) when using CVT (VS1, HS0)
0x21 DC	0x09	Enable DAC if RGB bypass is on

Entity Declaration:

Source file: [io/vga/vga_phy_ch7301c.vhdl](#)

PoC.io.vga.timing**Configuration:**

MODE = 0: VGA mode with 640x480 pixels, 60 Hz, frequency(clk) ~ 25 MHz
 MODE = 1: HD 720p with 1280x720 pixels, 60 Hz, frequency(clk) = 74,5 MHz
 MODE = 2: HD 1080p with 1920x1080 pixels, 60 Hz, frequency(clk) = 138,5 MHz

MODE = 2 uses reduced blanking => only suitable for LCDs.

For MODE = 0, CVT can be configured: - CVT = false: Use Safe Mode Timing (SMT).

The legacy fall-back mode supported by CRTs as well as LCDs. HSync: low-active. VSync: low-active. frequency(clk) = 25.175 MHz. (25 MHz works => 31 kHz / 59 Hz)

- **CVT = true: The “new” Coordinated Video Timing (since 2003).** The CVT supports some new features, such as reduced blanking (for LCDs) or aspect ratio encoding. See the web for more details. Standard CRT-based timing (CVT-GTF) has been implemented for best compatibility: HSync: low-active. VSync: high-active. frequency(clk) = 23.75 MHz. (25 MHz works => 31 kHz / 62 Hz)

Usage:

The frequency of `clk` must be equal to the pixel clock frequency of the selected video mode, see also above.

When using analog output, the VGA color signals must be blanked, during horizontal and vertical beam return. This could be achieved by combinatorial “anding” the color value with “beam_on” (part of “phy_ctrl”) inside the PHY.

When using digital output (DVI), then “beam_on” is equal to “DE” (Data Enable) of the DVI transmitter.

`xvalid` and `yvalid` show if `xpos` respectively `ypos` are in a valid range. `beam_on` is ‘1’ iff both `xvalid` and `yvalid` = ‘1’.

`xpos` and `ypos` also show the pixel location during blanking. This might be useful in some applications. But be careful, that the ranges differ between SMT and CVT.

Entity Declaration:

Source file: [io/vga/vga_timing.vhdl](#)

7.10.12 PoC.io Package

This package holds all component declarations for this namespace.

Source file: [io.pkg.vhdl](#)

7.10.13 PoC.io.7SegmentMux_BCD

This module is a 7 segment display controller that uses time multiplexing to control a common anode for each digit in the display. The shown characters are BCD encoded. A dot per digit is optional. A minus sign for negative numbers is supported.

Entity Declaration:

```
1  entity io_7SegmentMux_BCD is
2      generic (
3          CLOCK_FREQ      : FREQ          := 100 MHz;
4          REFRESH_RATE    : FREQ          := 1 kHz;
5          DIGITS           : positive      := 4
6      );
7      port (
8          Clock            : in  std_logic;
9
10         BCDDigits        : in  T_BCD_VECTOR(DIGITS - 1 downto 0);
11         BCDDots           : in  std_logic_vector(DIGITS - 1 downto 0);
12
13         SegmentControl    : out std_logic_vector(7 downto 0);
14         DigitControl      : out std_logic_vector(DIGITS - 1 downto 0)
15     );
16 end entity;
```

Source file: [io/io_7SegmentMux_BCD.vhdl](#)

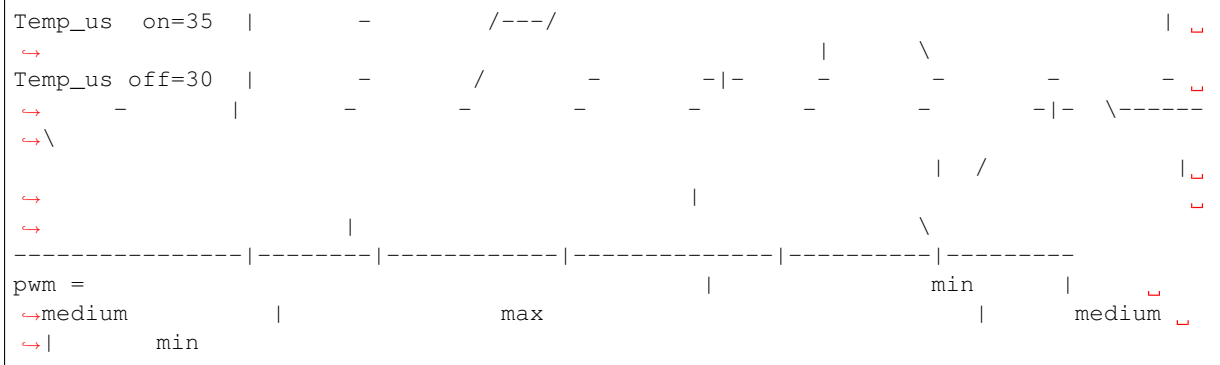
7.10.14 PoC.io.7SegmentMux_HEX

This module is a 7 segment display controller that uses time multiplexing to control a common anode for each digit in the display. The shown characters are HEX encoded. A dot per digit is optional.

Entity Declaration:

```
1  entity io_7SegmentMux_HEX is
2      generic (
3          CLOCK_FREQ      : FREQ          := 100 MHz;
4          REFRESH_RATE    : FREQ          := 1 kHz;
5          DIGITS           : positive      := 4
6      );
7      port (
8          Clock            : in  std_logic;
9
10         HexDigits         : in  T_SLVV_4(DIGITS - 1 downto 0);
11         HexDots            : in  std_logic_vector(DIGITS - 1 downto 0);
12
13         SegmentControl    : out std_logic_vector(7 downto 0);
14         DigitControl      : out std_logic_vector(DIGITS - 1 downto 0)
15     );
16 end entity;
```


(continued from previous page)

**Entity Declaration:**

```

1  entity io_FanControl is
2      generic (
3          CLOCK_FREQ          : FREQ;
4          ADD_INPUT_SYNCHRONIZERS : boolean := TRUE;
5          ENABLE_TACHO         : boolean := FALSE
6      );
7      port (
8          -- Global Control
9          Clock          : in std_logic;
10         Reset           : in std_logic;
11
12         -- Fan Control derived from internal System Health Monitor
13         Fan_PWM         : out std_logic;
14
15         -- Decoding of Speed Sensor (Requires ENABLE_TACHO)
16         Fan_Tacho       : in std_logic := 'X';
17         TachoFrequency : out std_logic_vector(15 downto 0)
18     );
19 end entity;

```

Source file: io/io_FanControl.vhdl

7.10.17 PoC.io.FrequencyCounter**Todo:** No documentation available.**Entity Declaration:**

```

1  entity io_FrequencyCounter is
2      generic (
3          CLOCK_FREQ          : FREQ          := 100 MHz;
4          TIMEBASE             : time          := 1 sec;
5          RESOLUTION           : positive      := 8
6      );
7      port (
8          Clock                : in std_logic;
9          Reset                 : in std_logic;
10         FreqIn                : in std_logic;
11         FreqOut               : out std_logic_vector(RESOLUTION - 1 downto 0)

```

(continues on next page)

(continued from previous page)

```

12     );
13 end entity;

```

Source file: `io/io_FrequencyCounter.vhdl`

7.10.18 PoC.io.GlitchFilter

This module filters glitches on a wire. The high and low spike suppression cycle counts can be configured.

Entity Declaration:

```

1 entity io_GlitchFilter is
2   generic (
3     HIGH_SPIKE_SUPPRESSION_CYCLES : natural := 5;
4     LOW_SPIKE_SUPPRESSION_CYCLES  : natural := 5;
5   );
6   port (
7     Clock   : in std_logic;
8     Input   : in std_logic;
9     Output  : out std_logic
10  );
11 end entity;

```

Source file: `io/io_GlitchFilter.vhdl`

7.10.19 PoC.io.KeyPadScanner

This module drives a one-hot encoded column vector to read back a rows vector. By scanning column-by-column it's possible to extract the current button state of the whole keypad. The scanner uses high-active logic. The keypad size and scan frequency can be configured. The outputed signal matrix is not debounced.

Entity Declaration:

```

1 entity io_KeyPadScanner is
2   generic (
3     CLOCK_FREQ      : FREQ := 100 MHz;
4     SCAN_FREQ       : FREQ := 1 kHz;
5     ROWS             : positive := 4;
6     COLUMNS         : positive := 4;
7     ADD_INPUT_SYNCHRONIZERS : boolean := TRUE;
8   );
9   port (
10    Clock      : in std_logic;
11    Reset      : in std_logic;
12    -- Matrix interface
13    KeyPadMatrix : out T_SLM(COLUMNS - 1 downto 0, ROWS - 1 downto 0);
14    -- KeyPad interface
15    ColumnVector : out std_logic_vector(COLUMNS - 1 downto 0);
16    RowVector    : in std_logic_vector(ROWS - 1 downto 0)
17  );
18 end entity;

```

Source file: `io/io_KeyPadScanner.vhdl`

7.10.20 PoC.io.PulseWidthModulation

This module generates a pulse width modulated signal, that can be configured in frequency (PWM_FREQ) and modulation granularity (PWM_RESOLUTION).

Entity Declaration:

```

1 entity io_PulseWidthModulation is
2   generic (
3     CLOCK_FREQ          : FREQ          := 100 MHz;
4     PWM_FREQ            : FREQ          := 1 kHz;
5     PWM_RESOLUTION      : positive      := 8
6   );
7   port (
8     Clock               : in  std_logic;
9     Reset               : in  std_logic;
10    PWMIn                : in  std_logic_vector (PWM_RESOLUTION - 1 downto 0);
11    PWMOut               : out std_logic
12  );
13 end entity;
```

Source file: `io/io_PulseWidthModulation.vhdl`

7.10.21 PoC.io.TimingCounter

This down-counter can be configured with a TIMING_TABLE (a ROM), from which the initial counter value is loaded. The table index can be selected by Slot. Timeout is a registered output. Up to 16 values fit into one ROM consisting of $\log_2 \text{ceil}(\text{imax}(\text{TIMING_TABLE})) + 1$ 6-input LUTs.

Entity Declaration:

```

1 entity io_TimingCounter is
2   generic (
3     TIMING_TABLE : T_NATVEC -- timing_
4   )
5   port (
6     Clock       : in  std_logic; -- clock
7     Enable      : in  std_logic; -- enable_
8     Load        : in  std_logic; -- load_
9     Slot         : in  natural range 0 to (TIMING_TABLE'length - 1); --
10    Timeout      : out std_logic -- timing_
11  )
12  reached
13 end entity;
```

Source file: `io/io_TimingCounter.vhdl`

7.11 PoC.mem

The namespace `PoC.mem` offers different on-chip and off-chip memory and memory-controller implementations.

Sub-Namespaces

- *PoC.mem.ldr2* - DDR2 memory controllers
- *PoC.mem.ldr3* - DDR3 memory controllers
- *PoC.mem.lut* - Lookup-Table (LUT) implementations
- *PoC.mem.ocram* - On-Chip RAM abstraction layer
- *PoC.mem.ocrom* - On-Chip ROM abstraction layer
- *PoC.mem.sdram* - SDRAM controllers

Package

PoC.mem

7.11.1 PoC.mem Package

This package holds all component declarations, types and functions of the *PoC.mem* namespace.

It provides the following enumerations:

- `T_MEM_FILEFORMAT` specifies whether a file is in Intel Hex, Lattice Mem, or Xilinx Mem format.
- `T_MEM_CONTENT` specifies whether data in text file is in binary, decimal or hexadecimal format.

It provides the following functions:

- `mem_FileExtension` returns the file extension of a given filename.
- `mem_ReadMemoryFile` reads initial memory content from a given file.

Source file: [mem.pkg.vhdl](#)

7.11.2 PoC.mem.ldr2

The namespace `PoC.mem.ldr2` is designated for own implementations of DDR2 memory controllers as well as for adapters for vendor-specific implementations. At the top-level, all controllers and adapters provide the same simple memory interface to the user application.

Entities

- *PoC.mem.ldr2.mem2mig_adapter_Spartan6* - Adapter for the Xilinx MIG core for Spartan-6 FPGAs

PoC.mem.ldr2.mem2mig_adapter_Spartan6

Adapter between the *PoC.Mem* interface and the User Interface of the Xilinx MIG IP core for the Spartan-6 FPGA Memory Controller Block (MCB). The MCB can be configured to have multiple ports. One instance of this adapter is required for every port. The control signals for one port of the MIG IP core are prefixed by “cX_pY”, meaning port Y on controller X.

Simplifies the User Interface (“user”) of the Xilinx MIG IP core (UG388). The PoC.Mem interface provides single-cycle fully pipelined read/write access to the memory. All accesses are word-aligned. Always all bytes of a word are written to the memory. More details can be found [here](#).

Generic parameters:

- `D_BITS`: Data bus width of the PoC.Mem and MIG / MCBinterface. Also size of one word in bits.
- `MEM_A_BITS`: Address bus width of the PoC.Mem interface.
- `APP_A_BTIS`: Address bus width of the MIG / MCB interface.

Contains only combinational logic.

Entity Declaration:

```

1  entity ddr2_mem2mig_adapter_Spartan6 is
2
3      generic (
4          D_BITS      : positive;
5          MEM_A_BITS  : positive;
6          APP_A_BITS  : positive
7      );
8
9      port (
10         -- PoC.Mem interface
11         mem_req      : in  std_logic;
12         mem_write    : in  std_logic;
13         mem_addr     : in  unsigned(MEM_A_BITS-1 downto 0);
14         mem_wdata    : in  std_logic_vector(D_BITS-1 downto 0);
15         mem_wmask    : in  std_logic_vector(D_BITS/8-1 downto 0) := (others => '0');
16         mem_rdy      : out std_logic;
17         mem_rstb     : out std_logic;
18         mem_rdata    : out std_logic_vector(D_BITS-1 downto 0);
19
20         -- Xilinx MIG IP Core interface
21         mig_calib_done : in  std_logic;
22         mig_cmd_full   : in  std_logic;
23         mig_wr_full    : in  std_logic;
24         mig_rd_empty   : in  std_logic;
25         mig_rd_data    : in  std_logic_vector((D_BITS)-1 downto 0);
26         mig_cmd_instr  : out std_logic_vector(2 downto 0);
27         mig_cmd_en     : out std_logic;
28         mig_cmd_bl     : out std_logic_vector(5 downto 0);
29         mig_cmd_byte_addr : out std_logic_vector(APP_A_BITS-1 downto 0);
30         mig_wr_data    : out std_logic_vector((D_BITS)-1 downto 0);
31         mig_wr_mask    : out std_logic_vector((D_BITS)/8-1 downto 0);
32         mig_wr_en     : out std_logic;
33         mig_rd_en     : out std_logic
34     );
35
36 end entity ddr2_mem2mig_adapter_Spartan6;

```

Source file: `mem/ddr2/ddr2_mem2mig_adapter_Spartan6.vhdl`

7.11.3 PoC.mem.ddr3

The namespace `PoC.mem.ddr3` is designated for own implementations of DDR3 memory controllers as well as for adapters for vendor-specific implementations. At the top-level, all controllers and adapters provide the same simple memory interface to the user application.

Entities

- *PoC.mem.ddr3.mem2mig_adapter_Series7* - Adapter for the Xilinx MIG core for 7-Series FPGAs

PoC.mem.ddr3.mem2mig_adapter_Series7

Adapter between the *PoC.Mem* interface and the application interface (“app”) of the Xilinx MIG IP core for 7-Series FPGAs.

Simplifies the application interface (“app”) of the Xilinx MIG IP core. The *PoC.Mem* interface provides single-cycle fully pipelined read/write access to the memory. All accesses are word-aligned. Always all bytes of a word are written to the memory. More details can be found [here](#).

Generic parameters:

- D_BITS: Data bus width of the PoC.Mem and “app” interface. Also size of one word in bits.
- DQ_BITS: Size of data bus between memory controller and external memory (DIMM, SoDIMM).
- MEM_A_BITS: Address bus width of the PoC.Mem interface.
- APP_A_BITS: Address bus width of the “app” interface.

Contains only combinational logic.

Entity Declaration:

```

1  entity ddr3_mem2mig_adapter_Series7 is
2
3      generic (
4          D_BITS      : positive;
5          DQ_BITS     : positive;
6          MEM_A_BITS  : positive;
7          APP_A_BITS  : positive
8      );
9
10     port (
11         -- PoC.Mem interface
12         mem_req      : in  std_logic;
13         mem_write    : in  std_logic;
14         mem_addr     : in  unsigned(MEM_A_BITS-1 downto 0);
15         mem_wdata    : in  std_logic_vector(D_BITS-1 downto 0);
16         mem_wmask    : in  std_logic_vector(D_BITS/8-1 downto 0) := (others => '0');
17         mem_rdy      : out std_logic;
18         mem_rstb     : out std_logic;
19         mem_rdata    : out std_logic_vector(D_BITS-1 downto 0);
20
21         -- Xilinx MIG IP Core interface
22         init_calib_complete : in  std_logic;
23         app_rd_data        : in  std_logic_vector((D_BITS)-1 downto 0);
24         app_rd_data_end    : in  std_logic;
25         app_rd_data_valid  : in  std_logic;
26         app_rdy            : in  std_logic;
27         app_wdf_rdy        : in  std_logic;
28         app_addr           : out std_logic_vector(APP_A_BITS-1 downto 0);
29         app_cmd            : out std_logic_vector(2 downto 0);
30         app_en             : out std_logic;
31         app_wdf_data       : out std_logic_vector((D_BITS)-1 downto 0);
32         app_wdf_end        : out std_logic;
33         app_wdf_mask       : out std_logic_vector((D_BITS)/8-1 downto 0);
34         app_wdf_wren       : out std_logic
35     );
36
37 end entity ddr3_mem2mig_adapter_Series7;

```

Source file: mem/ddr3/ddr3_mem2mig_adapter_Series7.vhdl

7.11.4 PoC.mem.lut

The namespace `PoC.mem.lut` offers different lookup-tables (LUTs).

Entities

- *PoC.mem.lut.Sine* - a Sine implementation with 1,2 or 4 quadrants.

PoC.mem.lut.Sine

Todo: No documentation available.

Entity Declaration:

Source file: `mem/lut/lut_Sine.vhdl`

7.11.5 PoC.mem.ocram

The namespace `PoC.mem.ocram` offers different on-chip RAM abstractions.

Package

The package `PoC.mem.ocram` holds all component declarations for this namespace.

```
library PoC;
use     PoC.ocram.all;
```

Entities

- *PoC.mem.ocram.sp* - An on-chip RAM with a single port interface.
- *PoC.mem.ocram.sdp* - An on-chip RAM with a simple dual-port interface.
- *PoC.mem.ocram.sdp_wf* - An on-chip RAM with a simple dual-port interface and write-first behavior.
- *PoC.mem.ocram.tdp* - An on-chip RAM with a true dual-port interface.
- *PoC.mem.ocram.tdp_wf* - An on-chip RAM with a true dual-port interface and write-first behavior.

Simulation Helper

- *PoC.mem.ocram.tdp_sim* - Simulation model of on-chip RAM with a true dual port interface.

Deprecated Entities

- *PoC.mem.ocram.esdp* - An on-chip RAM with an extended simple dual port interface.

PoC.mem.ocram Package

Source file: `ocram.pkg.vhdl`

PoC.mem.ocram.sp

Inferring / instantiating single port memory, with:

- single clock, clock enable,
- 1 read/write port.

Command Truth Table:

ce	we	Command
0	X	No operation
1	0	Read from memory
1	1	Write to memory

Both reading and writing are synchronous to the rising-edge of the clock. Thus, when reading, the memory data will be outputted after the clock edge, i.e, in the following clock cycle.

When writing data, the read output will output the new data (in the following clock cycle) which is aka. “write-first behavior”. This behavior also applies to Altera M20K memory blocks as described in the Altera: “Stratix 5 Device Handbook” (S5-5V1). The documentation in the Altera: “Embedded Memory User Guide” (UG-01068) is wrong.

Entity Declaration:

```

1 entity ocram_sp is
2   generic (
3     A_BITS      : positive;           -- number of address bits
4     D_BITS      : positive;           -- number of data bits
5     FILENAME    : string := ""        -- file-name for RAM_
6   )
7   port (
8     clk : in std_logic;               -- clock
9     ce  : in std_logic;               -- clock enable
10    we  : in std_logic;               -- write enable
11    a   : in unsigned(A_BITS-1 downto 0); -- address
12    d   : in std_logic_vector(D_BITS-1 downto 0); -- write data
13    q   : out std_logic_vector(D_BITS-1 downto 0); -- read output
14  );
15 end entity;
```

Source file: [mem/ocram/ocram_sp.vhdl](#)

PoC.mem.ocram.sdp

Inferring / instantiating simple dual-port memory, with:

- dual clock, clock enable,
- 1 read port plus 1 write port.

Both reading and writing are synchronous to the rising-edge of the clock. Thus, when reading, the memory data will be outputted after the clock edge, i.e, in the following clock cycle.

The generalized behavior across Altera and Xilinx FPGAs since Stratix/Cyclone and Spartan-3/Virtex-5, respectively, is as follows:

Mixed-Port Read-During-Write When reading at the write address, the read value will be unknown which is aka. “don’t care behavior”. This applies to all reads (at the same address) which are issued during the write-cycle time, which starts at the rising-edge of the write clock and (in the worst case) extends until the next rising-edge of the write clock.

For simulation, always our dedicated simulation model *PoC.mem.ocram.tdp_sim* is used.

Entity Declaration:

```

1 entity ocram_sdp is
2   generic (
3     A_BITS      : positive;           -- number of address bits
4     D_BITS      : positive;           -- number of data bits
5     FILENAME    : string := ""        -- file-name for RAM_
6   )
7   port (
8     rclk : in std_logic;               -- read clock
9     rce  : in std_logic;               -- read clock-enable
```

(continues on next page)

(continued from previous page)

```

10   wclk : in  std_logic;           -- write clock
11   wce  : in  std_logic;           -- write clock-enable
12   we   : in  std_logic;           -- write enable
13   ra   : in  unsigned(A_BITS-1 downto 0); -- read address
14   wa   : in  unsigned(A_BITS-1 downto 0); -- write address
15   d    : in  std_logic_vector(D_BITS-1 downto 0); -- data in
16   q    : out std_logic_vector(D_BITS-1 downto 0) -- data out
17 );
18 end entity;

```

Source file: `mem/ocram/ocram_sdp.vhdl`

PoC.mem.ocram.sdp_wf

Inferring / instantiating simple dual-port memory, with:

- single clock, clock enable,
- 1 read port plus 1 write port.

Command truth table:

ce	we	Command
0	X	No operation
1	0	Read only from memory
1	1	Read from and Write to memory

Both reading and writing are synchronous to the rising-edge of the clock. Thus, when reading, the memory data will be outputted after the clock edge, i.e, in the following clock cycle.

Mixed-Port Read-During-Write When reading at the write address, the read value will be the new data, aka. “write-first behavior”. Of course, the read is still synchronous, i.e, the latency is still one clock cycle.

Entity Declaration:

```

1  entity ocram_sdp_wf is
2    generic (
3      A_BITS    : positive;           -- number of address bits
4      D_BITS    : positive;           -- number of data bits
5      FILENAME  : string := "";       -- file-name for RAM_
6      ↪initialization
7    );
8    port (
9      clk : in  std_logic;           -- clock
10     ce  : in  std_logic;           -- clock-enable
11     we  : in  std_logic;           -- write enable
12     ra  : in  unsigned(A_BITS-1 downto 0); -- read address
13     wa  : in  unsigned(A_BITS-1 downto 0); -- write address
14     d   : in  std_logic_vector(D_BITS-1 downto 0); -- data in
15     q   : out std_logic_vector(D_BITS-1 downto 0) -- data out
16   );
17 end entity;

```

Source file: `mem/ocram/ocram_sdp_wf.vhdl`

PoC.mem.ocram.tdp

Inferring / instantiating true dual-port memory, with:

- dual clock, clock enable,
- 2 read/write ports.

Command truth table for port 1, same applies to port 2:

ce1	we1	Command
0	X	No operation
1	0	Read from memory
1	1	Write to memory

Both reading and writing are synchronous to the rising-edge of the clock. Thus, when reading, the memory data will be outputted after the clock edge, i.e, in the following clock cycle.

The generalized behavior across Altera and Xilinx FPGAs since Stratix/Cyclone and Spartan-3/Virtex-5, respectively, is as follows:

Same-Port Read-During-Write When writing data through port 1, the read output of the same port ($q1$) will output the new data ($d1$, in the following clock cycle) which is aka. “write-first behavior”.

Same applies to port 2.

Mixed-Port Read-During-Write When reading at the write address, the read value will be unknown which is aka. “don’t care behavior”. This applies to all reads (at the same address) which are issued during the write-cycle time, which starts at the rising-edge of the write clock and (in the worst case) extends until the next rising-edge of that write clock.

For simulation, always our dedicated simulation model *PoC.mem.ocram.tdp_sim* is used.

Entity Declaration:

```

1 entity ocram_tdp is
2   generic (
3     A_BITS    : positive;           -- number of address bits
4     D_BITS    : positive;           -- number of data bits
5     FILENAME  : string := "";       -- file-name for RAM_
6   )
7   port (
8     clk1 : in std_logic;           -- clock for 1st port
9     clk2 : in std_logic;           -- clock for 2nd port
10    ce1  : in std_logic;           -- clock-enable for 1st port
11    ce2  : in std_logic;           -- clock-enable for 2nd port
12    we1  : in std_logic;           -- write-enable for 1st port
13    we2  : in std_logic;           -- write-enable for 2nd port
14    a1   : in unsigned(A_BITS-1 downto 0); -- address for 1st port
15    a2   : in unsigned(A_BITS-1 downto 0); -- address for 2nd port
16    d1   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 1st port
17    d2   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 2nd port
18    q1   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 1st port
19    q2   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 2nd port
20  );
21 end entity;
```

Source file: `mem/ocram/ocram_tdp.vhdl`

PoC.mem.ocram.tdp_wf

Inferring / instantiating true dual-port memory, with:

- single clock, clock enable,

- 2 read/write ports.

Command truth table:

ce	we1	we2	Command
0	X	X	No operation
1	0	0	Read only from memory
1	0	1	Read from memory on port 1, write to memory on port 2
1	1	0	Write to memory on port 1, read from memory on port 2
1	1	1	Write to memory on both ports

Both reads and writes are synchronous to the clock.

The generalized behavior across Altera and Xilinx FPGAs since Stratix/Cyclone and Spartan-3/Virtex-5, respectively, is as follows:

Same-Port Read-During-Write When writing data through port 1, the read output of the same port (q1) will output the new data (d1, in the following clock cycle) which is aka. “write-first behavior”.

Same applies to port 2.

Mixed-Port Read-During-Write When reading at the write address, the read value will be the new data, aka. “write-first behavior”. Of course, the read is still synchronous, i.e, the latency is still one clock cycle.

If a write is issued on both ports to the same address, then the output of this unit and the content of the addressed memory cell are undefined.

For simulation, always our dedicated simulation model *PoC.mem.ocram.tdp_sim* is used.

Entity Declaration:

```

1  entity ocram_tdp_wf is
2      generic (
3          A_BITS      : positive;           -- number of address bits
4          D_BITS      : positive;           -- number of data bits
5          FILENAME     : string := "";      -- file-name for RAM_
6      ) -- initialization
7  port (
8      clk : in  std_logic;                  -- clock
9      ce  : in  std_logic;                  -- clock-enable
10     we1 : in  std_logic;                  -- write-enable for 1st port
11     we2 : in  std_logic;                  -- write-enable for 2nd port
12     a1  : in  unsigned(A_BITS-1 downto 0); -- address for 1st port
13     a2  : in  unsigned(A_BITS-1 downto 0); -- address for 2nd port
14     d1  : in  std_logic_vector(D_BITS-1 downto 0); -- write-data for 1st port
15     d2  : in  std_logic_vector(D_BITS-1 downto 0); -- write-data for 2nd port
16     q1  : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 1st port
17     q2  : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 2nd port
18 );
19 end entity;
```

Source file: `mem/ocram/ocram_tdp_wf.vhdl`

PoC.mem.ocram.tdp_sim

Simulation model for true dual-port memory, with:

- dual clock, clock enable,
- 2 read/write ports.

The interface matches that of the IP core PoC.mem.ocram.tdp. But the implementation there is restricted to the description supported by various synthesis compilers. The implementation here also simulates the correct Mixed-Port Read-During-Write Behavior and handles X propagation.

Entity Declaration:

```

1 entity ocram_tdp_sim is
2   generic (
3     A_BITS      : positive;           -- number of address bits
4     D_BITS      : positive;           -- number of data bits
5     FILENAME    : string := ""        -- file-name for RAM_
6   )
7   port (
8     clk1 : in std_logic;              -- clock for 1st port
9     clk2 : in std_logic;              -- clock for 2nd port
10    ce1  : in std_logic;              -- clock-enable for 1st port
11    ce2  : in std_logic;              -- clock-enable for 2nd port
12    we1  : in std_logic;              -- write-enable for 1st port
13    we2  : in std_logic;              -- write-enable for 2nd port
14    a1   : in unsigned(A_BITS-1 downto 0); -- address for 1st port
15    a2   : in unsigned(A_BITS-1 downto 0); -- address for 2nd port
16    d1   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 1st port
17    d2   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 2nd port
18    q1   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 1st port
19    q2   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 2nd port
20  );
21 end entity;

```

Source file: [mem/ocram/ocram_tdp_sim.vhdl](#)

PoC.mem.ocram.esdp

Inferring / instantiating enhanced simple dual-port memory, with:

- dual clock, clock enable,
- 1 read/write port (1st port) plus 1 read port (2nd port).

Deprecated since version 1.1: Please use [PoC.mem.ocram.tdp](#) for new designs. This component has been provided because older FPGA compilers were not able to infer true dual-port memory from an RTL description.

Command truth table for port 1:

ce1	we1	Command
0	X	No operation
1	0	Read from memory
1	1	Write to memory

Command truth table for port 2:

ce2	Command
0	No operation
1	Read from memory

Both reading and writing are synchronous to the rising-edge of the clock. Thus, when reading, the memory data will be outputted after the clock edge, i.e, in the following clock cycle.

The generalized behavior across Altera and Xilinx FPGAs since Stratix/Cyclone and Spartan-3/Virtex-5, respectively, is as follows:

Same-Port Read-During-Write When writing data through port 1, the read output of the same port (`q1`) will output the new data (`d1`, in the following clock cycle) which is aka. “write-first behavior”.

Mixed-Port Read-During-Write When reading at the write address, the read value will be unknown which is aka. “don’t care behavior”. This applies to all reads (at the same address) which are issued during the write-cycle time, which starts at the rising-edge of the write clock (`clk1`) and (in the worst case) extends until the next rising-edge of the write clock.

For simulation, always our dedicated simulation model *PoC.mem.ocram.tdp_sim* is used.

Entity Declaration:

```

1  entity ocram_esdp is
2      generic (
3          A_BITS      : positive;           -- number of address bits
4          D_BITS      : positive;           -- number of data bits
5          FILENAME    : string := ""        -- file-name for RAM_
6      ) -- initialization
7      port (
8          clk1 : in std_logic;              -- clock for 1st port
9          clk2 : in std_logic;              -- clock for 2nd port
10         ce1  : in std_logic;              -- clock-enable for 1st port
11         ce2  : in std_logic;              -- clock-enable for 2nd port
12         we1  : in std_logic;              -- write-enable for 1st port
13         a1   : in unsigned(A_BITS-1 downto 0); -- address for 1st port
14         a2   : in unsigned(A_BITS-1 downto 0); -- address for 2nd port
15         d1   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 1st port
16         q1   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 1st port
17         q2   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 2nd port
18     );
19 end entity;
```

Source file: `mem/ocram/ocram_esdp.vhdl`

7.11.6 PoC.mem.ocrom

The namespace `PoC.mem.ocrom` offers different on-chip ROM abstractions.

Package

The package `PoC.mem.ocrom` holds all component declarations for this namespace.

```

library PoC;
use      PoC.ocrom.all;
```

Entities

- *ocrom_sp* is a on-chip RAM with a single port interface.
- *ocrom_dp* is a on-chip RAM with a dual port interface.

PoC.mem.ocrom Package

Source file: `ocrom.pkg.vhdl`

PoC.mem.ocrom.sp

Inferring / instantiating single-port read-only memory

- single clock, clock enable
- 1 read port

Entity Declaration:

```

1 entity ocrom_sp is
2   generic (
3     A_BITS    : positive;
4     D_BITS    : positive;
5     FILENAME  : string    := ""
6   );
7   port (
8     clk : in  std_logic;
9     ce  : in  std_logic;
10    a   : in  unsigned(A_BITS-1 downto 0);
11    q   : out std_logic_vector(D_BITS-1 downto 0)
12  );
13 end entity;
```

Source file: mem/ocrom/ocrom_sp.vhdl

PoC.mem.ocrom.dp

Inferring / instantiating dual-port read-only memory, with:

- dual clock, clock enable,
- 2 read ports.

The generalized behavior across Altera and Xilinx FPGAs since Stratix/Cyclone and Spartan-3/Virtex-5, respectively, is as follows:

WARNING: The simulated behavior on RT-level is not correct.

TODO: add timing diagram TODO: implement correct behavior for RT-level simulation

Entity Declaration:

```

1 entity ocrom_dp is
2   generic (
3     A_BITS    : positive;
4     D_BITS    : positive;
5     FILENAME  : string    := ""
6   );
7   port (
8     clk1 : in  std_logic;
9     clk2 : in  std_logic;
10    ce1  : in  std_logic;
11    ce2  : in  std_logic;
12    a1   : in  unsigned(A_BITS-1 downto 0);
13    a2   : in  unsigned(A_BITS-1 downto 0);
14    q1   : out std_logic_vector(D_BITS-1 downto 0);
15    q2   : out std_logic_vector(D_BITS-1 downto 0)
16  );
17 end entity;
```

Source file: `mem/ocrom/ocrom_dp.vhdl`

7.11.7 PoC.mem.sdram

The namespace `PoC.mem.sdram` offers components for the access of external SDRAMs. A common finite state-machine is used to address the memory via banks, rows and columns. Different physical layers are provide for the single-data-rate (SDR) or double-data-rate (DDR, DDR2, ...) data bus. One has to instantiate the specific module required by the FPGA board.

SDRAM Controller for the Altera DE0 Board

The module `sdram_ctrl_de0` combines the finite state machine `sdram_ctrl_fsm` and the DE0 specific physical layer `sdram_ctrl_phy_de0`. It has been tested with the IS42S16400F SDR memory at a frequency of 133 MHz. A usage example is given in [PoC-Examples](#).

SDRAM Controller for the Xilinx Spartan-3E Starter Kit (S3ESK)

The module `sdram_ctrl_s3esk` combines the finite state machine `sdram_ctrl_fsm` and the S3ESK specific physical layer `sdram_ctrl_phy_s3esk`. It has been tested with the MT46V32M16-6T DDR memory at a frequency of 100 MHz (DDR-200). A usage example is given in [PoC-Examples](#).

Note: See also [PoC.xil.mig](#) for board specific memory controller implementations created by Xilinx's Memory Interface Generator (MIG).

PoC.mem.sdram.ctrl_fsm

This file contains the FSM as well as parts of the datapath. The board specific physical layer is defined in another file.

Configuration

SDRAM_TYPE activates some special cases:

- 0 for SDR-SDRAM
- 1 for DDR-SDRAM
- 2 for DDR2-SDRAM (no special support yet like ODT)

`2**A_BITS` specifies the number of memory cells in the SDRAM. This is the size of th memory in bits divided by the native data-path width of the SDRAM (also in bits).

`D_BITS` is the native data-path width of the SDRAM. The width might be doubled by the physical interface for DDR interfaces.

Furthermore, the memory array is divided into `2**R_BITS` rows, `2**C_BITS` columns and `2**B_BITS` banks.

Note: For example, the MT46V32M16 has 512 Mbit = 8M x 4 banks x 16 bit = 32M cells x 16 bit, with 8K rows and 1K columns. Thus, the configuration is:

- $A_BITS = \log_2(32\text{ M}) = 25$
- $D_BITS = 16$
- data-path width of phy on user side: 32-bit because of DDR
- $R_BITS = \log_2(8\text{ K}) = 13$

- $C_BITS = \log_2(1\text{ K}) = 10$
- $B_BITS = \log_2(4) = 2$

Set CAS latency (CL, MR_CL) and burst length (BL, MR_BL) according to your needs.

If you have a DDR-SDRAM then set INIT_DLL = true, otherwise false.

The definition and values of generics T_* can be calculated from the datasheets of the specific SDRAM (e.g. MT46V). Just divide the minimum/maximum times by clock period. Auto refreshs are applied periodically, the datasheet either specifies the average refresh interval (T_REFI) or the total refresh cycle time (T_REF). In the latter case, divide the total time by the row count to get the average refresh interval. Subtract about 50 clock cycles to account for pending read/writes.

INIT_WAIT specifies the time period to wait after the SDRAM is powered up. It is typically 100–200 us long, see datasheet. The waiting time is specified in number of average refresh periods (specified by T_REFI): $INIT_WAIT = \text{ceil}(\text{wait_time} / \text{clock_period} / T_REFI)$ e.g. $INIT_WAIT = \text{ceil}(200\text{ us} / 10\text{ ns} / 700) = 29$

Operation

After user_cmd_valid is asserted high, the command (user_write) and address (user_addr) must be hold until user_got_cmd is asserted.

The FSM automatically waits for user_wdata_valid on writes. The data should be available soon. Otherwise the auto refresh might fail. The FSM only waits for the first word to write. All successive words of a burst must be valid in the following cycles. (A burst can't be stalled.) ATTENTION: During writes, user_cmd_got is asserted only if user_wdata_valid is set.

The write data must directly connected to the physical layer.

Entity Declaration:

```

1  entity sdram_ctrl_fsm is
2      generic (
3          SDRAM_TYPE : natural;           -- SDRAM type
4
5          A_BITS      : positive;         -- log2ceil(memory cell count)
6          D_BITS      : positive;         -- native data width
7          R_BITS      : positive;         -- log2ceil(rows)
8          C_BITS      : positive;         -- log2ceil(columns)
9          B_BITS      : positive;         -- log2ceil(banks)
10
11         CL : positive;  -- CAS Latency in clock cycles
12         BL : positive;  -- Burst Length
13
14         T_MRD      : integer;            -- in clock cycles
15         T_RAS      : integer;            -- in clock cycles
16         T_RCD      : integer;            -- in clock cycles
17         T_RFC      : integer;            -- (or T_RC) in clock cycles
18         T_RP       : integer;            -- in clock cycles
19         T_WR       : integer;            -- in clock cycles
20         T_WTR      : integer;            -- in clock cycles
21         T_REFI     : integer;            -- in clock cycles
22         INIT_WAIT  : integer);           -- in T_REFI periods
23  port (
24      clk : in std_logic;
25      rst : in std_logic;
26
27      user_cmd_valid : in std_logic;
28      user_wdata_valid : in std_logic;

```

(continues on next page)

(continued from previous page)

```

29  user_write      : in  std_logic;
30  user_addr      : in  std_logic_vector(A_BITS-1 downto 0);
31  user_got_cmd   : out std_logic;
32  user_got_wdata : out std_logic;
33
34  sd_cke_nxt : out std_logic;
35  sd_cs_nxt  : out std_logic;
36  sd_ras_nxt : out std_logic;
37  sd_cas_nxt : out std_logic;
38  sd_we_nxt  : out std_logic;
39  sd_a_nxt   : out std_logic_vector(imax(R_BITS,C_BITS+1)-1 downto 0);
40  sd_ba_nxt  : out std_logic_vector(B_BITS-1 downto 0);
41  rden_nxt   : out std_logic;
42  wren_nxt   : out std_logic);
43
44
45  end sdram_ctrl_fsm;

```

Source file: `mem/sdram/sdram_ctrl_fsm.vhdl`

PoC.mem.sdram.ctrl_de0

Complete controller for ISSI SDR-SDRAM for Altera DE0 Board.

SDRAM Device: IS42S16400F

Configuration

Parameter	Description
CLK_PERIOD	Clock period in nano seconds. All SDRAM timings are calculated for the device stated above.
CL	CAS latency, choose according to clock frequency.
BL	Burst length. Choose BL=1 for single cycle memory transactions as required for the PoC.Mem interface.

Tested with: CLK_PERIOD = 7.5 (133 MHz), CL=2, BL=1.

Operation

Command, address and write data is sampled with `clk`. Read data is also aligned with `clk`.

For description on `clkout` see `sdram_ctrl_phy_de0`.

Synchronous resets are used.

Entity Declaration:

```

1  entity sdram_ctrl_de0 is
2
3      generic (
4          CLK_PERIOD : real;
5          CL          : positive;
6          BL          : positive);
7
8      port (
9          clk          : in  std_logic;

```

(continues on next page)

(continued from previous page)

```

10  clkout      : in    std_logic;
11  rst        : in    std_logic;
12
13  user_cmd_valid : in  std_logic;
14  user_wdata_valid : in std_logic;
15  user_write    : in  std_logic;
16  user_addr     : in  std_logic_vector(21 downto 0);
17  user_wdata    : in  std_logic_vector(15 downto 0);
18  user_got_cmd  : out std_logic;
19  user_got_wdata : out std_logic;
20  user_rdata    : out std_logic_vector(15 downto 0);
21  user_rstb     : out std_logic;
22
23  sd_ck        : out  std_logic;
24  sd_cke       : out  std_logic;
25  sd_cs        : out  std_logic;
26  sd_ras       : out  std_logic;
27  sd_cas       : out  std_logic;
28  sd_we        : out  std_logic;
29  sd_ba        : out  std_logic_vector(1 downto 0);
30  sd_a         : out  std_logic_vector(11 downto 0);
31  sd_dq        : inout std_logic_vector(15 downto 0);
32
33  end sdram_ctrl_de0;

```

Source file: `mem/sdram/sdram_ctrl_de0.vhdl`

PoC.mem.sdram.ctrl_phy_de0

Physical layer used by module `sdram_ctrl_de0`.

Instantiates input and output buffer components and adjusts the timing for the Altera DE0 board.

Clock and Reset Signals

Port	Description
clk	Base clock for command and write data path.
rst	Reset for clk.

Command signals and write data are sampled with `clk`. Read data is also aligned with `clk`.

Write and read enable (`wren_nxt`, `rden_nxt`) must be hold for:

- 1 clock cycle if $BL = 1$,
- 2 clock cycles if $BL = 2$, or
- 4 clock cycles if $BL = 4$, or
- 8 clock cycles if $BL = 8$.

They must be first asserted with the read and write command. Proper delay is included in this unit.

The first word to write must be asserted with the write command. Proper delay is included in this unit.

Synchronous resets are used. Reset must be hold for at least two cycles.

Entity Declaration:

```
1 entity sdram_ctrl_phy_de0 is
2   generic (
3     CL : positive);           -- CAS latency
4   port (
5     clk      : in std_logic;
6     clkout   : in std_logic;
7     rst      : in std_logic;
8
9     sd_cke_nxt : in std_logic;
10    sd_cs_nxt  : in std_logic;
11    sd_ras_nxt : in std_logic;
12    sd_cas_nxt : in std_logic;
13    sd_we_nxt  : in std_logic;
14    sd_ba_nxt  : in std_logic_vector(1 downto 0);
15    sd_a_nxt   : in std_logic_vector(11 downto 0);
16
17    wren_nxt   : in std_logic;
18    wdata_nxt  : in std_logic_vector(15 downto 0);
19
20    rden_nxt   : in std_logic;
21    rdata      : out std_logic_vector(15 downto 0);
22    rstb       : out std_logic;
23
24    sd_ck      : out std_logic;
25    sd_cke     : out std_logic;
26    sd_cs      : out std_logic;
27    sd_ras     : out std_logic;
28    sd_cas     : out std_logic;
29    sd_we      : out std_logic;
30    sd_ba      : out std_logic_vector(1 downto 0);
31    sd_a       : out std_logic_vector(11 downto 0);
32    sd_dq      : inout std_logic_vector(15 downto 0));
33
34 end sdram_ctrl_phy_de0;
```

Source file: mem/sdram/sdram_ctrl_phy_de0.vhdl

PoC.mem.sdram.ctrl_s3esk

Controller for Micron DDR-SDRAM on Spartan-3E Starter Kit Board.

SDRAM Device: MT46V32M16-6T

Configuration

Parameter	Description
CLK_PERIOD	Clock period in nano seconds. All SDRAM timings are calculated for the device stated above.
CL	CAS latency, choose according to clock frequency.
BL	Burst length. Choose BL=2 for single cycle memory transactions as required for the PoC.Mem interface.

Tested with: CLK_PERIOD = 10.0, CL=2, BL=2.

Operation

Command, address and write data are sampled with the rising edge of `clk`.

Read data is aligned with `clk_fb90_n`. Either process data in this clock domain, or connect a FIFO to transfer data into another clock domain of your choice. This FIFO should be capable of storing at least one burst (size $BL/2$) + start of next burst (size 1).

Synchronous resets are used.

Entity Declaration:

```

1  entity sdram_ctrl_s3esk is
2
3      generic (
4          CLK_PERIOD : real;
5          BL          : positive);
6
7      port (
8          clk          : in    std_logic;
9          clk_n        : in    std_logic;
10         clk90         : in    std_logic;
11         clk90_n      : in    std_logic;
12         rst          : in    std_logic;
13         rst90        : in    std_logic;
14         rst180       : in    std_logic;
15         rst270       : in    std_logic;
16         clk_fb90     : in    std_logic;
17         clk_fb90_n   : in    std_logic;
18         rst_fb90     : in    std_logic;
19         rst_fb270    : in    std_logic;
20
21         user_cmd_valid : in    std_logic;
22         user_wdata_valid : in    std_logic;
23         user_write     : in    std_logic;
24         user_addr      : in    std_logic_vector(24 downto 0);
25         user_wdata     : in    std_logic_vector(31 downto 0);
26         user_got_cmd   : out   std_logic;
27         user_got_wdata : out   std_logic;
28         user_rdata     : out   std_logic_vector(31 downto 0);
29         user_rstb      : out   std_logic;
30
31         sd_ck_p        : out   std_logic;
32         sd_ck_n        : out   std_logic;
33         sd_cke         : out   std_logic;
34         sd_cs          : out   std_logic;
35         sd_ras         : out   std_logic;
36         sd_cas         : out   std_logic;
37         sd_we          : out   std_logic;
38         sd_ba          : out   std_logic_vector(1 downto 0);
39         sd_a           : out   std_logic_vector(12 downto 0);
40         sd_ldqs        : out   std_logic;
41         sd_udqs        : out   std_logic;
42         sd_dq          : inout  std_logic_vector(15 downto 0));
43
44  end sdram_ctrl_s3esk;
```

Source file: `mem/sdram/sdram_ctrl_s3esk.vhdl`

PoC.mem.sdram.ctrl_phy_s3esk

Physical layer used by module *sdram_ctrl_s3esk*.

Instantiates input and output buffer components and adjusts the timing for the Spartan-3E Starter Kit Board.

Clock and Reset Signals

Port	Description
clk	Base clock for command and write data path.
clk_n	clk phase shifted by 180 degrees.
clk90	clk phase shifted by 90 degrees.
clk90_n	clk phase shifted by 270 degrees.
clk_fb (on PCB)	Driven by external feedback (sd_ck_fb) of DDR-SDRAM clock (sd_ck_p). Actually unused, just referenced below.
clk_fb90	clk_fb phase shifted by 90 degrees.
clk_fb90_n	clk_fb phase shifted by 270 degrees.
rst	Reset for clk.
rst180	Reset for clk_n
rst90	Reset for clk90.
rst270	Reset for clk270.
rst_fb90	Reset for clk_fb90.
rst_fb90_n	Reset for clk_fb90_n.

Operation

Command signals and write data are sampled with the rising edge of clk.

Read data is aligned with clk_fb90_n. Either process data in this clock domain, or connect a FIFO to transfer data into another clock domain of your choice. This FIFO should capable of storing at least one burst (size BL/2) + start of next burst (size 1).

Write and read enable (wren_nxt, rden_nxt) must be hold for:

- 1 clock cycle if BL = 2,
- 2 clock cycles if BL = 4, or
- 4 clock cycles if BL = 8.

They must be first asserted with the read and write command. Proper delay is included in this unit.

The first word to write must be asserted with the write command. Proper delay is included in this unit.

The SDRAM clock is regenerated in this module. The following timing is chosen for minimum latency (should work up to 100 MHz):

- `rising_edge(clk90)` triggers `rising_edge(sd_ck_p)`,
- `rising_edge(clk90_n)` triggers `falling_edge(sd_ck_p)`.

XST options: Disable equivalent register removal.

Synchronous resets are used. Reset must be hold for at least two cycles.

Entity Declaration:

```

1  entity sdram_ctrl_phy_s3esk is
2      port (
3          clk      : in std_logic;
4          clk_n    : in std_logic;
5          clk90    : in std_logic;
6          clk90_n  : in std_logic;
7          rst      : in std_logic;
8          rst90    : in std_logic;
9          rst180   : in std_logic;
10         rst270   : in std_logic;
11
12         clk_fb90  : in std_logic;
13         clk_fb90_n : in std_logic;
14         rst_fb90  : in std_logic;
15         rst_fb270 : in std_logic;
16
17         sd_cke_nxt : in std_logic;
18         sd_cs_nxt  : in std_logic;
19         sd_ras_nxt : in std_logic;
20         sd_cas_nxt : in std_logic;
21         sd_we_nxt  : in std_logic;
22         sd_ba_nxt  : in std_logic_vector(1 downto 0);
23         sd_a_nxt   : in std_logic_vector(12 downto 0);
24
25         wren_nxt   : in std_logic;
26         wdata_nxt  : in std_logic_vector(31 downto 0);
27
28         rden_nxt   : in std_logic;
29         rdata      : out std_logic_vector(31 downto 0);
30         rstb       : out std_logic;
31
32         sd_ck_p    : out std_logic;
33         sd_ck_n    : out std_logic;
34         sd_cke     : out std_logic;
35         sd_cs      : out std_logic;
36         sd_ras     : out std_logic;
37         sd_cas     : out std_logic;
38         sd_we      : out std_logic;
39         sd_ba      : out std_logic_vector(1 downto 0);
40         sd_a       : out std_logic_vector(12 downto 0);
41         sd_ldqs    : out std_logic;
42         sd_udqs    : out std_logic;
43         sd_dq      : inout std_logic_vector(15 downto 0));
44
45 end sdram_ctrl_phy_s3esk;

```

Source file: mem/sdram/sdram_ctrl_phy_s3esk.vhdl

7.12 PoC.misc

The namespace `PoC.misc` offers different yet uncategorized entities.

Sub-Namespace

- *PoC.misc.filter* contains 1-bit filter algorithms.
- *PoC.misc.stat* contains statistic modules.
- *PoC.misc.sync* offers clock-domain-crossing (CDC) modules.

Package

The package *PoC.misc* holds all component declarations for this namespace.

Entities

- *PoC.misc.Delay*
- *PoC.misc.FrequencyMeasurement*
- *PoC.misc.PulseTrain*
- *PoC.misc.Sequencer*
- *PoC.misc.StrobeGenerator*
- *PoC.misc.StrobeLimiter*
- IP:misc_WordAligner

7.12.1 PoC.misc.filter

These are filter entities. . . .

Entities

- *PoC.misc.filter.and*
- *PoC.misc.filter.mean*
- *PoC.misc.filter.or*

PoC.misc.filter.and

Todo: No documentation available.

Entity Declaration:

```

1  entity filter_and is
2      generic (
3          TAPS           : positive      := 4;          --
4          INIT           : std_logic     := '0';        --
5          ADD_OUTPUT_REG : boolean       := FALSE      --
6      );
7      port (
8          Clock           : in  std_logic;              -- clock
9          DataIn          : in  std_logic;              -- data to filter
10         DataOut         : out std_logic               -- filtered signal
11     );
12 end entity;
```

Source file: misc/filter/filter_and.vhdl

PoC.misc.filter.mean

Todo: No documentation available.

Entity Declaration:

```

1 entity filter_mean is
2   generic (
3     TAPS          : positive      := 4;      --
4     INIT          : std_logic     := '1';    --
5     ADD_OUTPUT_REG : boolean      := FALSE   --
6   );
7   port (
8     Clock         : in  std_logic;    -- clock
9     DataIn        : in  std_logic;    -- data to filter
10    DataOut       : out std_logic     -- filtered signal
11  );
12 end entity;
```

Source file: `misc/filter/filter_mean.vhdl`

PoC.misc.filter.or

Todo: No documentation available.

Entity Declaration:

```

1 entity filter_or is
2   generic (
3     TAPS          : positive      := 4;      --
4     INIT          : std_logic     := '1';    --
5     ADD_OUTPUT_REG : boolean      := FALSE   --
6   );
7   port (
8     Clock         : in  std_logic;    -- clock
9     DataIn        : in  std_logic;    -- data to filter
10    DataOut       : out std_logic     -- filtered signal
11  );
12 end entity;
```

Source file: `misc/filter/filter_or.vhdl`

7.12.2 PoC.misc.gearbox

These are gearbox entities. . .

Entities

- *PoC.misc.gearbox.down_cc*
- *PoC.misc.gearbox.down_dc*
- *PoC.misc.gearbox.up_cc*
- *PoC.misc.gearbox.up_dc*

PoC.misc.gearbox.down_cc

This module provides a downscaling gearbox with a common clock (cc) interface. It performs a ‘word’ to ‘byte’ splitting. The default order is LITTLE_ENDIAN (starting at byte(0)). Input “In_Data” and output

“Out_Data” are of the same clock domain “Clock”. Optional input and output registers can be added by enabling (ADD_***PUT_REGISTERS = TRUE).

Entity Declaration:

```

1 entity gearbox_down_cc is
2   generic (
3     INPUT_BITS          : positive := 32;
4     OUTPUT_BITS         : positive := 24;
5     META_BITS           : natural  := 0;
6     ADD_INPUT_REGISTERS : boolean  := FALSE;
7     ADD_OUTPUT_REGISTERS : boolean  := FALSE
8   );
9   port (
10    Clock      : in  std_logic;
11
12    In_Sync     : in  std_logic;
13    In_Valid    : in  std_logic;
14    In_Next     : out std_logic;
15    In_Data     : in  std_logic_vector (INPUT_BITS - 1 downto 0);
16    In_Meta     : in  std_logic_vector (META_BITS - 1 downto 0);
17
18    Out_Sync    : out std_logic;
19    Out_Valid   : out std_logic;
20    Out_Data    : out std_logic_vector (OUTPUT_BITS - 1 downto 0);
21    Out_Meta    : out std_logic_vector (META_BITS - 1 downto 0);
22    Out_First   : out std_logic;
23    Out_Last    : out std_logic
24  );
25 end entity;

```

Source file: misc/gearbox/gearbox_down_cc.vhdl

PoC.misc.gearbox.down_dc

This module provides a downscaling gearbox with a dependent clock (dc) interface. It performs a ‘word’ to ‘byte’ splitting. The default order is LITTLE_ENDIAN (starting at byte(0)). Input “In_Data” is of clock domain “Clock1”; output “Out_Data” is of clock domain “Clock2”. Optional input and output registers can be added by enabling (ADD_***PUT_REGISTERS = TRUE).

Assertions:

- Clock periods of Clock1 and Clock2 MUST be multiples of each other.
- Clock1 and Clock2 MUST be phase aligned (related) to each other.

Entity Declaration:

```

1 entity gearbox_down_dc is
2   generic (
3     INPUT_BITS          : positive := 32;
4     OUTPUT_BITS         : positive := 8;
5     OUTPUT_ORDER        : T_BIT_ORDER := LSB_FIRST;
6     FIRST: start at byte(0), MSB_FIRST: start at byte(n-1)
7   );
8   port (
9     In_Data      : in  std_logic_vector (INPUT_BITS - 1 downto 0);
10    In_Meta      : in  std_logic_vector (META_BITS - 1 downto 0);
11    In_Valid      : in  std_logic;
12    In_Sync       : in  std_logic;
13    Out_Data      : out std_logic_vector (OUTPUT_BITS - 1 downto 0);
14    Out_Meta      : out std_logic_vector (META_BITS - 1 downto 0);
15    Out_Valid     : out std_logic;
16    Out_Sync      : out std_logic;
17    Out_First     : out std_logic;
18    Out_Last      : out std_logic
19  );
20 end entity;

```

(continues on next page)

(continued from previous page)

```

6   ADD_INPUT_REGISTERS    : boolean           := FALSE;           -- add_
   ↪input register @Clock1
7   ADD_OUTPUT_REGISTERS   : boolean           := FALSE           -- add_
   ↪output register @Clock2
8   );
9   port (
10    Clock1                : in  std_logic;           --_
   ↪input clock domain
11    Clock2                : in  std_logic;           --_
   ↪output clock domain
12    In_Data               : in  std_logic_vector(INPUT_BITS - 1 downto 0); --_
   ↪input word
13    Out_Data              : out std_logic_vector(OUTPUT_BITS - 1 downto 0) --_
   ↪output word
14  );
15 end entity;

```

Source file: `misc/gearbox/gearbox_down_dc.vhdl`

PoC.misc.gearbox.up_cc

This module provides a downscaling gearbox with a common clock (cc) interface. It performs a ‘byte’ to ‘word’ collection. The default order is LITTLE_ENDIAN (starting at byte(0)). Input “In_Data” and output “Out_Data” are of the same clock domain “Clock”. Optional input and output registers can be added by enabling (ADD_***PUT_REGISTERS = TRUE).

Entity Declaration:

```

1 entity gearbox_up_cc is
2   generic (
3     INPUT_BITS            : positive := 24;
4     OUTPUT_BITS           : positive := 32;
5     META_BITS             : natural  := 0;
6     ADD_INPUT_REGISTERS   : boolean   := FALSE;
7     ADD_OUTPUT_REGISTERS  : boolean   := FALSE
8   );
9   port (
10    Clock                 : in  std_logic;
11
12    In_Sync               : in  std_logic;
13    In_Valid              : in  std_logic;
14    In_Data               : in  std_logic_vector(INPUT_BITS - 1 downto 0);
15    In_Meta               : in  std_logic_vector(META_BITS - 1 downto 0);
16
17    Out_Sync               : out std_logic;
18    Out_Valid              : out std_logic;
19    Out_Data               : out std_logic_vector(OUTPUT_BITS - 1 downto 0);
20    Out_Meta               : out std_logic_vector(META_BITS - 1 downto 0);
21    Out_First              : out std_logic;
22    Out_Last               : out std_logic
23  );
24 end entity;

```

Source file: `misc/gearbox/gearbox_up_cc.vhdl`

PoC.misc.gearbox.up_dc

This module provides a upscaling gearbox with a dependent clock (dc) interface. It performs a ‘byte’ to ‘word’ collection. The default order is LITTLE_ENDIAN (starting at byte(0)). Input “In_Data” is of clock domain “Clock1”; output “Out_Data” is of clock domain “Clock2”. The “In_Align” is required to mark the starting byte in the word. An optional input register can be added by enabling (ADD_INPUT_REGISTERS = TRUE).

Assertions:

- Clock periods of Clock1 and Clock2 MUST be multiples of each other.
- Clock1 and Clock2 MUST be phase aligned (related) to each other.

Entity Declaration:

```

1 entity gearbox_up_dc is
2   generic (
3     INPUT_BITS          : positive      := 8;           --
    ↪input bit width
4     INPUT_ORDER         : T_BIT_ORDER   := LSB_FIRST;   -- LSB_
    ↪FIRST: start at byte(0), MSB_FIRST: start at byte(n-1)
5     OUTPUT_BITS         : positive      := 32;          --
    ↪output bit width
6     ADD_INPUT_REGISTERS : boolean        := FALSE       -- add_
    ↪input register @Clock1
7   );
8   port (
9     Clock1              : in  std_logic;               --
    ↪input clock domain
10    Clock2               : in  std_logic;               --
    ↪output clock domain
11    In_Align             : in  std_logic;               --
    ↪align word (one cycle high impulse)
12    In_Data              : in  std_logic_vector(INPUT_BITS - 1 downto 0); --
    ↪input word
13    Out_Data             : out std_logic_vector(OUTPUT_BITS - 1 downto 0); --
    ↪output word
14    Out_Valid            : out std_logic                --
    ↪output is valid
15  );
16 end entity;
```

Source file: [misc/gearbox/gearbox_up_dc.vhdl](#)

7.12.3 PoC.misc.stat

These are stat entities...

Entities

- *PoC.misc.stat.Average*
- *PoC.misc.stat.Histogram*
- *PoC.misc.stat.Maximum*
- *PoC.misc.stat.Minimum*

PoC.misc.stat.Average

Todo: No documentation available.

Entity Declaration:

```

1  entity stat_Average is
2      generic (
3          DATA_BITS      : positive      := 8;
4          COUNTER_BITS    : positive      := 16
5      );
6      port (
7          Clock            : in  std_logic;
8          Reset            : in  std_logic;
9
10         Enable           : in  std_logic;
11         DataIn           : in  std_logic_vector(DATA_BITS - 1 downto 0);
12
13         Count            : out std_logic_vector(COUNTER_BITS - 1 downto 0);
14         Sum              : out std_logic_vector(COUNTER_BITS - 1 downto 0);
15         Average          : out std_logic_vector(COUNTER_BITS - 1 downto 0);
16         Valid            : out std_logic
17     );
18 end entity;
```

Source file: [misc/stat/stat_Average.vhdl](#)

PoC.misc.stat.Histogram

Todo: No documentation available.

Entity Declaration:

```

1  entity stat_Histogram is
2      generic (
3          DATA_BITS      : positive      := 16;
4          COUNTER_BITS    : positive      := 16
5      );
6      port (
7          Clock            : in  std_logic;
8          Reset            : in  std_logic;
9
10         Enable           : in  std_logic;
11         DataIn           : in  std_logic_vector(DATA_BITS - 1 downto 0);
12
13         Histogram        : out T_SLM(2*DATA_BITS - 1 downto 0, COUNTER_BITS - 1 downto 0)
14     );
15 end entity;
```

Source file: [misc/stat/stat_Histogram.vhdl](#)

PoC.misc.stat.Maximum

Todo: No documentation available.

Entity Declaration:

```
1 entity stat_Maximum is
2   generic (
3     DEPTH          : positive      := 8;
4     DATA_BITS     : positive      := 16;
5     COUNTER_BITS   : positive      := 16
6   );
7   port (
8     Clock          : in  std_logic;
9     Reset          : in  std_logic;
10
11     Enable         : in  std_logic;
12     DataIn         : in  std_logic_vector(DATA_BITS - 1 downto 0);
13
14     Valid          : out std_logic_vector(DEPTH - 1 downto 0);
15     Maximums       : out T_SLM(DEPTH - 1 downto 0, DATA_BITS - 1 downto 0);
16     Counts         : out T_SLM(DEPTH - 1 downto 0, COUNTER_BITS - 1 downto 0)
17   );
18 end entity;
```

Source file: [misc/stat/stat_Maximum.vhdl](#)

PoC.misc.stat.Minimum

Todo: No documentation available.

Entity Declaration:

```
1 entity stat_Minimum is
2   generic (
3     DEPTH          : positive      := 8;
4     DATA_BITS     : positive      := 16;
5     COUNTER_BITS   : positive      := 16
6   );
7   port (
8     Clock          : in  std_logic;
9     Reset          : in  std_logic;
10
11     Enable         : in  std_logic;
12     DataIn         : in  std_logic_vector(DATA_BITS - 1 downto 0);
13
14     Valid          : out std_logic_vector(DEPTH - 1 downto 0);
15     Minimums       : out T_SLM(DEPTH - 1 downto 0, DATA_BITS - 1 downto 0);
16     Counts         : out T_SLM(DEPTH - 1 downto 0, COUNTER_BITS - 1 downto 0)
17   );
18 end entity;
```

Source file: [misc/stat/stat_Minimum.vhdl](#)

7.12.4 PoC.misc.sync

The namespace `PoC.misc.sync` offers different clock-domain-crossing (CDC) synchronizer circuits. All synchronizers are based on the basic 2 flip-flop synchronizer called *sync_Bits*. PoC has two platform specific implementations for Altera and Xilinx, which are choosen, if the appropriate `MY_DEVICE` constant is configured in `my_config.vhdl`.

Decision Table:

Behavior	Flag ¹	Strobe ²	Continuous Data	Reset ⁴	Pulse ³
1 Bit	<i>sync_Bits</i>	<i>sync_Strobe</i>	<i>fifo_ic_got</i> ⁵	<i>sync_Reset</i>	<i>sync_Pulse</i>
n Bit	<i>sync_Vector</i>	<i>sync_Command</i>	<i>fifo_ic_got</i> ⁵		

Basic 2 Flip-Flop Synchronizer

The basic 2 flip-flop synchronizer is called *sync_Bits*. It's possible to configure the bit count of indivital bits. If a vector shall be synchronized, use one of the special synchronizers like *sync_Vector*. The vendor specific implementations are named *sync_Bits_Altera* and *sync_Bits_Xilinx* respectively.

A second variant of the 2-FF synchronizer is called *sync_Reset*. It's for Reset-signals, implementing asynchronous assertion and synchronous deassertion. The vendor specific implementations are named *sync_Reset_Altera* and *sync_Reset_Xilinx* respectively.

A third variant of a 2-FF synchronizer is called *sync_Pulse*. It's for very short Pulsed-signals. It uses an addition asynchronous capture FF to latch the very short pulse. The vendor specific implementations are named *sync_Pulse_Altera* and *sync_Pulse_Xilinx* respectively.

Special Synchronizers

Based on the 2-FF synchronizer, several "high-level" synchronizers are build.

- *sync_Strobe* synchronizer *strobe*-signals across clock-domain-boundaries. A busy signal indicates the synchronization status and can be used as a internal gate-signal to disallow new incoming strobes. A *strobe*-signal is only for one clock period active.
- *sync_Command* like *sync_Strobe*, it synchronizes a one clock period active signal across the clock-domain-boundary, but the input has multiple bits. After the multi bit strobe (Command) was transfered, the output goes to its idle value.
- *sync_Vector* synchronizes a complete vector across the clock-domain-boundary. A changed detection on the input vector causes a register to latch the current state. The changed event is transfered to the new clock-domain and triggers a register to store the latched content, but in the new clock domain.

See also:

PoC.fifo.ic_got For a cross-clock capable FIFO.

PoC.misc.sync Package

Source file: *sync.pkg.vhdl*

¹ A *flag* or *status* signal is a continuous, long time stable signal.

² A *strobe* signal is active for only one cycle.

⁴ To be refumented

³ A *pulse* signal is a very short event.

⁵ See the `PoC.fifo` namespace for cross-clock capable FIFOs.

PoC.misc.sync.Bits

This module synchronizes multiple flag bits into clock-domain `ClOck`. The clock-domain boundary crossing is done by two synchronizer D-FFs. All bits are independent from each other. If a known vendor like Altera or Xilinx are recognized, a vendor specific implementation is chosen.

Attention: Use this synchronizer only for long time stable signals (flags).

Constraints:

General: Please add constraints for meta stability to all ‘_meta’ signals and timing ignore constraints to all ‘_async’ signals.

Xilinx: In case of a Xilinx device, this module will instantiate the optimized module `PoC.xil.sync.Bits`. Please attend to the notes of `sync_Bits.vhdl`.

Altera sdc file: TODO

Entity Declaration:

```

1 entity sync_Bits is
2   generic (
3     BITS          : positive          := 1;           -- number of_
↳ bit to be synchronized
4     INIT          : std_logic_vector  := x"00000000"; -- _
↳ initialization bits
5     SYNC_DEPTH    : T_MISC_SYNC_DEPTH := T_MISC_SYNC_DEPTH'low -- generate_
↳ SYNC_DEPTH many stages, at least 2
6   );
7   port (
8     Clock         : in  std_logic;      -- <Clock> _
↳ output clock domain
9     Input         : in  std_logic_vector(BITS - 1 downto 0); -- @async: _
↳ input bits
10    Output        : out std_logic_vector(BITS - 1 downto 0) -- @Clock: _
↳ output bits
11  );
12 end entity;
```

See also:

PoC.misc.sync.Reset For a special 2 D-FF synchronizer for *reset*-signals.

PoC.misc.sync.Pulse For a special 1+2 D-FF synchronizer for *pulse*-signals.

PoC.misc.sync.Strobe For a synchronizer for *strobe*-signals.

PoC.misc.sync.Vector For a multiple bits capable synchronizer.

Source file: `misc/sync/sync_Bits.vhdl`

PoC.misc.sync.Command

This module synchronizes a vector of bits from clock-domain `ClOck1` to clock-domain `ClOck2`. The clock-domain boundary crossing is done by a change comparator, a T-FF, two synchronizer D-FFs and a reconstructive XOR indicating a value change on the input. This changed signal is used to capture the input for the new output. A busy flag is additionally calculated for the input clock-domain. The output has strobe character and is reset to it's `INIT` value after one clock cycle.

Constraints: This module uses sub modules which need to be constrained. Please attend to the notes of the instantiated sub modules.

Entity Declaration:

```

1 entity sync_Command is
2   generic (
3     BITS          : positive          := 8;           -- number of_
↳bit to be synchronized
4     INIT          : std_logic_vector  := x"00000000";  --
5     SYNC_DEPTH    : T_MISC_SYNC_DEPTH := T_MISC_SYNC_DEPTH'low -- generate_
↳SYNC_DEPTH many stages, at least 2
6   );
7   port (
8     Clock1        : in  std_logic;      -- <Clock> _
↳input clock
9     Clock2        : in  std_logic;      -- <Clock> _
↳output clock
10    Input          : in  std_logic_vector(BITS - 1 downto 0); -- @Clock1:_
↳input vector
11    Output         : out std_logic_vector(BITS - 1 downto 0); -- @Clock2:_
↳output vector
12    Busy           : out std_logic;      -- @Clock1:_
↳busy bit
13    Changed        : out std_logic      -- @Clock2:_
↳changed bit
14  );
15 end entity;
```

Source file: misc/sync/sync_Command.vhdl

PoC.misc.sync.Pulse

This module synchronizes multiple pulsed bits into the clock-domain `Clock`. The clock-domain boundary crossing is done by two synchronizer D-FFs. All bits are independent from each other. If a known vendor like Altera or Xilinx are recognized, a vendor specific implementation is chosen.

Attention: Use this synchronizer for very short signals (pulse).

Constraints:

General: Please add constraints for meta stability to all ‘_meta’ signals and timing ignore constraints to all ‘_async’ signals.

Xilinx: In case of a Xilinx device, this module will instantiate the optimized module `PoC.xil.sync.Pulse`. Please attend to the notes of `sync_Bits.vhdl`.

Altera sdc file: TODO

Entity Declaration:

```

1 entity sync_Pulse is
2   generic (
3     BITS          : positive          := 1;           -- number of_
↳bit to be synchronized
4     SYNC_DEPTH    : T_MISC_SYNC_DEPTH := T_MISC_SYNC_DEPTH'low -- generate_
↳SYNC_DEPTH many stages, at least 2
5   );
6   port (
7     Clock         : in  std_logic;      -- <Clock> _
↳output clock domain
```

(continues on next page)

(continued from previous page)

```

8   Input      : in std_logic_vector (BITS - 1 downto 0);      -- @async: ␣
↪input bits
9   Output     : out std_logic_vector (BITS - 1 downto 0)      -- @Clock: ␣
↪output bits
10  );
11  end entity;
```

See also:*PoC.misc.sync.Bits* For a common 2 D-FF synchronizer for *flag*-signals.*PoC.misc.sync.Reset* For a special 2 D-FF synchronizer for *reset*-signals.*PoC.misc.sync.Strobe* For a synchronizer for *strobe*-signals.*PoC.misc.sync.Vector* For a multiple bits capable synchronizer.Source file: `misc/sync/sync_Pulse.vhdl`**PoC.misc.sync.Reset**

This module synchronizes an asynchronous reset signal to the clock `Clock`. The `Input` can be asserted and de-asserted at any time. The `Output` is asserted asynchronously and de-asserted synchronously to the clock.

Attention: Use this synchronizer only to asynchronously reset your design. The ‘Output’ should be feed by global buffer to the destination FFs, so that, it reaches their reset inputs within one clock cycle.

Constraints:

General: Please add constraints for meta stability to all ‘_meta’ signals and timing ignore constraints to all ‘_async’ signals.

Xilinx: In case of a Xilinx device, this module will instantiate the optimized module `xil_SyncReset`. Please attend to the notes of `xil_SyncReset`.

Altera sdc file: TODO

Entity Declaration:

```

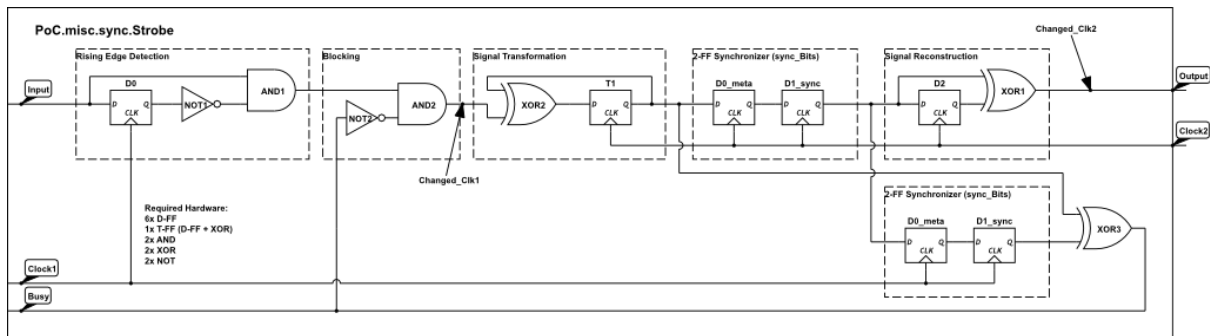
1  entity sync_Reset is
2    generic (
3      SYNC_DEPTH      : T_MISC_SYNC_DEPTH      := T_MISC_SYNC_DEPTH'low      -- generate_
↪SYNC_DEPTH many stages, at least 2
4    );
5    port (
6      Clock           : in std_logic;          -- <Clock> ␣
↪output clock domain
7      Input           : in std_logic;          -- @async: ␣
↪reset input
8      Output          : out std_logic          -- @Clock: ␣
↪reset output
9    );
10  end entity;
```

Source file: `misc/sync/sync_Reset.vhdl`

PoC.misc.sync.Strobe

This module synchronizes multiple high-active bits from clock-domain `Clock1` to clock-domain `Clock2`. The clock-domain boundary crossing is done by a T-FF, two synchronizer D-FFs and a reconstructive XOR. A busy flag is additionally calculated and can be used to block new inputs. All bits are independent from each other. Multiple consecutive strobes are suppressed by a rising edge detection.

Attention: Use this synchronizer only for one-cycle high-active signals (strobes).



Constraints: This module uses sub modules which need to be constrained. Please attend to the notes of the instantiated sub modules.

Entity Declaration:

```

1  entity sync_Strobe is
2      generic (
3          BITS                : positive                := 1;
4          ↪number of bit to be synchronized
5          GATED_INPUT_BY_BUSY : boolean                 := TRUE;
6          ↪gated input (by busy signal)
7          SYNC_DEPTH          : T_MISC_SYNC_DEPTH      := T_MISC_SYNC_DEPTH'low
8          ↪generate SYNC_DEPTH many stages, at least 2
9      );
10     port (
11         Clock1                : in  std_logic;
12         ↪input clock domain
13         Clock2                : in  std_logic;
14         ↪output clock domain
15         Input                 : in  std_logic_vector(BITS - 1 downto 0);
16         ↪input bits
17         Output                : out std_logic_vector(BITS - 1 downto 0);
18         ↪output bits
19         Busy                  : out std_logic_vector(BITS - 1 downto 0);
20         ↪busy bits
21     );
22 end entity;

```

Source file: `misc/sync/sync_Strobe.vhdl`

PoC.misc.sync.Vector

This module synchronizes a vector of bits from clock-domain `Clock1` to clock-domain `Clock2`. The clock-domain boundary crossing is done by a change comparator, a T-FF, two synchronizer D-FFs and a reconstructive XOR indicating a value change on the input. This changed signal is used to capture the input for the new output. A busy flag is additionally calculated for the input clock domain.

Constraints: This module uses sub modules which need to be constrained. Please attend to the notes of the instantiated sub modules.

Entity Declaration:

```

1 entity sync_Vector is
2   generic (
3     MASTER_BITS    : positive          := 8;           -- number of_
↳ bit to be synchronized
4     SLAVE_BITS     : natural           := 0;
5     INIT           : std_logic_vector := x"00000000";   --
6     SYNC_DEPTH    : T_MISC_SYNC_DEPTH := T_MISC_SYNC_DEPTH'low -- generate_
↳ SYNC_DEPTH many stages, at least 2
7   );
8   port (
9     Clock1         : in std_logic;
↳ -- <Clock> input clock
10    Clock2         : in std_logic;
↳ -- <Clock> output clock
11    Input          : in std_logic_vector (MASTER_BITS + SLAVE_BITS) - 1 downto 0);
↳ -- @Clock1: input vector
12    Output         : out std_logic_vector (MASTER_BITS + SLAVE_BITS) - 1 downto 0);
↳ -- @Clock2: output vector
13    Busy           : out std_logic;
↳ -- @Clock1: busy bit
14    Changed        : out std_logic
↳ -- @Clock2: changed bit
15  );
16 end entity;
```

Source file: misc/sync/sync_Vector.vhdl

7.12.5 PoC.misc Package

This package holds all component declarations for this namespace.

Source file: misc.pkg.vhdl

7.12.6 PoC.misc.Delay

Todo: No documentation available.

Entity Declaration:

```

1 entity misc_Delay is
2   generic (
3     BITS           : positive;
4     TAPS           : T_NATIVEC        -- select one or multiple delay tap points
5   );
6   port (
7     Clock          : in std_logic;
↳ clock
8     Reset          : in std_logic := '0';
↳ reset; avoid reset to enable SRL16/SRL32 usage
```

(continues on next page)

(continued from previous page)

```

9      Enable      : in std_logic := '1';
↳enable
10     DataIn      : in std_logic_vector (BITS - 1 downto 0);
↳data to delay
11     DataOut     : out T_SLM(TAPS'length - 1 downto 0, BITS - 1 downto 0)
↳delayed ouputs, tapped at TAPS(i)
12     );
13 end entity;
```

Source file: misc/misc_Delay.vhdl

7.12.7 PoC.misc.FrequencyMeasurement

This module counts 1 second in a reference timer at reference clock. This reference time is used to start and stop a timer at input clock. The counter value is the measured frequency in Hz.

Entity Declaration:

```

1 entity misc_FrequencyMeasurement is
2   generic (
3     REFERENCE_CLOCK_FREQ : FREQ      := 100 MHz
4   );
5   port (
6     Reference_Clock : in std_logic;
7     Input_Clock    : in std_logic;
8
9     Start          : in std_logic;
10    Done            : out std_logic;
11    Result          : out T_SLV_32
12  );
13 end entity;
```

Source file: misc/misc_FrequencyMeasurement.vhdl

7.12.8 PoC.misc.PulseTrain

This module generates pulse trains. This module was written as a answer for a StackOverflow question: <http://stackoverflow.com/questions/25783320>

Entity Declaration:

Source file: misc/misc_PulseTrain.vhdl

7.12.9 PoC.misc.Sequencer

Todo: No documentation available.

Entity Declaration:

Source file: misc/misc_Sequencer.vhdl

7.12.10 PoC.misc.StrobeGenerator

Todo: No documentation available.

Entity Declaration:

Source file: `misc/misc_StrobeGenerator.vhdl`

7.12.11 PoC.misc.StrobeLimiter

Todo: No documentation available.

Entity Declaration:

Source file: `misc/misc_StrobeLimiter.vhdl`

7.12.12 WordAligner

Todo: No documentation available.

Entity Declaration:

Source file: `misc/misc_WordAligner.vhdl`

7.13 PoC.net

These are bus entities...

Sub-Namespaces

- *PoC.net.arp*
- *PoC.net.eth*
- *PoC.net.icmpv4*
- *PoC.net.icmpv6*
- *PoC.net.ipv4*
- *PoC.net.ipv6*
- *PoC.net.mac*
- *PoC.net.ndp*
- *PoC.net.stack*
- *PoC.net.udp*

Entities

- *PoC.net.FrameChecksum*
- IP:net_FrameLoopback

7.13.1 PoC.net.arp

These are ARP entities....

PoC.net.arp.Broadcast_Receiver

Todo: No documentation available.

Entity Declaration:

```

1  entity arp_BroadCast_Receiver is
2      generic (
3          ALLOWED_PROTOCOL_IPV4      : boolean           := TRUE;
4          ALLOWED_PROTOCOL_IPV6      : boolean           := FALSE;
5      );
6      port (
7          Clock                       : in  std_logic;
8          Reset                       : in  std_logic;
9
10         RX_Valid                    : in  std_logic;
11         RX_Data                     : in  T_SLV_8;
12         RX_SOF                      : in  std_logic;
13         RX_EOF                      : in  std_logic;
14         RX_Ack                      : out std_logic;
15         RX_Meta_rst                 : out std_logic;
16         RX_Meta_SrcMACAddress_nxt   : out std_logic;
17         RX_Meta_SrcMACAddress_Data  : in  T_SLV_8;
18         RX_Meta_DestMACAddress_nxt  : out std_logic;
19         RX_Meta_DestMACAddress_Data : in  T_SLV_8;
20
21         Clear                       : in  std_logic;
22         Error                       : out std_logic;
23
24         RequestReceived              : out std_logic;
25         Address_rst                  : in  std_logic;
26         SenderMACAddress_nxt         : in  std_logic;
27         SenderMACAddress_Data        : out T_SLV_8;
28         SenderIPAddress_nxt          : in  std_logic;
29         SenderIPAddress_Data         : out T_SLV_8;
30         TargetIPAddress_nxt          : in  std_logic;
31         TargetIPAddress_Data         : out T_SLV_8;
32     );
33 end entity;
```

Source file: net/arp/arp_BroadCast_Receiver.vhdl

PoC.net.arp.Broadcast_Requester

Todo: No documentation available.

Entity Declaration:

```

1 entity arp_BroadCast_Requester is
2   generic (
3     ALLOWED_PROTOCOL_IPV4      : boolean           := TRUE;
4     ALLOWED_PROTOCOL_IPV6      : boolean           := FALSE;
5   );
6   port (
7     Clock                      : in  std_logic;
8     Reset                      : in  std_logic;
9
10    SendRequest                 : in  std_logic;
11    Complete                    : out std_logic;
12
13    Address_rst                 : out std_logic;
14    SenderMACAddress_nxt        : out std_logic;
15    SenderMACAddress_Data       : in   T_SLV_8;
16    SenderIPv4Address_nxt       : out std_logic;
17    SenderIPv4Address_Data      : in   T_SLV_8;
18    TargetMACAddress_nxt        : out std_logic;
19    TargetMACAddress_Data       : in   T_SLV_8;
20    TargetIPv4Address_nxt       : out std_logic;
21    TargetIPv4Address_Data      : in   T_SLV_8;
22
23    TX_Valid                    : out std_logic;
24    TX_Data                     : out T_SLV_8;
25    TX_SOF                      : out std_logic;
26    TX_EOF                      : out std_logic;
27    TX_Ack                      : in  std_logic;
28    TX_Meta_DestMACAddress_rst   : in  std_logic;
29    TX_Meta_DestMACAddress_nxt   : in  std_logic;
30    TX_Meta_DestMACAddress_Data : out T_SLV_8;
31  );
32 end entity;
```

Source file: net/arp/arp_BroadCast_Requester.vhdl

PoC.net.arp.Cache

Todo: No documentation available.

Entity Declaration:

```

1 entity arp_Cache is
2   generic (
3     CLOCK_FREQ                 : FREQ              := 125 MHz;
4     REPLACEMENT_POLICY         : string            := "LRU";
5     TAG_BYTE_ORDER              : T_BYTE_ORDER      := BIG_
6     ↪ ENDIAN;
7     DATA_BYTE_ORDER            : T_BYTE_ORDER      := BIG_
8     ↪ ENDIAN;
```

(continues on next page)

(continued from previous page)

```

7     INITIAL_CACHE_CONTENT      : T_NET_ARP_ARPCACHE_VECTOR
8 );
9 port (
10     Clock                      : in  std_logic;           --
11     Reset                      : in  std_logic;           --
12
13     Command                    : in  T_NET_ARP_ARPCACHE_COMMAND;
14     Status                     : out  T_NET_ARP_ARPCACHE_STATUS;
15     NewIPv4Address_rst         : out  std_logic;
16     NewIPv4Address_nxt         : out  std_logic;
17     NewIPv4Address_Data        : in   T_SLV_8;
18     NewMACAddress_rst         : out  std_logic;
19     NewMACAddress_nxt         : out  std_logic;
20     NewMACAddress_Data        : in   T_SLV_8;
21
22     Lookup                     : in  std_logic;
23     IPv4Address_rst           : out  std_logic;
24     IPv4Address_nxt           : out  std_logic;
25     IPv4Address_Data          : in   T_SLV_8;
26
27     CacheResult                : out  T_CACHE_RESULT;
28     MACAddress_rst             : in  std_logic;
29     MACAddress_nxt             : in  std_logic;
30     MACAddress_Data            : out  T_SLV_8;
31 );
32 end entity;
```

Source file: [net/arp/arp_Cache.vhdl](#)

PoC.net.arp.IPPool

Todo: No documentation available.

Entity Declaration:

```

1 entity arp_IPPool is
2     generic (
3         IPPool_SIZE             : positive;
4         INITIAL_IPV4_ADDRESSES  : T_NET_IPV4_ADDRESS_VECTOR := (0 to 7 => C_
↳ NET_IPV4_ADDRESS_EMPTY)
5     );
6     port (
7         Clock                   : in  std_logic;           ↳
↳ --
8         Reset                   : in  std_logic;           ↳
↳ --
9
10    -- Command                   : in  T_ETHERNET_ARP_IPPOOL_COMMAND;
11    -- IPv4Address                : in  T_NET_IPV4_ADDRESS;
12    -- MACAddress                 : in  T_ETHERNET_MAC_ADDRESS;
13
14    Lookup                       : in  std_logic;
15    IPv4Address_rst              : out  std_logic;
16    IPv4Address_nxt              : out  std_logic;
17    IPv4Address_Data             : in   T_SLV_8;
18
```

(continues on next page)

(continued from previous page)

```

19     PoolResult                : out T_CACHE_RESULT
20 );
21 end entity;
```

Source file: [net/arp/arp_IPPool.vhdl](#)

PoC.net.arp.Tester

Todo: No documentation available.

Entity Declaration:

Source file: [net/arp/arp_Tester.vhdl](#)

PoC.net.arp.UniCast_Receiver

Todo: No documentation available.

Entity Declaration:

```

1 entity arp_UniCast_Receiver is
2   generic (
3     ALLOWED_PROTOCOL_IPV4      : boolean           := TRUE;
4     ALLOWED_PROTOCOL_IPV6      : boolean           := FALSE;
5   );
6   port (
7     Clock                      : in  std_logic;
8     Reset                      : in  std_logic;
9
10    RX_Valid                   : in  std_logic;
11    RX_Data                    : in  T_SLV_8;
12    RX_SOF                     : in  std_logic;
13    RX_EOF                     : in  std_logic;
14    RX_Ack                     : out  std_logic;
15    RX_Meta_rst                : out  std_logic;
16    RX_Meta_SrcMACAddress_nxt   : out  std_logic;
17    RX_Meta_SrcMACAddress_Data : in  T_SLV_8;
18    RX_Meta_DestMACAddress_nxt  : out  std_logic;
19    RX_Meta_DestMACAddress_Data : in  T_SLV_8;
20
21    Clear                      : in  std_logic;
22    Error                      : out  std_logic;
23
24    ResponseReceived           : out  std_logic;
25    Address_rst                : in  std_logic;
26    SenderMACAddress_nxt       : in  std_logic;
27    SenderMACAddress_Data      : out  T_SLV_8;
28    SenderIPAddress_nxt        : in  std_logic;
29    SenderIPAddress_Data       : out  T_SLV_8;
```

(continues on next page)

(continued from previous page)

```

30     TargetIPAddress_nxt      : in  std_logic;
31     TargetIPAddress_Data    : out  T_SLV_8;
32     TargetMACAddress_nxt    : in  std_logic;
33     TargetMACAddress_Data   : out  T_SLV_8
34 );
35 end entity;
```

Source file: `net/arp/arp_UniCast_Receiver.vhdl`

PoC.net.arp.UniCast_Responder

Todo: No documentation available.

Entity Declaration:

```

1  entity arp_UniCast_Responder is
2      generic (
3          ALLOWED_PROTOCOL_IPV4      : boolean           := TRUE;
4          ALLOWED_PROTOCOL_IPV6      : boolean           := FALSE;
5      );
6      port (
7          Clock                      : in  std_logic;
8          Reset                      : in  std_logic;
9
10         SendResponse               : in  std_logic;
11         Complete                   : out std_logic;
12
13         Address_rst                 : out std_logic;
14         SenderMACAddress_nxt        : out std_logic;
15         SenderMACAddress_Data       : in  T_SLV_8;
16         SenderIPv4Address_nxt       : out std_logic;
17         SenderIPv4Address_Data      : in  T_SLV_8;
18         TargetMACAddress_nxt        : out std_logic;
19         TargetMACAddress_Data       : in  T_SLV_8;
20         TargetIPv4Address_nxt       : out std_logic;
21         TargetIPv4Address_Data      : in  T_SLV_8;
22
23         TX_Valid                   : out std_logic;
24         TX_Data                    : out T_SLV_8;
25         TX_SOF                     : out std_logic;
26         TX_EOF                     : out std_logic;
27         TX_Ack                     : in  std_logic;
28         TX_Meta_DestMACAddress_rst  : in  std_logic;
29         TX_Meta_DestMACAddress_nxt  : in  std_logic;
30         TX_Meta_DestMACAddress_Data : out T_SLV_8;
31     );
32 end entity;
```

Source file: `net/arp/arp_UniCast_Responder.vhdl`

PoC.net.arp.Wrapper

Todo: No documentation available.

Entity Declaration:

```

1  entity arp_Wrapper is
2      generic (
3          CLOCK_FREQ                : FREQ                        := 125 MHz;
4          INTERFACE_MACADDRESS      : T_NET_MAC_ADDRESS          := C_NET_MAC_ADDRESS_EMPTY;
5          INITIAL_IPV4ADDRESSES     : T_NET_IPV4_ADDRESS_VECTOR  := (0 => C_NET_IPV4_ADDRESS_EMPTY);
6          INITIAL_ARPCACHE_CONTENT  : T_NET_ARP_ARPCACHE_VECTOR  := (0 => (Tag => C_NET_IPV4_ADDRESS_EMPTY, MAC => C_NET_MAC_ADDRESS_EMPTY));
7          APR_REQUEST_TIMEOUT       : time                       := 100 ms;
8      );
9      port (
10         Clock                     : in  std_logic;
11         Reset                     : in  std_logic;
12
13         IPPool_Announce           : in  std_logic;
14         IPPool_Announced         : out std_logic;
15
16         IPCache_Lookup            : in  std_logic;
17         IPCache_IPv4Address_rst   : out std_logic;
18         IPCache_IPv4Address_nxt   : out std_logic;
19         IPCache_IPv4Address_Data  : in  T_SLV_8;
20
21         IPCache_Valid             : out std_logic;
22         IPCache_MACAddress_rst    : in  std_logic;
23         IPCache_MACAddress_nxt    : in  std_logic;
24         IPCache_MACAddress_Data   : out T_SLV_8;
25
26         Eth_UC_TX_Valid           : out std_logic;
27         Eth_UC_TX_Data            : out T_SLV_8;
28         Eth_UC_TX_SOF             : out std_logic;
29         Eth_UC_TX_EOF             : out std_logic;
30         Eth_UC_TX_Ack             : in  std_logic;
31         Eth_UC_TX_Meta_rst        : in  std_logic;
32         Eth_UC_TX_Meta_DestMACAddress_nxt : in std_logic;
33         Eth_UC_TX_Meta_DestMACAddress_Data : out T_SLV_8;
34
35         Eth_UC_RX_Valid           : in  std_logic;
36         Eth_UC_RX_Data            : in  T_SLV_8;
37         Eth_UC_RX_SOF             : in  std_logic;
38         Eth_UC_RX_EOF             : in  std_logic;
39         Eth_UC_RX_Ack             : out std_logic;
40         Eth_UC_RX_Meta_rst        : out std_logic;
41         Eth_UC_RX_Meta_SrcMACAddress_nxt : out std_logic;
42         Eth_UC_RX_Meta_SrcMACAddress_Data : in  T_SLV_8;
43         Eth_UC_RX_Meta_DestMACAddress_nxt : out std_logic;
44         Eth_UC_RX_Meta_DestMACAddress_Data : in  T_SLV_8;
45
46         Eth_BC_RX_Valid           : in  std_logic;
47         Eth_BC_RX_Data            : in  T_SLV_8;
48         Eth_BC_RX_SOF             : in  std_logic;
49         Eth_BC_RX_EOF             : in  std_logic;
50         Eth_BC_RX_Ack             : out std_logic;

```

(continues on next page)

(continued from previous page)

```

51     Eth_BC_RX_Meta_rst           : out std_logic;
52     Eth_BC_RX_Meta_SrcMACAddress_nxt : out std_logic;
53     Eth_BC_RX_Meta_SrcMACAddress_Data : in  T_SLV_8;
54     Eth_BC_RX_Meta_DestMACAddress_nxt : out std_logic;
55     Eth_BC_RX_Meta_DestMACAddress_Data : in  T_SLV_8
56 );
57 end entity;
```

Source file: [net/arp/arp_Wrapper.vhdl](#)

7.13.2 PoC.net.eth

These are eth entities. . . .

PoC.net.eth.GEMAC_GMII

Todo: No documentation available.

Entity Declaration:

Source file: [net/eth/eth_GEMAC_GMII.vhdl](#)

PoC.net.eth.GEMAC_RX

Todo: No documentation available.

Entity Declaration:

Source file: [net/eth/eth_GEMAC_RX.vhdl](#)

PoC.net.eth.GEMAC_TX

Todo: No documentation available.

Entity Declaration:

Source file: [net/eth/eth_GEMAC_TX.vhdl](#)

PoC.net.eth.PHYController

Todo: No documentation available.

Entity Declaration:

Source file: [net/eth/eth_PHYController.vhdl](#)

PoC.net.eth.PHYController_Marvell_88E1111

Todo: No documentation available.

Entity Declaration:

Source file: [net/eth/eth_PHYController_Marvell_88E1111.vhdl](#)

PoC.net.eth.Wrapper

Todo: No documentation available.

Entity Declaration:

Source file: [net/eth/eth_Wrapper.vhdl](#)

7.13.3 PoC.net.icmpv4

These are icmpv4 entities....

PoC.net.icmpv4.RX

Todo: No documentation available.

Entity Declaration:

```

1  entity icmpv4_RX is
2      generic (
3          DEBUG                      : boolean                := FALSE
4      );
5      port (
6          Clock                      : in  std_logic;
7          Reset                      : in  std_logic;
8          -- CSE interface
9          Command                    : in  T_NET_ICMPV4_RX_COMMAND;
10         Status                     : out T_NET_ICMPV4_RX_STATUS;
11         Error                      : out T_NET_ICMPV4_RX_ERROR;
12         -- IN port
13         In_Valid                   : in  std_logic;
14         In_Data                    : in  T_SLV_8;

```

(continues on next page)

(continued from previous page)

```

15     In_SOF                                : in  std_logic;
16     In_EOF                                : in  std_logic;
17     In_Ack                                : out std_logic;
18     In_Meta_rst                           : out std_logic;
19     In_Meta_SrcMACAddress_nxt             : out std_logic;
20     In_Meta_SrcMACAddress_Data            : in   T_SLV_8;
21     In_Meta_DestMACAddress_nxt            : out std_logic;
22     In_Meta_DestMACAddress_Data           : in   T_SLV_8;
23     In_Meta_SrcIPv4Address_nxt            : out std_logic;
24     In_Meta_SrcIPv4Address_Data           : in   T_SLV_8;
25     In_Meta_DestIPv4Address_nxt           : out std_logic;
26     In_Meta_DestIPv4Address_Data          : in   T_SLV_8;
27     In_Meta_Length                        : in   T_SLV_16;
28     -- OUT Port
29     Out_Meta_rst                          : in  std_logic;
30     Out_Meta_SrcMACAddress_nxt             : in  std_logic;
31     Out_Meta_SrcMACAddress_Data            : out T_SLV_8;
32     Out_Meta_DestMACAddress_nxt            : in  std_logic;
33     Out_Meta_DestMACAddress_Data           : out T_SLV_8;
34     Out_Meta_SrcIPv4Address_nxt            : in  std_logic;
35     Out_Meta_SrcIPv4Address_Data           : out T_SLV_8;
36     Out_Meta_DestIPv4Address_nxt           : in  std_logic;
37     Out_Meta_DestIPv4Address_Data          : out T_SLV_8;
38     Out_Meta_Length                       : out T_SLV_16;
39     Out_Meta_Type                          : out T_SLV_8;
40     Out_Meta_Code                         : out T_SLV_8;
41     Out_Meta_Identification                : out T_SLV_16;
42     Out_Meta_SequenceNumber                : out T_SLV_16;
43     Out_Meta_Payload_nxt                  : in  std_logic;
44     Out_Meta_Payload_last                  : out std_logic;
45     Out_Meta_Payload_Data                  : out T_SLV_8
46 );
47 end entity;

```

Source file: `net/icmpv4/icmpv4_RX.vhdl`

PoC.net.icmpv4.TX

Todo: No documentation available.

Entity Declaration:

```

1  entity icmpv4_TX is
2      generic (
3          DEBUG                                : boolean                := FALSE;
4          SOURCE_IPV4ADDRESS                   : T_NET_IPV4_ADDRESS      := C_NET_IPV4_
5      ) ADDRESS_EMPTY
6  );
7  port (
8      Clock                                : in  std_logic;
9      --
10     Reset                                : in  std_logic;
11     --
12     -- CSE interface
13     Command                             : in   T_NET_ICMPV4_TX_COMMAND;
14     Status                              : out  T_NET_ICMPV4_TX_STATUS;

```

(continues on next page)

(continued from previous page)

```

12      Error                                : out T_NET_ICMPV4_TX_ERROR;
13      -- OUT port
14      Out_Valid                            : out std_logic;
15      Out_Data                             : out T_SLV_8;
16      Out_SOF                              : out std_logic;
17      Out_EOF                              : out std_logic;
18      Out_Ack                              : in  std_logic;
19      Out_Meta_rst                         : in  std_logic;
20      Out_Meta_SrcIPv4Address_nxt         : in  std_logic;
21      Out_Meta_SrcIPv4Address_Data        : out T_SLV_8;
22      Out_Meta_DestIPv4Address_nxt        : in  std_logic;
23      Out_Meta_DestIPv4Address_Data       : out T_SLV_8;
24      Out_Meta_Length                     : out T_SLV_16;
25      -- IN port
26      In_Meta_rst                         : out std_logic;
27      In_Meta_IPv4Address_nxt             : out std_logic;
28      In_Meta_IPv4Address_Data            : in  T_SLV_8;
29      In_Meta_Type                        : in  T_SLV_8;
30      In_Meta_Code                        : in  T_SLV_8;
31      In_Meta_Identification               : in  T_SLV_16;
32      In_Meta_SequenceNumber              : in  T_SLV_16;
33      In_Meta_Payload_nxt                 : out std_logic;
34      In_Meta_Payload_last                : in  std_logic;
35      In_Meta_Payload_Data                : in  T_SLV_8
36  );
37  end entity;
```

Source file: [net/icmpv4/icmpv4_TX.vhdl](#)

PoC.net.icmpv4.Wrapper

Todo: No documentation available.

Entity Declaration:

```

1  entity icmpv4_Wrapper is
2      generic (
3          DEBUG                                : boolean           := FALSE;
4          SOURCE_IPV4ADDRESS                   : T_NET_IPV4_ADDRESS := C_NET_IPV4_
5          ↪ ADDRESS_EMPTY
6      );
7      port (
8          Clock                                : in  std_logic;
9          Reset                                : in  std_logic;
10         -- CSE interface
11         Command                              : in  T_NET_ICMPV4_COMMAND;
12         Status                               : out T_NET_ICMPV4_STATUS;
13         Error                                : out T_NET_ICMPV4_ERROR;
14         -- Echo-Request destination address
15         IPv4Address_rst                      : out std_logic;
16         IPv4Address_nxt                      : out std_logic;
17         IPv4Address_Data                     : in  T_SLV_8;
18         -- to IPv4 layer
19         IP_TX_Valid                          : out std_logic;
20         IP_TX_Data                           : out T_SLV_8;
21         IP_TX_SOF                            : out std_logic;
```

(continues on next page)

(continued from previous page)

```

21     IP_TX_EOF                                : out std_logic;
22     IP_TX_Ack                                : in  std_logic;
23     IP_TX_Meta_rst                           : in  std_logic;
24     IP_TX_Meta_SrcIPv4Address_nxt            : in  std_logic;
25     IP_TX_Meta_SrcIPv4Address_Data           : out T_SLV_8;
26     IP_TX_Meta_DestIPv4Address_nxt          : in  std_logic;
27     IP_TX_Meta_DestIPv4Address_Data          : out T_SLV_8;
28     IP_TX_Meta_Length                        : out T_SLV_16;
29     -- from IPv4 layer
30     IP_RX_Valid                              : in  std_logic;
31     IP_RX_Data                               : in  T_SLV_8;
32     IP_RX_SOF                                : in  std_logic;
33     IP_RX_EOF                                : in  std_logic;
34     IP_RX_Ack                                : out std_logic;
35     IP_RX_Meta_rst                           : out std_logic;
36     IP_RX_Meta_SrcMACAddress_nxt             : out std_logic;
37     IP_RX_Meta_SrcMACAddress_Data            : in  T_SLV_8;
38     IP_RX_Meta_DestMACAddress_nxt            : out std_logic;
39     IP_RX_Meta_DestMACAddress_Data           : in  T_SLV_8;
40     -- IP_RX_Meta_EthType                    : in  T_SLV_16;
41     IP_RX_Meta_SrcIPv4Address_nxt            : out std_logic;
42     IP_RX_Meta_SrcIPv4Address_Data           : in  T_SLV_8;
43     IP_RX_Meta_DestIPv4Address_nxt           : out std_logic;
44     IP_RX_Meta_DestIPv4Address_Data          : in  T_SLV_8;
45     -- IP_RX_Meta_TrafficClass                : in  T_SLV_8;
46     -- IP_RX_Meta_FlowLabel                  : in  T_SLV_24;
47     IP_RX_Meta_Length                        : in  T_SLV_16
48     -- IP_RX_Meta_Protocol                    : in  T_SLV_8
49 );
50 end entity;
```

Source file: net/icmpv4/icmpv4_Wrapper.vhdl

7.13.4 PoC.net.icmpv6

These are icmpv6 entities. . . .

PoC.net.icmpv6.RX

Todo: No documentation available.

Entity Declaration:

Source file: net/icmpv6/icmpv6_RX.vhdl

PoC.net.icmpv6.TX

Todo: No documentation available.

Entity Declaration:

Source file: [net/icmpv6/icmpv6_TX.vhdl](#)

PoC.net.icmpv6.Wrapper

Todo: No documentation available.

Entity Declaration:

Source file: [net/icmpv6/icmpv6_Wrapper.vhdl](#)

7.13.5 PoC.net.ipv4

These are ipv4 entities...

PoC.net.ipv4.RX

Todo: No documentation available.

Entity Declaration:

```

1  entity ipv4_RX is
2      generic (
3          DEBUG                      : boolean           := FALSE
4      );
5      port (
6          Clock                      : in  std_logic;    --
7          Reset                      : in  std_logic;    --
8          -- STATUS port
9          Error                      : out std_logic;
10         -- IN port
11         In_Valid                   : in  std_logic;
12         In_Data                    : in  T_SLV_8;
13         In_SOF                     : in  std_logic;
14         In_EOF                     : in  std_logic;
15         In_Ack                     : out std_logic;
16         In_Meta_rst                : out std_logic;
17         In_Meta_SrcMACAddress_nxt  : out std_logic;
18         In_Meta_SrcMACAddress_Data : in  T_SLV_8;
19         In_Meta_DestMACAddress_nxt : out std_logic;
20         In_Meta_DestMACAddress_Data : in  T_SLV_8;
21         In_Meta_EthType            : in  T_SLV_16;
22         -- OUT port
23         Out_Valid                   : out std_logic;
24         Out_Data                    : out T_SLV_8;
25         Out_SOF                     : out std_logic;
26         Out_EOF                     : out std_logic;
27         Out_Ack                     : in  std_logic;
28         Out_Meta_rst                : in  std_logic;
29         Out_Meta_SrcMACAddress_nxt : in  std_logic;

```

(continues on next page)

(continued from previous page)

```

30 Out_Meta_SrcMACAddress_Data      : out T_SLV_8;
31 Out_Meta_DestMACAddress_nxt     : in  std_logic;
32 Out_Meta_DestMACAddress_Data    : out T_SLV_8;
33 Out_Meta_EthType                : out T_SLV_16;
34 Out_Meta_SrcIPv4Address_nxt     : in  std_logic;
35 Out_Meta_SrcIPv4Address_Data    : out T_SLV_8;
36 Out_Meta_DestIPv4Address_nxt    : in  std_logic;
37 Out_Meta_DestIPv4Address_Data   : out T_SLV_8;
38 Out_Meta_Length                 : out T_SLV_16;
39 Out_Meta_Protocol               : out T_SLV_8
40 );
41 end entity;
```

Source file: `net/ipv4/ipv4_RX.vhdl`

PoC.net.ipv4.TX

Todo: No documentation available.

Entity Declaration:

```

1  entity ipv4_TX is
2      generic (
3          DEBUG                      : boolean           := FALSE
4      );
5      port (
6          Clock                     : in  std_logic;      --
7          Reset                     : in  std_logic;      --
8          -- IN port
9          In_Valid                  : in  std_logic;
10         In_Data                    : in  T_SLV_8;
11         In_SOF                     : in  std_logic;
12         In_EOF                     : in  std_logic;
13         In_Ack                     : out std_logic;
14         In_Meta_rst                : out std_logic;
15         In_Meta_SrcIPv4Address_nxt : out std_logic;
16         In_Meta_SrcIPv4Address_Data : in  T_SLV_8;
17         In_Meta_DestIPv4Address_nxt : out std_logic;
18         In_Meta_DestIPv4Address_Data : in  T_SLV_8;
19         In_Meta_Length              : in  T_SLV_16;
20         In_Meta_Protocol            : in  T_SLV_8;
21         -- ARP port
22         ARP_IPCache_Query           : out std_logic;
23         ARP_IPCache_IPv4Address_rst : in  std_logic;
24         ARP_IPCache_IPv4Address_nxt : in  std_logic;
25         ARP_IPCache_IPv4Address_Data : out T_SLV_8;
26         ARP_IPCache_Valid           : in  std_logic;
27         ARP_IPCache_MACAddress_rst  : out std_logic;
28         ARP_IPCache_MACAddress_nxt  : out std_logic;
29         ARP_IPCache_MACAddress_Data : in  T_SLV_8;
30         -- OUT port
31         Out_Valid                   : out std_logic;
32         Out_Data                     : out T_SLV_8;
33         Out_SOF                     : out std_logic;
34         Out_EOF                     : out std_logic;
35         Out_Ack                     : in  std_logic;
```

(continues on next page)

(continued from previous page)

```
36     Out_Meta_rst                : in std_logic;
37     Out_Meta_DestMACAddress_nxt : in std_logic;
38     Out_Meta_DestMACAddress_Data : out T_SLV_8
39 );
40 end entity;
```

Source file: [net/ipv4/ipv4_TX.vhdl](#)

PoC.net.ipv4.FrameLoopback

Todo: No documentation available.

Entity Declaration:

```
1  entity ipv4_FrameLoopback is
2      generic (
3          MAX_FRAMES                : positive           := 4
4      );
5      port (
6          Clock                    : in std_logic;
7          Reset                    : in std_logic;
8          -- IN port
9          In_Valid                 : in std_logic;
10         In_Data                   : in T_SLV_8;
11         In_SOF                    : in std_logic;
12         In_EOF                    : in std_logic;
13         In_Ack                    : out std_logic;
14         In_Meta_rst               : out std_logic;
15         In_Meta_SrcIPv4Address_nxt : out std_logic;
16         In_Meta_SrcIPv4Address_Data : in T_SLV_8;
17         In_Meta_DestIPv4Address_nxt : out std_logic;
18         In_Meta_DestIPv4Address_Data : in T_SLV_8;
19         In_Meta_Length            : in T_SLV_16;
20         -- OUT port
21         Out_Valid                 : out std_logic;
22         Out_Data                   : out T_SLV_8;
23         Out_SOF                    : out std_logic;
24         Out_EOF                    : out std_logic;
25         Out_Ack                    : in std_logic;
26         Out_Meta_rst               : in std_logic;
27         Out_Meta_SrcIPv4Address_nxt : in std_logic;
28         Out_Meta_SrcIPv4Address_Data : out T_SLV_8;
29         Out_Meta_DestIPv4Address_nxt : in std_logic;
30         Out_Meta_DestIPv4Address_Data : out T_SLV_8;
31         Out_Meta_Length            : out T_SLV_16
32     );
33 end entity;
```

Source file: [net/ipv4/ipv4_FrameLoopback.vhdl](#)

PoC.net.ipv4.Wrapper

Todo: No documentation available.

Entity Declaration:

```

1  entity ipv4_Wrapper is
2      generic (
3          DEBUG                      : boolean                := FALSE;
4          PACKET_TYPES               : T_NET_IPV4_PROTOCOL_VECTOR := (0 => x"00")
5      );
6      port (
7          Clock                      : in  std_logic;
8          Reset                      : in  std_logic;
9          -- to MAC layer
10         MAC_TX_Valid               : out std_logic;
11         MAC_TX_Data                : out T_SLV_8;
12         MAC_TX_SOF                 : out std_logic;
13         MAC_TX_EOF                 : out std_logic;
14         MAC_TX_Ack                 : in  std_logic;
15         MAC_TX_Meta_rst            : in  std_logic;
16         MAC_TX_Meta_DestMACAddress_nxt : in  std_logic;
17         MAC_TX_Meta_DestMACAddress_Data : out T_SLV_8;
18         -- from MAC layer
19         MAC_RX_Valid               : in  std_logic;
20         MAC_RX_Data                : in  T_SLV_8;
21         MAC_RX_SOF                 : in  std_logic;
22         MAC_RX_EOF                 : in  std_logic;
23         MAC_RX_Ack                 : out std_logic;
24         MAC_RX_Meta_rst            : out std_logic;
25         MAC_RX_Meta_SrcMACAddress_nxt : out std_logic;
26         MAC_RX_Meta_SrcMACAddress_Data : in  T_SLV_8;
27         MAC_RX_Meta_DestMACAddress_nxt : out std_logic;
28         MAC_RX_Meta_DestMACAddress_Data : in  T_SLV_8;
29         MAC_RX_Meta_EthType        : in  T_SLV_16;
30         -- to ARP
31         ARP_IPCache_Query          : out std_logic;
32         ARP_IPCache_IPv4Address_rst : in  std_logic;
33         ARP_IPCache_IPv4Address_nxt : in  std_logic;
34         ARP_IPCache_IPv4Address_Data : out T_SLV_8;
35         -- from ARP
36         ARP_IPCache_Valid          : in  std_logic;
37         ARP_IPCache_MACAddress_rst : out std_logic;
38         ARP_IPCache_MACAddress_nxt : out std_logic;
39         ARP_IPCache_MACAddress_Data : in  T_SLV_8;
40         -- from upper layer
41         TX_Valid                   : in  std_logic_vector(PACKET_TYPES'length - 1,
42         ↪downto 0);
43         TX_Data                    : in  T_SLVV_8(PACKET_TYPES'length - 1 downto,
44         ↪0);
45         TX_SOF                    : in  std_logic_vector(PACKET_TYPES'length - 1,
46         ↪downto 0);
47         TX_EOF                    : in  std_logic_vector(PACKET_TYPES'length - 1,
48         ↪downto 0);
49         TX_Ack                    : out std_logic_vector(PACKET_TYPES'length - 1,
50         ↪downto 0);
51         TX_Meta_rst               : out std_logic_vector(PACKET_TYPES'length - 1,
52         ↪downto 0);
53         TX_Meta_SrcIPv4Address_nxt : out std_logic_vector(PACKET_TYPES'length - 1,
54         ↪downto 0);
55         TX_Meta_SrcIPv4Address_Data : in  T_SLVV_8(PACKET_TYPES'length - 1 downto,
56         ↪0);
57         TX_Meta_DestIPv4Address_nxt : out std_logic_vector(PACKET_TYPES'length - 1,
58         ↪downto 0);
59         TX_Meta_DestIPv4Address_Data : in  T_SLVV_8(PACKET_TYPES'length - 1 downto,
60         ↪0);

```

(continues on next page)

(continued from previous page)

```

51     TX_Meta_Length      : in  T_SLVV_16 (PACKET_TYPES'length - 1 downto
↳ 0);
52     -- to upper layer
53     RX_Valid            : out std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
54     RX_Data             : out T_SLVV_8 (PACKET_TYPES'length - 1 downto
↳ 0);
55     RX_SOF              : out std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
56     RX_EOF              : out std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
57     RX_Ack              : in  std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
58     RX_Meta_rst         : in  std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
59     RX_Meta_SrcMACAddress_nxt : in  std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
60     RX_Meta_SrcMACAddress_Data : out T_SLVV_8 (PACKET_TYPES'length - 1
↳ downto 0);
61     RX_Meta_DestMACAddress_nxt : in  std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
62     RX_Meta_DestMACAddress_Data : out T_SLVV_8 (PACKET_TYPES'length - 1
↳ downto 0);
63     RX_Meta_EthType      : out T_SLVV_16 (PACKET_TYPES'length - 1
↳ downto 0);
64     RX_Meta_SrcIPv4Address_nxt : in  std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
65     RX_Meta_SrcIPv4Address_Data : out T_SLVV_8 (PACKET_TYPES'length - 1
↳ downto 0);
66     RX_Meta_DestIPv4Address_nxt : in  std_logic_vector (PACKET_TYPES'length - 1
↳ downto 0);
67     RX_Meta_DestIPv4Address_Data : out T_SLVV_8 (PACKET_TYPES'length - 1
↳ downto 0);
68     RX_Meta_Length      : out T_SLVV_16 (PACKET_TYPES'length - 1
↳ downto 0);
69     RX_Meta_Protocol     : out T_SLVV_8 (PACKET_TYPES'length - 1
↳ downto 0);
70 );
71 end entity;
```

Source file: `net/ipv4/ipv4_Wrapper.vhdl`

7.13.6 PoC.net.ipv6

These are ipv6 entities....

PoC.net.ipv6.RX

Todo: No documentation available.

Entity Declaration:

```

1 entity ipv6_RX is
2   generic (
3     DEBUG                      : boolean          := FALSE
```

(continues on next page)

(continued from previous page)

```

4 );
5 port (
6     Clock                : in  std_logic;          --
7     Reset                : in  std_logic;          --
8     -- STATUS port
9     Error                : out std_logic;
10    -- IN port
11    In_Valid              : in  std_logic;
12    In_Data                : in  T_SLV_8;
13    In_SOF                : in  std_logic;
14    In_EOF                : in  std_logic;
15    In_Ack                : out std_logic;
16    In_Meta_rst           : out std_logic;
17    In_Meta_SrcMACAddress_nxt : out std_logic;
18    In_Meta_SrcMACAddress_Data : in  T_SLV_8;
19    In_Meta_DestMACAddress_nxt : out std_logic;
20    In_Meta_DestMACAddress_Data : in  T_SLV_8;
21    In_Meta_EthType       : in  T_SLV_16;
22    -- OUT port
23    Out_Valid             : out std_logic;
24    Out_Data              : out T_SLV_8;
25    Out_SOF              : out std_logic;
26    Out_EOF              : out std_logic;
27    Out_Ack              : in  std_logic;
28    Out_Meta_rst         : in  std_logic;
29    Out_Meta_SrcMACAddress_nxt : in  std_logic;
30    Out_Meta_SrcMACAddress_Data : out T_SLV_8;
31    Out_Meta_DestMACAddress_nxt : in  std_logic;
32    Out_Meta_DestMACAddress_Data : out T_SLV_8;
33    Out_Meta_EthType     : out T_SLV_16;
34    Out_Meta_SrcIPv6Address_nxt : in  std_logic;
35    Out_Meta_SrcIPv6Address_Data : out T_SLV_8;
36    Out_Meta_DestIPv6Address_nxt : in  std_logic;
37    Out_Meta_DestIPv6Address_Data : out T_SLV_8;
38    Out_Meta_TrafficClass : out T_SLV_8;
39    Out_Meta_FlowLabel    : out T_SLV_24; --STD_LOGIC_VECTOR(19 downto_
↳ 0);
40    Out_Meta_Length      : out T_SLV_16;
41    Out_Meta_NextHeader   : out T_SLV_8
42 );
43 end entity;
```

Source file: net/ipv6/ipv6_RX.vhdl

PoC.net.ipv6.TX

Todo: No documentation available.

Entity Declaration:

```

1 entity ipv6_TX is
2     generic (
3         DEBUG                : boolean                := FALSE
4     );
5     port (
6         Clock                : in  std_logic;          --
```

(continues on next page)

(continued from previous page)

```

7      Reset                                : in  std_logic;          --
8      -- IN port
9      In_Valid                             : in  std_logic;
10     In_Data                              : in  T_SLV_8;
11     In_SOF                               : in  std_logic;
12     In_EOF                               : in  std_logic;
13     In_Ack                               : out std_logic;
14     In_Meta_rst                           : out std_logic;
15     In_Meta_SrcIPv6Address_nxt           : out std_logic;
16     In_Meta_SrcIPv6Address_Data          : in  T_SLV_8;
17     In_Meta_DestIPv6Address_nxt          : out std_logic;
18     In_Meta_DestIPv6Address_Data         : in  T_SLV_8;
19     In_Meta_TrafficClass                  : in  T_SLV_8;
20     In_Meta_FlowLabel                    : in  T_SLV_24; --STD_LOGIC_VECTOR(19 downto_
↳ 0);
21     In_Meta_Length                        : in  T_SLV_16;
22     In_Meta_NextHeader                    : in  T_SLV_8;
23     -- to NDP layer
24     NDP_NextHop_Query                    : out std_logic;
25     NDP_NextHop_IPv6Address_rst          : in  std_logic;
26     NDP_NextHop_IPv6Address_nxt          : in  std_logic;
27     NDP_NextHop_IPv6Address_Data         : out T_SLV_8;
28     -- from NDP layer
29     NDP_NextHop_Valid                     : in  std_logic;
30     NDP_NextHop_MACAddress_rst           : out std_logic;
31     NDP_NextHop_MACAddress_nxt           : out std_logic;
32     NDP_NextHop_MACAddress_Data          : in  T_SLV_8;
33     -- OUT port
34     Out_Valid                             : out std_logic;
35     Out_Data                              : out T_SLV_8;
36     Out_SOF                               : out std_logic;
37     Out_EOF                               : out std_logic;
38     Out_Ack                               : in  std_logic;
39     Out_Meta_rst                           : in  std_logic;
40     Out_Meta_DestMACAddress_nxt           : in  std_logic;
41     Out_Meta_DestMACAddress_Data         : out T_SLV_8
42 );
43 end entity;
```

Source file: `net/ipv6/ipv6_TX.vhdl`

PoC.net.ipv6.FrameLoopback

Todo: No documentation available.

Entity Declaration:

```

1  entity ipv6_FrameLoopback is
2      generic (
3          MAX_FRAMES                        : positive           := 4
4      );
5      port (
6          Clock                             : in  std_logic;
7          Reset                             : in  std_logic;
8          -- IN port
9          In_Valid                           : in  std_logic;
```

(continues on next page)

(continued from previous page)

```

10   In_Data                : in  T_SLV_8;
11   In_SOF                 : in  std_logic;
12   In_EOF                 : in  std_logic;
13   In_Ack                 : out std_logic;
14   In_Meta_rst            : out std_logic;
15   In_Meta_SrcIPv6Address_nxt : out std_logic;
16   In_Meta_SrcIPv6Address_Data : in  T_SLV_8;
17   In_Meta_DestIPv6Address_nxt : out std_logic;
18   In_Meta_DestIPv6Address_Data : in  T_SLV_8;
19   In_Meta_Length         : in  T_SLV_16;
20   -- OUT port
21   Out_Valid              : out std_logic;
22   Out_Data               : out T_SLV_8;
23   Out_SOF                : out std_logic;
24   Out_EOF                : out std_logic;
25   Out_Ack                : in  std_logic;
26   Out_Meta_rst           : in  std_logic;
27   Out_Meta_SrcIPv6Address_nxt : in  std_logic;
28   Out_Meta_SrcIPv6Address_Data : out T_SLV_8;
29   Out_Meta_DestIPv6Address_nxt : in  std_logic;
30   Out_Meta_DestIPv6Address_Data : out T_SLV_8;
31   Out_Meta_Length        : out T_SLV_16
32 );
33 end entity;

```

Source file: `net/ipv6/ipv6_FrameLoopback.vhdl`

PoC.net.ipv6.Wrapper

Todo: No documentation available.

Entity Declaration:

```

1  entity ipv6_Wrapper is
2    generic (
3      DEBUG                : boolean                := FALSE;
4      PACKET_TYPES         : T_NET_IPV6_NEXT_HEADER_VECTOR := (0 => x"00
↳ ")
5    );
6    port (
7      Clock                : in  std_logic;
8      Reset                : in  std_logic;
9      -- to MAC layer
10     MAC_TX_Valid          : out std_logic;
11     MAC_TX_Data           : out T_SLV_8;
12     MAC_TX_SOF            : out std_logic;
13     MAC_TX_EOF            : out std_logic;
14     MAC_TX_Ack            : in  std_logic;
15     MAC_TX_Meta_rst       : in  std_logic;
16     MAC_TX_Meta_DestMACAddress_nxt : in  std_logic;
17     MAC_TX_Meta_DestMACAddress_Data : out T_SLV_8;
18     -- from MAC layer
19     MAC_RX_Valid          : in  std_logic;
20     MAC_RX_Data           : in  T_SLV_8;
21     MAC_RX_SOF            : in  std_logic;
22     MAC_RX_EOF            : in  std_logic;

```

(continues on next page)

(continued from previous page)

```

23     MAC_RX_Ack                                : out std_logic;
24     MAC_RX_Meta_rst                          : out std_logic;
25     MAC_RX_Meta_SrcMACAddress_nxt           : out std_logic;
26     MAC_RX_Meta_SrcMACAddress_Data         : in  T_SLV_8;
27     MAC_RX_Meta_DestMACAddress_nxt         : out std_logic;
28     MAC_RX_Meta_DestMACAddress_Data        : in  T_SLV_8;
29     MAC_RX_Meta_EthType                     : in  T_SLV_16;
30     -- to NDP layer
31     NDP_NextHop_Query                       : out std_logic;
32     NDP_NextHop_IPv6Address_rst             : in  std_logic;
33     NDP_NextHop_IPv6Address_nxt            : in  std_logic;
34     NDP_NextHop_IPv6Address_Data           : out T_SLV_8;
35     -- from NDP layer
36     NDP_NextHop_Valid                      : in  std_logic;
37     NDP_NextHop_MACAddress_rst             : out std_logic;
38     NDP_NextHop_MACAddress_nxt            : out std_logic;
39     NDP_NextHop_MACAddress_Data            : in  T_SLV_8;
40     -- from upper layer
41     TX_Valid                               : in  std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
42     TX_Data                                : in  T_SLVV_8(PACKET_TYPES'length - 1 downto
↳0);
43     TX_SOF                                 : in  std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
44     TX_EOF                                 : in  std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
45     TX_Ack                                 : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
46     TX_Meta_rst                           : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
47     TX_Meta_SrcIPv6Address_nxt            : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
48     TX_Meta_SrcIPv6Address_Data           : in  T_SLVV_8(PACKET_TYPES'length - 1 downto
↳0);
49     TX_Meta_DestIPv6Address_nxt           : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
50     TX_Meta_DestIPv6Address_Data          : in  T_SLVV_8(PACKET_TYPES'length - 1 downto
↳0);
51     TX_Meta_TrafficClass                  : in  T_SLVV_8(PACKET_TYPES'length - 1 downto
↳0);
52     TX_Meta_FlowLabel                     : in  T_SLVV_24(PACKET_TYPES'length - 1 downto
↳0);
53     TX_Meta_Length                        : in  T_SLVV_16(PACKET_TYPES'length - 1 downto
↳0);
54     -- to upper layer
55     RX_Valid                               : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
56     RX_Data                                : out T_SLVV_8(PACKET_TYPES'length - 1 downto
↳0);
57     RX_SOF                                 : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
58     RX_EOF                                 : out std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
59     RX_Ack                                 : in  std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
60     RX_Meta_rst                           : in  std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
61     RX_Meta_SrcMACAddress_nxt             : in  std_logic_vector(PACKET_TYPES'length - 1,
↳downto 0);
62     RX_Meta_SrcMACAddress_Data            : out T_SLVV_8(PACKET_TYPES'length - 1 downto
↳0);

```

(continues on next page)

(continued from previous page)

```

63     RX_Meta_DestMACAddress_nxt      : in std_logic_vector(PACKET_TYPES'length - 1_
↪downto 0);
64     RX_Meta_DestMACAddress_Data    : out T_SLVV_8(PACKET_TYPES'length - 1 downto_
↪0);
65     RX_Meta_EthType                : out T_SLVV_16(PACKET_TYPES'length - 1 downto_
↪0);
66     RX_Meta_SrcIPv6Address_nxt     : in std_logic_vector(PACKET_TYPES'length - 1_
↪downto 0);
67     RX_Meta_SrcIPv6Address_Data    : out T_SLVV_8(PACKET_TYPES'length - 1 downto_
↪0);
68     RX_Meta_DestIPv6Address_nxt    : in std_logic_vector(PACKET_TYPES'length - 1_
↪downto 0);
69     RX_Meta_DestIPv6Address_Data    : out T_SLVV_8(PACKET_TYPES'length - 1 downto_
↪0);
70     RX_Meta_TrafficClass           : out T_SLVV_8(PACKET_TYPES'length - 1 downto_
↪0);
71     RX_Meta_FlowLabel              : out T_SLVV_24(PACKET_TYPES'length - 1 downto_
↪0);
72     RX_Meta_Length                 : out T_SLVV_16(PACKET_TYPES'length - 1 downto_
↪0);
73     RX_Meta_NextHeader              : out T_SLVV_8(PACKET_TYPES'length - 1 downto_
↪0);
74 );
75 end entity;

```

Source file: net/ipv6/ipv6_Wrapper.vhdl

7.13.7 PoC.net.mac

These are mac entities...

PoC.net.mac.RX_DestMAC_Switch

Todo: No documentation available.

Entity Declaration:

```

1  entity mac_RX_DestMAC_Switch is
2      generic (
3          DEBUG                      : boolean                      := FALSE;
4          MAC_ADDRESSES              : T_NET_MAC_ADDRESS_VECTOR    := (0 => C_NET_MAC_
↪ADDRESS_EMPTY);
5          MAC_ADDRESSES_MASKS        : T_NET_MAC_ADDRESS_VECTOR    := (0 => C_NET_MAC_
↪MASK_DEFAULT)
6      );
7      port (
8          Clock                      : in std_logic;
9          Reset                      : in std_logic;
10
11         In_Valid                   : in std_logic;
12         In_Data                    : in T_SLV_8;
13         In_SOF                     : in std_logic;
14         In_EOF                     : in std_logic;
15         In_Ack                     : out std_logic;
16

```

(continues on next page)

(continued from previous page)

```

17     Out_Valid                                : out std_logic_vector (MAC_ADDRESSES'length - 1_
↪downto 0);
18     Out_Data                                : out T_SLVV_8 (MAC_ADDRESSES'length - 1_
↪0);
19     Out_SOF                                 : out std_logic_vector (MAC_ADDRESSES'length - 1_
↪downto 0);
20     Out_EOF                                 : out std_logic_vector (MAC_ADDRESSES'length - 1_
↪downto 0);
21     Out_Ack                                 : in std_logic_vector (MAC_ADDRESSES'length - 1_
↪downto 0);
22     Out_Meta_DestMACAddress_rst             : in std_logic_vector (MAC_ADDRESSES'length - 1_
↪downto 0);
23     Out_Meta_DestMACAddress_nxt             : in std_logic_vector (MAC_ADDRESSES'length - 1_
↪downto 0);
24     Out_Meta_DestMACAddress_Data            : out T_SLVV_8 (MAC_ADDRESSES'length - 1_
↪downto 0);
25 );
26 end entity;

```

Source file: net/mac/mac_RX_DestMAC_Switch.vhdl

PoC.net.mac.RX_SrcMAC_Filter

Todo: No documentation available.

Entity Declaration:

```

1 entity mac_RX_SrcMAC_Filter is
2     generic (
3         DEBUG                                : boolean                := FALSE;
4         MAC_ADDRESSES                        : T_NET_MAC_ADDRESS_VECTOR := (0 => C_NET_
↪MAC_ADDRESS_EMPTY);
5         MAC_ADDRESSES_MASKS                  : T_NET_MAC_ADDRESS_VECTOR := (0 => C_NET_
↪MAC_MASK_DEFAULT);
6     );
7     port (
8         Clock                                : in std_logic;
9         Reset                                : in std_logic;
10
11         In_Valid                             : in std_logic;
12         In_Data                              : in T_SLV_8;
13         In_SOF                               : in std_logic;
14         In_EOF                               : in std_logic;
15         In_Ack                               : out std_logic;
16         In_Meta_rst                          : out std_logic;
17         In_Meta_DestMACAddress_nxt           : out std_logic;
18         In_Meta_DestMACAddress_Data          : in T_SLV_8;
19
20         Out_Valid                           : out std_logic;
21         Out_Data                             : out T_SLV_8;
22         Out_SOF                             : out std_logic;
23         Out_EOF                             : out std_logic;
24         Out_Ack                             : in std_logic;
25         Out_Meta_rst                        : in std_logic;
26         Out_Meta_DestMACAddress_nxt          : in std_logic;
27         Out_Meta_DestMACAddress_Data         : out T_SLV_8;
28         Out_Meta_SrcMACAddress_nxt           : in std_logic;

```

(continues on next page)

(continued from previous page)

```

29     Out_Meta_SrcMACAddress_Data    : out T_SLV_8
30 );
31 end entity;
```

Source file: [net/mac/mac_RX_SrcMAC_Filter.vhdl](#)

PoC.net.mac.RX_Type_Switch

Todo: No documentation available.

Entity Declaration:

```

1  entity mac_RX_Type_Switch is
2      generic (
3          DEBUG                                : boolean                := FALSE;
4          ETHERNET_TYPES                       : T_NET_MAC_ETHERNETTYPE_VECTOR := (0 => C_NET_
↳MAC_ETHERNETTYPE_EMPTY)
5      );
6      port (
7          Clock                                : in  std_logic;
8          Reset                                : in  std_logic;
9
10         In_Valid                             : in  std_logic;
11         In_Data                               : in  T_SLV_8;
12         In_SOF                               : in  std_logic;
13         In_EOF                               : in  std_logic;
14         In_Ack                               : out std_logic;
15         In_Meta_rst                           : out std_logic;
16         In_Meta_SrcMACAddress_nxt             : out std_logic;
17         In_Meta_SrcMACAddress_Data            : in  T_SLV_8;
18         In_Meta_DestMACAddress_nxt            : out std_logic;
19         In_Meta_DestMACAddress_Data           : in  T_SLV_8;
20
21         Out_Valid                             : out std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
22         Out_Data                               : out T_SLVV_8(ETHERNET_TYPES'length - 1 downto_
↳0);
23         Out_SOF                               : out std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
24         Out_EOF                               : out std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
25         Out_Ack                               : in  std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
26         Out_Meta_rst                           : in  std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
27         Out_Meta_SrcMACAddress_nxt             : in  std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
28         Out_Meta_SrcMACAddress_Data            : out T_SLVV_8(ETHERNET_TYPES'length - 1 downto_
↳0);
29         Out_Meta_DestMACAddress_nxt            : in  std_logic_vector(ETHERNET_TYPES'length - 1_
↳downto 0);
30         Out_Meta_DestMACAddress_Data           : out T_SLVV_8(ETHERNET_TYPES'length - 1 downto_
↳0);
31         Out_Meta_EthType                       : out T_NET_MAC_ETHERNETTYPE_VECTOR(ETHERNET_
↳TYPES'length - 1 downto 0)
32     );
```

(continues on next page)

(continued from previous page)

```
end entity;
```

Source file: `net/mac/mac_RX_Type_Switch.vhdl`

PoC.net.mac.TX_SrcMAC_Prepender

Todo: No documentation available.

Entity Declaration:

```

1  entity mac_TX_SrcMAC_Prepender is
2      generic (
3          DEBUG                      : boolean                := FALSE;
4          MAC_ADDRESSES              : T_NET_MAC_ADDRESS_VECTOR := (0 => C_NET_
↳MAC_ADDRESS_EMPTY)
5      );
6      port (
7          Clock                     : in  std_logic;
8          Reset                     : in  std_logic;
9          -- IN Port
10         In_Valid                  : in  std_logic_vector(MAC_ADDRESSES'length - 1_
↳downto 0);
11         In_Data                   : in  T_SLVV_8(MAC_ADDRESSES'length - 1 downto_
↳0);
12         In_SOF                    : in  std_logic_vector(MAC_ADDRESSES'length - 1_
↳downto 0);
13         In_EOF                    : in  std_logic_vector(MAC_ADDRESSES'length - 1_
↳downto 0);
14         In_Ack                    : out std_logic_vector(MAC_ADDRESSES'length - 1_
↳downto 0);
15         In_Meta_rst               : out std_logic_vector(MAC_ADDRESSES'length - 1_
↳downto 0);
16         In_Meta_DestMACAddress_nxt : out std_logic_vector(MAC_ADDRESSES'length - 1_
↳downto 0);
17         In_Meta_DestMACAddress_Data : in  T_SLVV_8(MAC_ADDRESSES'length - 1 downto_
↳0);
18         -- OUT Port
19         Out_Valid                  : out std_logic;
20         Out_Data                   : out T_SLV_8;
21         Out_SOF                    : out std_logic;
22         Out_EOF                    : out std_logic;
23         Out_Ack                    : in  std_logic;
24         Out_Meta_rst               : in  std_logic;
25         Out_Meta_DestMACAddress_nxt : in  std_logic;
26         Out_Meta_DestMACAddress_Data : out T_SLV_8
27     );
28  end entity;
```

Source file: `net/mac/mac_TX_SrcMAC_Prepender.vhdl`

PoC.net.mac.TX_DestMAC_Prepender

Todo: No documentation available.

Entity Declaration:

```

1 entity mac_TX_DestMAC_Prepender is
2   generic (
3     DEBUG                      : boolean           := FALSE
4   );
5   port (
6     Clock                      : in  std_logic;
7     Reset                      : in  std_logic;
8
9     In_Valid                   : in  std_logic;
10    In_Data                     : in  T_SLV_8;
11    In_SOF                      : in  std_logic;
12    In_EOF                      : in  std_logic;
13    In_Ack                      : out std_logic;
14    In_Meta_rst                 : out std_logic;
15    In_Meta_DestMACAddress_nxt  : out std_logic;
16    In_Meta_DestMACAddress_Data : in  T_SLV_8;
17
18    Out_Valid                   : out std_logic;
19    Out_Data                     : out T_SLV_8;
20    Out_SOF                      : out std_logic;
21    Out_EOF                      : out std_logic;
22    Out_Ack                      : in  std_logic
23  );
24 end entity;

```

Source file: [net/mac/mac_TX_DestMAC_Prepender.vhdl](#)

PoC.net.mac.TX_Type_Prepender

Todo: No documentation available.

Entity Declaration:

Source file: [net/mac/mac_TX_Type_Prepender.vhdl](#)

PoC.net.mac.FrameLoopback

Todo: No documentation available.

Entity Declaration:

```

1 entity mac_FrameLoopback is
2   generic (
3     MAX_FRAMES                 : positive           := 4
4   );
5   port (
6     Clock                      : in  std_logic;
7     Reset                      : in  std_logic;
8     -- IN Port
9     In_Valid                   : in  std_logic;

```

(continues on next page)

(continued from previous page)

```

10      In_Data          : in  T_SLV_8;
11      In_SOF          : in  std_logic;
12      In_EOF          : in  std_logic;
13      In_Ack          : out std_logic;
14      In_Meta_rst     : out std_logic;
15      In_Meta_SrcMACAddress_nxt : out std_logic;
16      In_Meta_SrcMACAddress_Data : in  T_SLV_8;
17      In_Meta_DestMACAddress_nxt : out std_logic;
18      In_Meta_DestMACAddress_Data : in  T_SLV_8;
19      -- OUT Port
20      Out_Valid       : out std_logic;
21      Out_Data        : out T_SLV_8;
22      Out_SOF         : out std_logic;
23      Out_EOF         : out std_logic;
24      Out_Ack         : in  std_logic;
25      Out_Meta_rst    : in  std_logic;
26      Out_Meta_SrcMACAddress_nxt : in  std_logic;
27      Out_Meta_SrcMACAddress_Data : out T_SLV_8;
28      Out_Meta_DestMACAddress_nxt : in  std_logic;
29      Out_Meta_DestMACAddress_Data : out T_SLV_8
30    );
31  end entity;

```

Source file: [net/mac/mac_FrameLoopback.vhdl](#)

PoC.net.mac.Wrapper

Todo: No documentation available.

Entity Declaration:

```

1  entity mac_Wrapper is
2    generic (
3      DEBUG          : boolean              := FALSE;
4      MAC_CONFIG     : T_NET_MAC_CONFIGURATION_VECTOR
5    );
6    port (
7      Clock          : in  std_logic;
8      Reset          : in  std_logic;
9
10     Eth_TX_Valid    : out std_logic;
11     Eth_TX_Data     : out T_SLV_8;
12     Eth_TX_SOF      : out std_logic;
13     Eth_TX_EOF      : out std_logic;
14     Eth_TX_Ack      : in  std_logic;
15
16     Eth_RX_Valid    : in  std_logic;
17     Eth_RX_Data     : in  T_SLV_8;
18     Eth_RX_SOF      : in  std_logic;
19     Eth_RX_EOF      : in  std_logic;
20     Eth_RX_Ack      : out std_logic;
21
22     TX_Valid        : in  std_logic_vector (getPortCount (MAC_CONFIG) -
↪1  downto 0);
23     TX_Data         : in  T_SLVV_8 (getPortCount (MAC_CONFIG) - 1 downto
↪0);

```

(continues on next page)

(continued from previous page)

```

24 TX_SOF : in std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
25 TX_EOF : in std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
26 TX_Ack : out std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
27 TX_Meta_rst : out std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
28 TX_Meta_DestMACAddress_nxt : out std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
29 TX_Meta_DestMACAddress_Data : in T_SLVV_8 (getPortCount (MAC_CONFIG) - 1 downto
↳0);
30
31 RX_Valid : out std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
32 RX_Data : out T_SLVV_8 (getPortCount (MAC_CONFIG) - 1 downto
↳0);
33 RX_SOF : out std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
34 RX_EOF : out std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
35 RX_Ack : in std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
36 RX_Meta_rst : in std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
37 RX_Meta_SrcMACAddress_nxt : in std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
38 RX_Meta_SrcMACAddress_Data : out T_SLVV_8 (getPortCount (MAC_CONFIG) - 1 downto
↳0);
39 RX_Meta_DestMACAddress_nxt : in std_logic_vector (getPortCount (MAC_CONFIG) -
↳1 downto 0);
40 RX_Meta_DestMACAddress_Data : out T_SLVV_8 (getPortCount (MAC_CONFIG) - 1 downto
↳0);
41 RX_Meta_EthType : out T_NET_MAC_ETHERNETTYPE_
↳VECTOR (getPortCount (MAC_CONFIG) - 1 downto 0)
42 );
43 end entity;
```

Source file: `net/mac/mac_Wrapper.vhdl`

7.13.8 PoC.net.ndp

These are ndp entities. ...

PoC.net.ndp.DestinationCache

Todo: No documentation available.

Entity Declaration:

Source file: `net/ndp/ndp_DestinationCache.vhdl`

PoC.net.ndp.FSMQuery

Todo: No documentation available.

Entity Declaration:

Source file: [net/ndp/ndp_FSMQuery.vhdl](#)

PoC.net.ndp.NeighborCache

Todo: No documentation available.

Entity Declaration:

Source file: [net/ndp/ndp_NeighborCache.vhdl](#)

PoC.net.ndp.Wrapper

Todo: No documentation available.

Entity Declaration:

Source file: [net/ndp/ndp_Wrapper.vhdl](#)

7.13.9 PoC.net.stack

These are udp entities. . . .

stack_IPv4

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

stack_IPv6

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

PoC.net.stack.UDIPv4

Todo: No documentation available.

Entity Declaration:

Source file: [net/stack/stack_UDIPv4.vhdl](#)

stack_UDIPv6

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

stack_MAC

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

7.13.10 PoC.net.udp

These are udp entities. . .

PoC.net.udp.RX

Todo: No documentation available.

Entity Declaration:

```

1  entity udp_RX is
2      generic (
3          DEBUG                : boolean           := FALSE;
4          IP_VERSION           : positive          := 6
5      );
6      port (
7          Clock                : in  std_logic;    --
8          Reset                : in  std_logic;    --
9          -- STATUS port
10         Error                : out std_logic;
11         -- IN port
12         In_Valid             : in  std_logic;
13         In_Data              : in  T_SLV_8;

```

(continues on next page)

(continued from previous page)

```

14     In_SOF                : in std_logic;
15     In_EOF                : in std_logic;
16     In_Ack                : out std_logic;
17     In_Meta_rst           : out std_logic;
18     In_Meta_SrcMACAddress_nxt : out std_logic;
19     In_Meta_SrcMACAddress_Data : in T_SLV_8;
20     In_Meta_DestMACAddress_nxt : out std_logic;
21     In_Meta_DestMACAddress_Data : in T_SLV_8;
22     In_Meta_EthType        : in T_SLV_16;
23     In_Meta_SrcIPAddress_nxt : out std_logic;
24     In_Meta_SrcIPAddress_Data : in T_SLV_8;
25     In_Meta_DestIPAddress_nxt : out std_logic;
26     In_Meta_DestIPAddress_Data : in T_SLV_8;
27     -- In_Meta_TrafficClass    : in T_SLV_8;
28     -- In_Meta_FlowLabel       : in T_SLV_24;
29     In_Meta_Length          : in T_SLV_16;
30     In_Meta_Protocol        : in T_SLV_8;
31     -- OUT port
32     Out_Valid               : out std_logic;
33     Out_Data                : out T_SLV_8;
34     Out_SOF                 : out std_logic;
35     Out_EOF                 : out std_logic;
36     Out_Ack                 : in std_logic;
37     Out_Meta_rst            : in std_logic;
38     Out_Meta_SrcMACAddress_nxt : in std_logic;
39     Out_Meta_SrcMACAddress_Data : out T_SLV_8;
40     Out_Meta_DestMACAddress_nxt : in std_logic;
41     Out_Meta_DestMACAddress_Data : out T_SLV_8;
42     Out_Meta_EthType        : out T_SLV_16;
43     Out_Meta_SrcIPAddress_nxt : in std_logic;
44     Out_Meta_SrcIPAddress_Data : out T_SLV_8;
45     Out_Meta_DestIPAddress_nxt : in std_logic;
46     Out_Meta_DestIPAddress_Data : out T_SLV_8;
47     -- Out_Meta_TrafficClass    : out T_SLV_8;
48     -- Out_Meta_FlowLabel       : out T_SLV_24;
49     Out_Meta_Length          : out T_SLV_16;
50     Out_Meta_Protocol        : out T_SLV_8;
51     Out_Meta_SrcPort         : out T_SLV_16;
52     Out_Meta_DestPort        : out T_SLV_16;
53 );
54 end entity;

```

Source file: net/udp/udp_RX.vhdl

PoC.net.udp.TX

Todo: No documentation available.

Entity Declaration:

```

1 entity udp_TX is
2   generic (
3     DEBUG                : boolean          := FALSE;
4     IP_VERSION            : positive         := 6
5   );
6   port (

```

(continues on next page)

(continued from previous page)

```

7      Clock                : in  std_logic;          --
8      Reset                : in  std_logic;          --
9      -- IN port
10     In_Valid              : in  std_logic;
11     In_Data                : in  T_SLV_8;
12     In_SOF                 : in  std_logic;
13     In_EOF                 : in  std_logic;
14     In_Ack                 : out std_logic;
15     In_Meta_rst            : out std_logic;
16     In_Meta_SrcIPAddress_nxt : out std_logic;
17     In_Meta_SrcIPAddress_Data : in  T_SLV_8;
18     In_Meta_DestIPAddress_nxt : out std_logic;
19     In_Meta_DestIPAddress_Data : in  T_SLV_8;
20     In_Meta_SrcPort         : in  T_SLV_16;
21     In_Meta_DestPort        : in  T_SLV_16;
22     In_Meta_Length          : in  T_SLV_16;
23     In_Meta_Checksum        : in  T_SLV_16;
24     -- OUT port
25     Out_Valid              : out std_logic;
26     Out_Data                : out T_SLV_8;
27     Out_SOF                 : out std_logic;
28     Out_EOF                 : out std_logic;
29     Out_Ack                 : in  std_logic;
30     Out_Meta_rst            : in  std_logic;
31     Out_Meta_SrcIPAddress_nxt : in  std_logic;
32     Out_Meta_SrcIPAddress_Data : out T_SLV_8;
33     Out_Meta_DestIPAddress_nxt : in  std_logic;
34     Out_Meta_DestIPAddress_Data : out T_SLV_8;
35     Out_Meta_Length          : out T_SLV_16
36 );
37 end entity;
```

Source file: `net/udp/udp_TX.vhdl`

PoC.net.udp.FrameLoopback

Todo: No documentation available.

Entity Declaration:

```

1  entity udp_FrameLoopback is
2      generic (
3          IP_VERSION          : positive          := 6;
4          MAX_FRAMES          : positive          := 4;
5      );
6      port (
7          Clock                : in  std_logic;
8          Reset                : in  std_logic;
9          -- IN port
10         In_Valid              : in  std_logic;
11         In_Data                : in  T_SLV_8;
12         In_SOF                 : in  std_logic;
13         In_EOF                 : in  std_logic;
14         In_Ack                 : out std_logic;
15         In_Meta_rst            : out std_logic;
16         In_Meta_DestIPAddress_nxt : out std_logic;
```

(continues on next page)

(continued from previous page)

```

17 In_Meta_DestIPAddress_Data : in T_SLV_8;
18 In_Meta_SrcIPAddress_nxt : out std_logic;
19 In_Meta_SrcIPAddress_Data : in T_SLV_8;
20 In_Meta_DestPort : in T_NET_UDP_PORT;
21 In_Meta_SrcPort : in T_NET_UDP_PORT;
22 -- OUT port
23 Out_Valid : out std_logic;
24 Out_Data : out T_SLV_8;
25 Out_SOF : out std_logic;
26 Out_EOF : out std_logic;
27 Out_Ack : in std_logic;
28 Out_Meta_rst : in std_logic;
29 Out_Meta_DestIPAddress_nxt : in std_logic;
30 Out_Meta_DestIPAddress_Data : out T_SLV_8;
31 Out_Meta_SrcIPAddress_nxt : in std_logic;
32 Out_Meta_SrcIPAddress_Data : out T_SLV_8;
33 Out_Meta_DestPort : out T_NET_UDP_PORT;
34 Out_Meta_SrcPort : out T_NET_UDP_PORT
35 );
36 end entity;

```

Source file: [net/udp/udp_FrameLoopback.vhdl](#)

PoC.net.udp.Wrapper

Todo: No documentation available.

Entity Declaration:

```

1 entity udp_Wrapper is
2   generic (
3     DEBUG : boolean := FALSE;
4     IP_VERSION : positive := 6;
5     PORTPAIRS : T_NET_UDP_PORTPAIR_VECTOR := (0 => (x
6     ↪ "0000", x"0000"))
7   );
8   port (
9     Clock : in std_logic;
10    Reset : in std_logic;
11    -- from IP layer
12    IP_TX_Valid : out std_logic;
13    IP_TX_Data : out T_SLV_8;
14    IP_TX_SOF : out std_logic;
15    IP_TX_EOF : out std_logic;
16    IP_TX_Ack : in std_logic;
17    IP_TX_Meta_rst : in std_logic;
18    IP_TX_Meta_SrcIPAddress_nxt : in std_logic;
19    IP_TX_Meta_SrcIPAddress_Data : out T_SLV_8;
20    IP_TX_Meta_DestIPAddress_nxt : in std_logic;
21    IP_TX_Meta_DestIPAddress_Data : out T_SLV_8;
22    IP_TX_Meta_Length : out T_SLV_16;
23    -- to IP layer
24    IP_RX_Valid : in std_logic;
25    IP_RX_Data : in T_SLV_8;
26    IP_RX_SOF : in std_logic;
27    IP_RX_EOF : in std_logic;

```

(continues on next page)

(continued from previous page)

```

27     IP_RX_Ack                                : out std_logic;
28     IP_RX_Meta_rst                          : out std_logic;
29     IP_RX_Meta_SrcMACAddress_nxt            : out std_logic;
30     IP_RX_Meta_SrcMACAddress_Data           : in  T_SLV_8;
31     IP_RX_Meta_DestMACAddress_nxt           : out std_logic;
32     IP_RX_Meta_DestMACAddress_Data          : in  T_SLV_8;
33     IP_RX_Meta_EthType                      : in  T_SLV_16;
34     IP_RX_Meta_SrcIPAddress_nxt             : out std_logic;
35     IP_RX_Meta_SrcIPAddress_Data            : in  T_SLV_8;
36     IP_RX_Meta_DestIPAddress_nxt            : out std_logic;
37     IP_RX_Meta_DestIPAddress_Data           : in  T_SLV_8;
38     -- IP_RX_Meta_TrafficClass               : in  T_SLV_8;
39     -- IP_RX_Meta_FlowLabel                  : in  T_SLV_24;
40     IP_RX_Meta_Length                       : in  T_SLV_16;
41     IP_RX_Meta_Protocol                     : in  T_SLV_8;
42     -- from upper layer
43     TX_Valid                                : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
44     TX_Data                                 : in  T_SLVV_8(PORTPAIRS'length - 1 downto_
↳0);
45     TX_SOF                                  : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
46     TX_EOF                                  : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
47     TX_Ack                                  : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
48     TX_Meta_rst                            : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
49     TX_Meta_SrcIPAddress_nxt                : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
50     TX_Meta_SrcIPAddress_Data               : in  T_SLVV_8(PORTPAIRS'length - 1 downto_
↳0);
51     TX_Meta_DestIPAddress_nxt               : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
52     TX_Meta_DestIPAddress_Data              : in  T_SLVV_8(PORTPAIRS'length - 1 downto_
↳0);
53     TX_Meta_SrcPort                         : in  T_SLVV_16(PORTPAIRS'length - 1 downto_
↳0);
54     TX_Meta_DestPort                       : in  T_SLVV_16(PORTPAIRS'length - 1 downto_
↳0);
55     TX_Meta_Length                         : in  T_SLVV_16(PORTPAIRS'length - 1 downto_
↳0);
56     -- to upper layer
57     RX_Valid                                : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
58     RX_Data                                 : out T_SLVV_8(PORTPAIRS'length - 1 downto_
↳0);
59     RX_SOF                                  : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
60     RX_EOF                                  : out std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
61     RX_Ack                                  : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
62     RX_Meta_rst                            : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
63     RX_Meta_SrcMACAddress_nxt               : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);
64     RX_Meta_SrcMACAddress_Data              : out T_SLVV_8(PORTPAIRS'length - 1 downto_
↳0);
65     RX_Meta_DestMACAddress_nxt              : in  std_logic_vector(PORTPAIRS'length - 1_
↳downto 0);

```

(continues on next page)

(continued from previous page)

```

66     RX_Meta_DestMACAddress_Data      : out T_SLVV_8(PORTPAIRS'length - 1 downto
↪0);
67     RX_Meta_EthType                  : out T_SLVV_16(PORTPAIRS'length - 1 downto
↪0);
68     RX_Meta_SrcIPAddress_nxt         : in  std_logic_vector(PORTPAIRS'length - 1
↪downto 0);
69     RX_Meta_SrcIPAddress_Data        : out T_SLVV_8(PORTPAIRS'length - 1 downto
↪0);
70     RX_Meta_DestIPAddress_nxt        : in  std_logic_vector(PORTPAIRS'length - 1
↪downto 0);
71     RX_Meta_DestIPAddress_Data       : out T_SLVV_8(PORTPAIRS'length - 1 downto
↪0);
72     -- RX_Meta_TrafficClass           : out T_SLVV_8(PORTPAIRS'length - 1 downto
↪0);
73     -- RX_Meta_FlowLabel              : out T_SLVV_24(PORTPAIRS'length - 1
↪downto 0);
74     RX_Meta_Length                   : out T_SLVV_16(PORTPAIRS'length - 1 downto
↪0);
75     RX_Meta_Protocol                 : out T_SLVV_8(PORTPAIRS'length - 1 downto
↪0);
76     RX_Meta_SrcPort                  : out T_SLVV_16(PORTPAIRS'length - 1 downto
↪0);
77     RX_Meta_DestPort                  : out T_SLVV_16(PORTPAIRS'length - 1 downto
↪0);
78     );
79 end entity;
```

Source file: net/udp/udp_Wrapper.vhdl

7.13.11 PoC.net Package

Source file: net.pkg.vhdl

7.13.12 PoC.net.FrameChecksum

Todo: No documentation available.

Entity Declaration:

```

1  entity net_FrameChecksum is
2      generic (
3          MAX_FRAMES                  : positive      := 8;
4          MAX_FRAME_LENGTH            : positive      := 2048;
5          META_BITS                   : T_POSVEC      := (0 => 8);
6          META_FIFO_DEPTH             : T_POSVEC      := (0 => 16)
7      );
8      port (
9          Clock                       : in  std_logic;
10         Reset                         : in  std_logic;
11         -- IN port
12         In_Valid                     : in  std_logic;
13         In_Data                       : in  T_SLV_8;
14         In_SOF                       : in  std_logic;
15         In_EOF                       : in  std_logic;
16         In_Ack                       : out std_logic;
```

(continues on next page)

(continued from previous page)

```

17     In_Meta_rst                : out std_logic;
18     In_Meta_nxt                : out std_logic_vector(META_BITS'length - 1_
↳downto 0);
19     In_Meta_Data               : in  std_logic_vector(isum(META_BITS) - 1_
↳downto 0);
20     -- OUT port
21     Out_Valid                  : out std_logic;
22     Out_Data                    : out T_SLV_8;
23     Out_SOF                     : out std_logic;
24     Out_EOF                     : out std_logic;
25     Out_Ack                     : in  std_logic;
26     Out_Meta_rst               : in  std_logic;
27     Out_Meta_nxt               : in  std_logic_vector(META_BITS'length - 1_
↳downto 0);
28     Out_Meta_Data              : out std_logic_vector(isum(META_BITS) - 1_
↳downto 0);
29     Out_Meta_Length             : out T_SLV_16;
30     Out_Meta_Checksum           : out T_SLV_16
31 );
32 end entity;
```

Source file: [net/net_FrameChecksum.vhdl](#)

7.13.13 PoC.net.FrameLoopback

Todo: No documentation available.

Entity Declaration:

```

1  entity FrameLoopback is
2      generic (
3          DATA_BW                : positive      := 8;
4          META_BW                  : natural        := 0;
5      );
6      port (
7          Clock                    : in  std_logic;
8          Reset                     : in  std_logic;
9
10         In_Valid                  : in  std_logic;
11         In_Data                    : in  std_logic_vector(DATA_BW - 1_downto 0);
12         In_Meta                    : in  std_logic_vector(META_BW - 1_downto 0);
13         In_SOF                     : in  std_logic;
14         In_EOF                     : in  std_logic;
15         In_Ack                     : out std_logic;
16
17
18         Out_Valid                  : out std_logic;
19         Out_Data                    : out std_logic_vector(DATA_BW - 1_downto 0);
20         Out_Meta                    : out std_logic_vector(META_BW - 1_downto 0);
21         Out_SOF                     : out std_logic;
22         Out_EOF                     : out std_logic;
23         Out_Ack                     : in  std_logic
24     );
25 end entity;
```

Source file: [net/net_FrameLoopback.vhdl](#)

7.14 PoC.sort

These are sorting entities. . . .

Sub-Namespaces

- *PoC.sort.sortnet*

Entities

- IP:sort_ExpireList
- IP:sort_InsertSort
- *PoC.sort.LeastFrequentlyUsed*
- *PoC.sort.lru_cache*
- *PoC.sort.lru_list*

7.14.1 PoC.sort.sortnet

This sub-namespace contains sorting network implementations.

Entities

- *PoC.sort.sortnet.BitonicSort*
- *PoC.sort.sortnet.MergeSort_Streamed*
- *PoC.sort.sortnet.OddEvenMergeSort*
- *PoC.sort.sortnet.OddEvenSort*
- *PoC.sort.sortnet.Stream_Adapter*
- *PoC.sort.sortnet.Stream_Adapter2*
- *PoC.sort.sortnet.Transform*

PoC.sort.sortnet Package

```
type T_SORTNET_IMPL is (  
    SORT_SORTNET_IMPL_ODDEVEN_SORT,  
    SORT_SORTNET_IMPL_ODDEVEN_MERGESORT,  
    SORT_SORTNET_IMPL_BITONIC_SORT  
);
```

T_SORTNET_IMPL

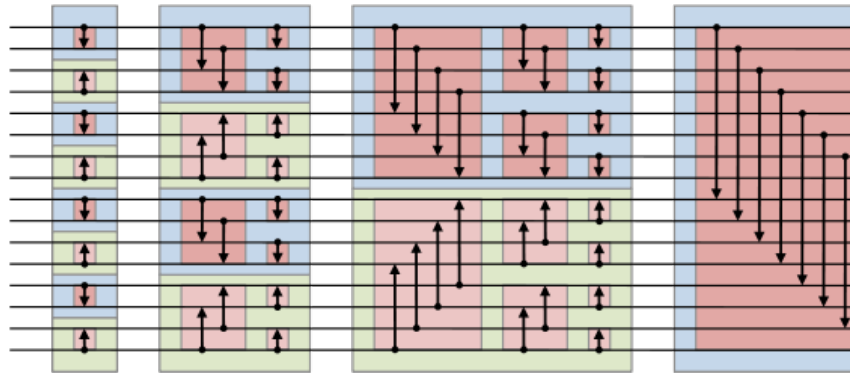
SORT_SORTNET_IMPL_ODDEVEN_SORT Instantiate a *PoC.sort.sortnet.OddEvenSort* sorting network.

SORT_SORTNET_IMPL_ODDEVEN_MERGESORT Instantiate a *PoC.sort.sortnet.OddEvenMergeSort* sorting network.

SORT_SORTNET_IMPL_BITONIC_SORT Instantiate a *PoC.sort.sortnet.BitonicSort* sorting network.

Source file: *sortnet.pkg.vhdl*

PoC.sort.sortnet.BitonicSort



This sorting network uses the *bitonic sort* algorithm.

Entity Declaration:

```

1  entity sortnet_BitonicSort is
2      generic (
3          INPUTS          : positive := 32;      -- input count
4          KEY_BITS         : positive := 32;      -- the first KEY_BITS of In_Data_
5          DATA_BITS       : positive := 64;      -- inclusive KEY_BITS
6          META_BITS        : natural  := 2;      -- additional bits, not sorted_
7          PIPELINE_STAGE_AFTER : natural := 2;    -- add a pipeline stage after n_
8          ADD_INPUT_REGISTERS : boolean := FALSE; --
9          ADD_OUTPUT_REGISTERS : boolean := TRUE  --
10     );
11     port (
12         Clock      : in  std_logic;
13         Reset      : in  std_logic;
14
15         Inverse     : in  std_logic := '0';
16
17         In_Valid    : in  std_logic;
18         In_IsKey    : in  std_logic;
19         In_Data     : in  T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
20         In_Meta     : in  std_logic_vector(META_BITS - 1 downto 0);
21
22         Out_Valid   : out std_logic;
23         Out_IsKey   : out std_logic;
24         Out_Data    : out T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
25         Out_Meta    : out std_logic_vector(META_BITS - 1 downto 0);
26     );
27 end entity;
```

Source file: sort/sortnet/sortnet_BitonicSort.vhdl

PoC.sort.sortnet.MergeSort_Streamed

Todo: No documentation available.

Entity Declaration:

```

1 entity sortnet_MergeSort_Streamed is
2   generic (
3     FIFO_DEPTH    : positive := 32;
4     KEY_BITS      : positive := 32;
5     DATA_BITS    : positive := 32;
6   );
7   port (
8     Clock      : in  std_logic;
9     Reset      : in  std_logic;
10
11     Inverse     : in  std_logic := '0';
12
13     In_Valid    : in  std_logic;
14     In_Data     : in  std_logic_vector(DATA_BITS - 1 downto 0);
15     In_SOF      : in  std_logic;
16     In_IsKey    : in  std_logic;
17     In_EOF      : in  std_logic;
18     In_Ack      : out std_logic;
19
20     Out_Sync    : out std_logic;
21     Out_Valid   : out std_logic;
22     Out_Data    : out std_logic_vector(DATA_BITS - 1 downto 0);
23     Out_SOF     : out std_logic;
24     Out_IsKey   : out std_logic;
25     Out_EOF     : out std_logic;
26     Out_Ack     : in  std_logic;
27   );
28 end entity;

```

Source file: sort/sortnet/sortnet_MergeSort_Streamed.vhdl

PoC.sortnet.OddEvenMergeSort

Todo: No documentation available.

Entity Declaration:

```

1 entity sortnet_OddEvenMergeSort is
2   generic (
3     INPUTS          : positive := 128;  -- input count
4     KEY_BITS        : positive := 32;    -- the first KEY_BITS of In_Data_
5     DATA_BITS      : positive := 32;    -- inclusive KEY_BITS
6     META_BITS       : natural  := 2;     -- additional bits, not sorted but_
7     PIPELINE_STAGE_AFTER : natural := 2;  -- add a pipeline stage after n_
8     ADD_INPUT_REGISTERS : boolean := FALSE; --
9     ADD_OUTPUT_REGISTERS : boolean := TRUE  --
10  );
11  port (
12    Clock      : in  std_logic;
13    Reset      : in  std_logic;
14
15    Inverse     : in  std_logic := '0';

```

(continues on next page)

(continued from previous page)

```

16      In_Valid      : in  std_logic;
17      In_IsKey     : in  std_logic;
18      In_Data      : in  T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
19      In_Meta      : in  std_logic_vector(META_BITS - 1 downto 0);
20
21
22      Out_Valid     : out std_logic;
23      Out_IsKey     : out std_logic;
24      Out_Data      : out T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
25      Out_Meta      : out std_logic_vector(META_BITS - 1 downto 0)
26  );
27  end entity;

```

Source file: sort/sortnet/sortnet_OddEvenMergeSort.vhdl

PoC.sort.sortnet.OddEvenSort

Todo: No documentation available.

Entity Declaration:

```

1  entity sortnet_OddEvenSort is
2      generic (
3          INPUTS          : positive := 8;      -- input count
4          KEY_BITS        : positive := 32;      -- the first KEY_BITS of In_Data_
5          DATA_BITS      : positive := 32;      -- inclusive KEY_BITS
6          META_BITS       : natural  := 2;      -- additional bits, not sorted but_
7          PIPELINE_STAGE_AFTER : natural := 2;    -- add a pipeline stage after n_
8          ADD_INPUT_REGISTERS : boolean := FALSE; --
9          ADD_OUTPUT_REGISTERS : boolean := TRUE  --
10     );
11     port (
12         Clock      : in  std_logic;
13         Reset      : in  std_logic;
14
15         Inverse     : in  std_logic := '0';
16
17         In_Valid    : in  std_logic;
18         In_IsKey    : in  std_logic;
19         In_Data     : in  T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
20         In_Meta     : in  std_logic_vector(META_BITS - 1 downto 0);
21
22         Out_Valid   : out std_logic;
23         Out_IsKey   : out std_logic;
24         Out_Data    : out T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
25         Out_Meta    : out std_logic_vector(META_BITS - 1 downto 0)
26     );
27  end entity;

```

Source file: sort/sortnet/sortnet_OddEvenSort.vhdl

PoC.sort.sortnet.Stream_Adapter

Todo: No documentation available.

Entity Declaration:

```

1 entity sortnet_Stream_Adapter is
2   generic (
3     STREAM_DATA_BITS      : positive      := 32;
4     STREAM_META_BITS      : positive      := 2;
5     SORTNET_IMPL          : T_SORTNET_IMPL := SORT_SORTNET_IMPL_ODDEVEN_MERGESORT;
6     SORTNET_SIZE          : positive      := 32;
7     SORTNET_KEY_BITS      : positive      := 32;
8     SORTNET_DATA_BITS     : natural       := 32;
9     INVERSE               : boolean       := FALSE
10  );
11  port (
12    Clock      : in  std_logic;
13    Reset      : in  std_logic;
14
15    In_Valid   : in  std_logic;
16    In_IsKey   : in  std_logic;
17    In_Data    : in  std_logic_vector (STREAM_DATA_BITS - 1 downto 0);
18    In_Meta    : in  std_logic_vector (STREAM_META_BITS - 1 downto 0);
19    In_Ack     : out std_logic;
20
21    Out_Valid  : out std_logic;
22    Out_IsKey  : out std_logic;
23    Out_Data   : out std_logic_vector (STREAM_DATA_BITS - 1 downto 0);
24    Out_Meta   : out std_logic_vector (STREAM_META_BITS - 1 downto 0);
25    Out_Ack    : in  std_logic
26  );
27 end entity;
```

Source file: [sort/sortnet/sortnet_Stream_Adapter.vhdl](#)

PoC.sort.sortnet.Stream_Adapter2

Todo: No documentation available.

Entity Declaration:

```

1 entity sortnet_Stream_Adapter2 is
2   generic (
3     STREAM_DATA_BITS      : positive      := 32;
4     STREAM_META_BITS      : positive      := 2;
5     DATA_COLUMNS         : positive      := 2;
6     SORTNET_IMPL          : T_SORTNET_IMPL := SORT_SORTNET_IMPL_ODDEVEN_MERGESORT;
7     SORTNET_SIZE          : positive      := 32;
8     SORTNET_KEY_BITS      : positive      := 32;
9     SORTNET_DATA_BITS     : natural       := 32;
10    SORTNET_REG_AFTER      : natural       := 2;
11    MERGENET_STAGES        : positive      := 2
12  );
13  port (
```

(continues on next page)

(continued from previous page)

```

14   Clock      : in  std_logic;
15   Reset      : in  std_logic;
16
17   Inverse     : in  std_logic      := '0';
18
19   In_Valid    : in  std_logic;
20   In_Data     : in  std_logic_vector (STREAM_DATA_BITS - 1 downto 0);
21   In_Meta     : in  std_logic_vector (STREAM_META_BITS - 1 downto 0);
22   In_SOF      : in  std_logic;
23   In_IsKey    : in  std_logic;
24   In_EOF      : in  std_logic;
25   In_Ack      : out std_logic;
26
27   Out_Valid   : out std_logic;
28   Out_Data    : out std_logic_vector (STREAM_DATA_BITS - 1 downto 0);
29   Out_Meta    : out std_logic_vector (STREAM_META_BITS - 1 downto 0);
30   Out_SOF     : out std_logic;
31   Out_IsKey   : out std_logic;
32   Out_EOF     : out std_logic;
33   Out_Ack     : in  std_logic
34 );
35 end entity;

```

Source file: `sort/sortnet/sortnet_Stream_Adapter2.vhdl`

PoC.sort.sortnet.Transform

Todo: No documentation available.

Entity Declaration:

```

1  entity sortnet_Transform is
2    generic (
3      ROWS          : positive      := 16;
4      COLUMNS      : positive      := 4;
5      DATA_BITS    : positive      := 8
6    );
7    port (
8      Clock         : in  std_logic;
9      Reset         : in  std_logic;
10
11      In_Valid      : in  std_logic;
12      In_Data       : in  T_SLM(ROWS - 1 downto 0, DATA_BITS - 1 downto 0);
13      In_SOF        : in  std_logic;
14      In_EOF        : in  std_logic;
15
16      Out_Valid     : out std_logic;
17      Out_Data      : out T_SLM(COLUMNS - 1 downto 0, DATA_BITS - 1 downto 0);
18      Out_SOF       : out std_logic;
19      Out_EOF       : out std_logic
20    );
21 end entity;

```

Source file: `sort/sortnet/sortnet_Transform.vhdl`

7.14.2 PoC.sort.ExpireList

Todo: No documentation available.

Entity Declaration:

Source file: `sort/sort_ExpireList.vhdl`

7.14.3 PoC.sort.InsertSort

Todo: No documentation available.

Entity Declaration:

Source file: `sort/sort_InsertSort.vhdl`

7.14.4 PoC.sort.LeastFrequentlyUsed

Todo: No documentation available.

Entity Declaration:

Source file: `sort/sort_LeastFrequentlyUsed.vhdl`

7.14.5 PoC.sort.lru_cache

This is an optimized implementation of `sort_lru_list` to be used for caches. Only keys are stored within this list, and these keys are the index of the cache lines. The list initially contains all indices from 0 to `ELEMENTS-1`. The least-recently used index `KeyOut` is always valid.

The first outputted least-recently used index will be `ELEMENTS-1`.

The inputs `Insert`, `Free`, `KeyIn`, and `Reset` are synchronous to the rising-edge of the clock `clock`. All control signals are high-active.

Supported operations:

- **Insert:** Mark index `KeyIn` as recently used, e.g., when a cache-line was accessed.
- **Free:** Mark index `KeyIn` as least-recently used. Apply this operation, when a cache-line gets invalidated.

Entity Declaration:

```

1 entity sort_lru_cache is
2   generic (
3     ELEMENTS      : positive      := 32
4   );
5   port (
6     Clock      : in std_logic;
7     Reset      : in std_logic;
8
9     Insert     : in std_logic;
10    Free        : in std_logic;
11    KeyIn       : in std_logic_vector(log2ceilnz(ELEMENTS) - 1 downto 0);
12
13    KeyOut      : out std_logic_vector(log2ceilnz(ELEMENTS) - 1 downto 0)
14  );
15 end entity;

```

Source file: `sort/sort_lru_cache.vhdl`

7.14.6 PoC.sort.lru_list

List storing (key, value) pairs. The least-recently inserted pair is outputted on DataOut if Valid = '1'. If Valid = '0', then the list empty.

The inputs Insert, Remove, DataIn, and Reset are synchronous to the rising-edge of the clock clock. All control signals are high-active.

Supported operations:

- **Insert:** Insert DataIn as recently used (key, value) pair. If key is already within the list, then the corresponding value is updated and the pair is moved to the recently used position.
- **Remove:** Remove (key, value) pair with the given key. The list is not modified if key is not within the list.

Entity Declaration:

```

1 entity sort_lru_list is
2   generic (
3     ELEMENTS      : positive      := 16;
4     KEY_BITS       : positive      := 4;
5     DATA_BITS     : positive      := 8;
6     INITIAL_ELEMENTS : T_SLM      := (0 to 15 => (0_
7   =>to 7 => '0'));
8     INITIAL_VALIDS  : std_logic_vector := (0 to 15 => '0')
9   );
10  port (
11    Clock      : in std_logic;
12    Reset      : in std_logic;
13
14    Insert     : in std_logic;
15    Remove     : in std_logic;
16    DataIn     : in std_logic_vector(DATA_BITS - 1 downto 0);
17
18    Valid      : out std_logic;
19    DataOut    : out std_logic_vector(DATA_BITS - 1 downto 0)
20  );
21 end entity;

```

Source file: `sort/sort_lru_list.vhdl`

7.15 PoC.xil

This namespace is for Xilinx specific modules.

Sub-Namespaces

- *PoC.xil.mig*
- *PoC.xil.reconfig*

Entities

- *PoC.xil.BSCAN*
- *PoC.xil.ChipScopeICON*
- *PoC.xil.DRP_BusMux*
- *PoC.xil.DRP_BusSync*
- *PoC.xil.ICAP*
- *PoC.xil.Reconfigurator*
- *PoC.xil.SystemMonitor*
- IP:xil_SystemMonitor_Virtex6
- IP:xil_SystemMonitor_Series7

7.15.1 PoC.xil.mig

The namespace `PoC.xil.mig` offers pre-configured memory controllers generated with Xilinx's Memory Interface Generator (MIG).

- **for Spartan-6 boards:**
 - *mig_Atlys_1x128* - A DDR2 memory controller for the Digilent Atlys board.
- **for Kintex-7 boards:**
 - *mig_KC705_MT8JTF12864HZ_1G6* - A DDR3 memory controller for the Xilinx KC705 board.
- **for Virtex-7 boards:**

mig_Atlys_1x128

This DDR2 memory controller is pre-configured for the Digilent Atlys development board. The board is equipped with a single 1 GiBit DDR2 memory chip (128 MiByte) from MIRA (MIRA P3R1GE3EGF G8E DDR2).

Run the following two steps to create the IP core:

1. Generate the source files from the IP core using Xilinx MIG and afterwards patch them

```
PS> .\poc.ps1 coregen PoC.xil.mig.Atlys_1x128 --board=Atlys
```
2. Compile the patched sources into a ready to use netlist (*.ngc) and constraint file (*.ucf)

```
PS> .\poc.ps1 xst PoC.xil.mig.Atlys_1x128 --board=Atlys
```

See also:

Using PoC -> Synthesis For how to run Core Generator and XST from PoC.

mig_KC705_MT8JTF12864HZ_1G6

This DDR2 memory controller is pre-configured for the Xilinx KC705 development board. The board is equipped with a single 1 GiBit DDR3 memory chip (128 MiByte) from Micron Technology (MT8JTF12864HZ-1G6G1).

Run the following two steps to create the IP core:

1. Generate the source files from the IP core using Xilinx MIG and afterwards patch them PS> .\poc.ps1 coregen PoC.xil.mig.KC705_MT8JTF12864HZ_1G6 --board=KC705
2. Compile the patched sources into a ready to use netlist (*.ngc) and constraint file (*.ucf) PS> .\poc.ps1 xst PoC.xil.mig.KC705_MT8JTF12864HZ_1G6 --board=KC705

See also:

Using PoC -> Synthesis For how to run Core Generator and XST from PoC.

7.15.2 PoC.xil.reconfig

These are reconfig entities. ...

Entities

- *PoC.xil.reconfig.icap_fsm*
- *PoC.xil.reconfig.icap_wrapper*

PoC.xil.reconfig.icap_fsm

This module parses the data stream to the Xilinx “Internal Configuration Access Port” (ICAP) primitives to generate control signals. Tested on:

- Virtex-6
- Virtex-7

Entity Declaration:

```

1  entity reconfig_icap_fsm is
2      port (
3          clk          : in  std_logic;
4          reset        : in  std_logic;          -- high-active reset
5          -- interface to connect to the icap
6          icap_in      : out std_logic_vector(31 downto 0); -- data that will go into_
7          --the icap
8          icap_out     : in  std_logic_vector(31 downto 0); -- data from the icap
9          icap_csb     : out std_logic;
10         icap_rw      : out std_logic;
11
12         -- data interface, no internal fifos
13         in_data       : in  std_logic_vector(31 downto 0); -- new configuration data
14         in_data_valid : in  std_logic;          -- input data is valid
15         in_data_rden  : out std_logic;          -- possible to send data
16         out_data      : out std_logic_vector(31 downto 0); -- data read from the fifo
17         out_data_valid : out std_logic;          -- data from icap is valid
18         out_data_full : in  std_logic;          -- receiving buffer is full, halt_
19         --icap
20
21         -- control structures
22         status        : out std_logic_vector(31 downto 0) -- status vector
23     );
24 end reconfig_icap_fsm;

```

Source file: `xil/reconfig/reconfig_icap_fsm.vhdl`

PoC.xil.reconfig.icap_wrapper

This module was designed to connect the Xilinx “Internal Configuration Access Port” (ICAP) to a PCIe endpoint on a Dini board. Tested on:

tbd

Entity Declaration:

```
1 entity reconfig_icap_wrapper is
2   generic (
3     MIN_DEPTH_OUT      : positive := 256;
4     MIN_DEPTH_IN       : positive := 256
5   );
6   port (
7     clk                : in  std_logic;
8     reset              : in  std_logic;
9     clk_icap           : in  std_logic;    -- clock signal for ICAP, max 100 MHz (double_
↳check with manual)
10
11     icap_busy          : out std_logic;    -- the ICAP is processing the data
12     icap_readback      : out std_logic;    -- high during a readback
13     icap_partial_res   : out std_logic;    -- high during reconfiguration
14
15     -- data in
16     write_put          : in  std_logic;
17     write_full         : out std_logic;
18     write_data         : in  std_logic_vector(31 downto 0);
19     write_done         : in  std_logic;    -- high pulse/edge after all data was written
20
21     -- data out
22     read_got           : in  std_logic;
23     read_valid         : out std_logic;
24     read_data          : out std_logic_vector(31 downto 0)
25   );
26 end reconfig_icap_wrapper;
```

Source file: `xil/reconfig/reconfig_icap_wrapper.vhdl`

7.15.3 PoC.xil Package

This package holds all component declarations for this namespace.

Source file: `xil.pkg.vhdl`

7.15.4 PoC.xil.BSCAN

This module wraps Xilinx “Boundary Scan” (JTAG) primitives in a generic module. Supported devices are:

- Spartan-3, Spartan-6
- Virtex-5, Virtex-6
- Series-7 (Artix-7, Kintex-7, Virtex-7, Zynq-7000)

Entity Declaration:

```

1  entity xil_BSCAN is
2      generic (
3          JTAG_CHAIN          : natural;
4          DISABLE_JTAG       : boolean    := FALSE
5      );
6      port (
7          Reset               : out std_logic;
8          RunTest             : out std_logic;
9          Sel                 : out std_logic;
10         Capture             : out std_logic;
11         drck                 : out std_logic;
12         Shift               : out std_logic;
13         Test_Clock          : out std_logic;
14         Test_DataIn         : out std_logic;
15         Test_DataOut        : in  std_logic;
16         Test_ModeSelect     : out std_logic;
17         Update              : out std_logic
18     );
19 end entity;

```

Source file: `xil/xil_BSCAN.vhdl`

7.15.5 PoC.xil.ChipScopeICON

This module wraps 15 ChipScope ICON IP core netlists generated from ChipScope ICON xco files. The generic parameter `PORTS` selects the appropriate ICON instance with 1 to 15 ICON ControlBus ports. Each ControlBus port is of type `T_XIL_CHIPSCOPE_CONTROL` and of mode `inout`.

Compile required CoreGenerator IP Cores to Netlists with PoC

Please use the provided Xilinx ISE compile command `ise` in PoC to recreate the needed source and netlist files on your local machine.

```

cd PoCRoot
.\poc.ps1 ise PoC.xil.ChipScopeICON --board=KC705

```

Entity Declaration:

```

1  entity xil_ChipScopeICON is
2      generic (
3          PORTS              : positive
4      );
5      port (
6          ControlBus        : inout T_XIL_CHIPSCOPE_CONTROL_VECTOR(PORTS - 1 downto 0)
7      );
8  end entity;

```

See also:

Using PoC -> Synthesis For how to run synthesis with PoC and CoreGenerator.

Source file: `xil/xil_ChipScopeICON.vhdl`

7.15.6 PoC.xil.DRP_BusMux

Todo: No documentation available.

Entity Declaration:

Source file: `xil/xil_DRP_BusMux.vhdl`

7.15.7 PoC.xil.DRP_BusSync

Todo: No documentation available.

Entity Declaration:

Source file: `xil/xil_DRP_BusSync.vhdl`

7.15.8 PoC.xil.ICAP

This module wraps Xilinx “Internal Configuration Access Port” (ICAP) primitives in a generic module. Supported devices are:

- Spartan-6
- Virtex-4, Virtex-5, Virtex-6
- Series-7 (Artix-7, Kintex-7, Virtex-7, Zynq-7000)

Entity Declaration:

```
1  entity xil_ICAP is
2      generic (
3          ICAP_WIDTH  : string := "X32";           -- Specifies the input and output data_
↳width to be used
4
5                  -- Spartan 6: fixed to 16 bit
6                  -- Virtex 4:  X8 or X32
7                  -- Rest:  X8, X16, X32
8          DEVICE_ID   : bit_vector := X"1234567";  -- pre-programmed Device ID value_
↳for simulation
9
10                 -- supported by Spartan 6, Virtex 6 and above
11          SIM_CFG_FILE_NAME : string := "NONE"      -- Raw Bitstream (RBT) file to be_
↳parsed by the simulation model
12
13                 -- supported by Spartan 6, Virtex 6 and above
14      );
15      port (
16          clk      : in std_logic;                -- up to 100 MHz (Virtex-6 and above, Virtex-
↳5??)
17          disable  : in std_logic;                -- low active enable -> high active disable
18          rd_wr    : in std_logic;                -- 0 - write, 1 - read
19          busy     : out std_logic;                -- on Series-7 devices always '0'
20          data_in  : in std_logic_vector(31 downto 0); -- on Spartan-6 only 15 downto 0
21          data_out : out std_logic_vector(31 downto 0); -- on Spartan-6 only 15 downto 0
```

(continues on next page)

(continued from previous page)

```

19     );
20 end entity;

```

Source file: xil/xil_ICAP.vhdl

7.15.9 PoC.xil.Reconfigurator

Many complex primitives in a Xilinx device offer a Dynamic Reconfiguration Port (DRP) to reconfigure a primitive at runtime without reconfiguring the whole FPGA.

This module is a DRP master that can be pre-configured at compile time with different configuration sets. The configuration sets are mapped into a ROM. The user can select a stored configuration with `ConfigSelect`. Sending a strobe to `Reconfig` will start the reconfiguration process. The operation completes with another strobe on `ReconfigDone`.

Entity Declaration:

```

1  entity xil_Reconfigurator is
2      generic (
3          DEBUG                : boolean                := FALSE;
4          CLOCK_FREQ           : FREQ                   := 100 MHz;
5          CONFIG_ROM           : in  T_XIL_DRP_CONFIG_ROM := (0 downto 0 => C_XIL_DRP_CONFIG_
6          SET_EMPTY)          --
7      );
8      port (
9          Clock                : in  std_logic;
10         Reset                 : in  std_logic;
11         Reconfig              : in  std_logic;
12         ReconfigDone          : out std_logic;
13         ConfigSelect           : in  std_logic_vector(log2ceilnz(CONFIG_ROM'length) - 1_
14         downto 0);
15         DRP_en                : out std_logic;
16         DRP_Address            : out T_XIL_DRP_ADDRESS;
17         DRP_we                 : out std_logic;
18         DRP_DataIn             : in  T_XIL_DRP_DATA;
19         DRP_DataOut            : out T_XIL_DRP_DATA;
20         DRP_Ack                : in  std_logic
21     );
22 end entity;

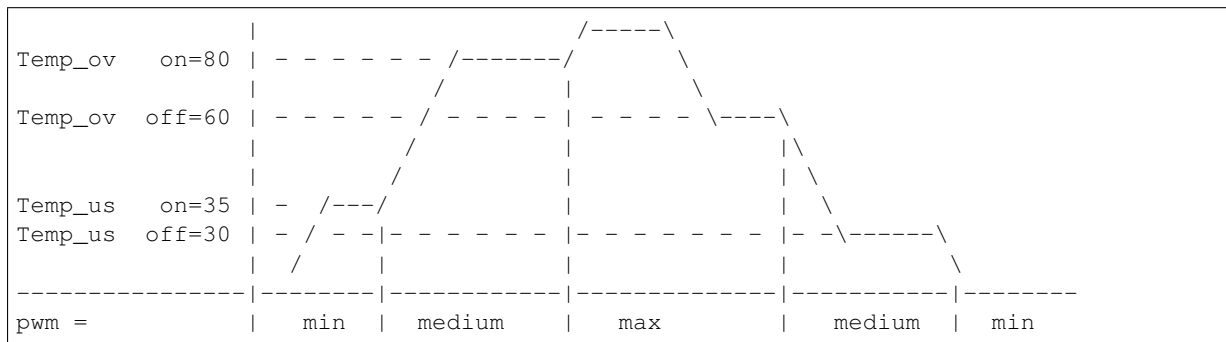
```

Source file: xil/xil_Reconfigurator.vhdl

7.15.10 PoC.xil.SystemMonitor

This module wraps a `SYSMON` or `XADC` to report if preconfigured temperature values are overrun. The `XADC` was formerly known as “System Monitor”.

Temperature Curve



Entity Declaration:

```

1  entity xil_SystemMonitor is
2      port (
3          Reset                : in  std_logic;           -- Reset signal for the System_
4          Monitor control logic
5          Alarm_UserTemp       : out std_logic;           -- Temperature-sensor alarm output
6          Alarm_OverTemp       : out std_logic;           -- Over-Temperature alarm output
7          Alarm                : out std_logic;           -- OR'ed output of all the Alarms
8          VP                   : in  std_logic;           -- Dedicated Analog Input Pair
9          VN                   : in  std_logic;
10     );
11 end entity;

```

Source file: xil/xil_SystemMonitor.vhdl

Third Party Libraries

The PoC-Library is shipped with different third party libraries, which are located in the `<PoCRoot>/lib/` folder. This document lists all these libraries, their websites and licenses.

8.1 Cocotb

`Cocotb` is a coroutine based cosimulation library for writing VHDL and Verilog testbenches in Python.

Folder:	<code><PoCRoot>/lib/cocotb/</code>
Copyright:	Copyright © 2013, Potential Ventures Ltd., SolarFlare Communications Inc.
License:	<i>Revised BSD License (local copy)</i>
Documentation:	http://cocotb.readthedocs.org/
Source:	https://github.com/potentialventures/cocotb

8.2 OSVVM

Open Source VHDL Verification Methodology (OS-VVM) is an intelligent testbench methodology that allows mixing of “Intelligent Coverage” (coverage driven randomization) with directed, algorithmic, file based, and constrained random test approaches. The methodology can be adopted in part or in whole as needed. With OSVVM you can add advanced verification methodologies to your current testbench without having to learn a new language or throw out your existing testbench or testbench models.

Folder:	<code><PoCRoot>/lib/osvmm/</code>
Copyright:	Copyright © 2012-2016 by SynthWorks Design Inc.
License:	<i>Artistic License 2.0 (local copy)</i>
Website:	http://osvmm.org/
Source:	https://github.com/JimLewis/OSVVM

8.3 UVVM

The Open Source **UVVM (Universal VHDL Verification Methodology) - VVC (VHDL Verification Component) Framework** for making structured VHDL testbenches for verification of FPGA. UVVM consists currently of: Utility Library, VVC Framework and Verification IPs (VIP) for various protocols.

For what do I need this VVC Framework? The VVC Framework is a VHDL Verification Component system that allows multiple interfaces on a DUT to be stimulated/handled simultaneously in a very structured manner, and controlled by a very simple to understand software like a test sequencer. VVC Framework is unique as an open source VHDL approach to building a structured testbench architecture using Verification components and a simple protocol to access these. As an example a simple command like `uart_expect (UART_VVCT, my_data)`, or `axilite_write (AXILITE_VVCT, my_addr, my_data, my_message)` will automatically tell the respective VVC (for UART or AXI-Lite) to execute the `uart_receive()` or `axilite_write()` BFM respectively.

Folder:	<PoCRoot>\lib\uvvm\
Copyright:	Copyright © 2016 by Bitvis AS
License:	<i>The MIT License (local copy)</i>
Website:	http://bitvis.no/
Source:	https://github.com/UVVM/UVVM_All

8.4 VUnit

VUnit is an open source unit testing framework for VHDL released under the terms of *Mozilla Public License, v. 2.0*. It features the functionality needed to realize continuous and automated testing of your VHDL code. VUnit doesn't replace but rather complements traditional testing methodologies by supporting a "test early and often" approach through automation.

Folder:	<PoCRoot>\lib\vunit\
Copyright:	Copyright © 2014-2016, Lars Asplund lars.anders.asplund@gmail.com
License:	<i>Mozilla Public License, Version 2.0 (local copy)</i>
Website:	https://vunit.github.io/
Source:	https://github.com/VUnit/vunit

8.5 Updating Linked Git Submodules

The third party libraries are embedded as Git submodules. So if the PoC-Library was not cloned with option `--recursive` it's required to run the sub-module initialization manually:

8.5.1 On Linux

```
cd PoCRoot
git submodule init
git submodule update
```

We recommend to rename the default remote repository name from 'origin' to 'github'.

```
cd PoCRoot\lib\
```

Todo: write Bash code for Linux

8.5.2 On OS X

Please see the Linux instructions.

8.5.3 On Windows

```
cd PoCRoot
git submodule init
git submodule update
```

We recommend to rename the default remote repository name from ‘origin’ to ‘github’.

```
cd PoCRoot\lib\
foreach($dir in (dir -Directory)) {
    cd $dir
    git remote rename origin github
    cd ..
}
```


9.1 IP Core Constraint Files

- fifo
- misc
 - sync
- net
 - eth

9.1.1 fifo

- fifo_ic_got

fifo_ic_got

9.1.2 misc

- sync

sync

- sync_Bits
- sync_Reset
- sync_Vector
- sync_Command

`fifo_ic_got`

`fifo_ic_got`

`fifo_ic_got`

`fifo_ic_got`

9.1.3 net

- eth

`eth`

- eth_RSLayer_GMII_GMII_KC705
- eth_RSLayer_GMII_GMII_ML505
- eth_RSLayer_GMII_GMII_ML605

`eth_RSLayer_GMII_GMII_KC705`

`eth_RSLayer_GMII_GMII_ML505`

`eth_RSLayer_GMII_GMII_ML605`

9.2 Board Constraint Files

- Altera Boards
 - Cyclone III
 - Stratix IV
 - Stratix V
- Lattice Boards
- Xilinx Boards
 - Artix-7
 - Kintex-7
 - Spartan-3 Boards
 - Spartan-6 Boards
 - Virtex-5
 - Virtex-6
 - Virtex-7
 - Zynq-7000

9.2.1 Altera

- Cyclone III * DE0 * DE0 nano
- Stratix IV * DE4
- Stratix V * DE5

Cyclone III

- DE0
- DE0 nano

ECP5 Versa

ECP5 Versa

Stratix IV

- DE4

DE4

Stratix V

- DE5

DE5

9.2.2 Lattice

- ECP5 * ECP5 Versa

ECP5

- ECP5 Versa

ECP5 Versa

9.2.3 Xilinx

- **Spartan-3 Boards**
 - Spartan-3 Starter Kit (S3SK)
 - Spartan-3E Starter Kit (S3ESK)
- **Spartan-6 Boards**
 - Atlys
- **Artix-7**
 - AC701
- **Kintex-7**

- KC705
- **Virtex-5**
 - ML505
 - ML506
 - XUPV5
- **Virtex-6**
 - ML605
- **Virtex-7**
 - VC707
- **Zynq-7000**
 - ZC706
 - ZedBoard

Spartan-3

- Spartan-3 Starter Kit (S3SK)
- Spartan-3E Starter Kit (S3ESK)

S3SK

S3ESK

Spartan-6

- Atlys

Atlys

Artix-7

- AC701

AC701

Kintex-7

- KC705

KC705

Virtex-5

- ML505
- ML506
- XUPV5

ML505

ML506

XUPV5

Virtex-6

- ML605

ML605

Virtex-7

- VC707

VC707

Zynq-7000

- ZC706
- ZedBoard

ZC706

ZedBoard

CHAPTER 10

Tool Chain Specifics

Attention: This page is under construction.

10.1 Aldec Active-HDL

Todo:

- No GUI mode supported
 - VHDL-2008 parser bug in Active-HDL 10.3
-

10.2 Mentor QuestaSim

Special feature: embedded poc prodecures to recompile relaunch, rerun and save waveforms. . .

10.3 Xilinx ISE

- Describe the `use_new_parser yes` option

10.4 Xilinx Vivado

- Describe the `vivado` branch (Git).

CHAPTER 11

Examples

Note: Under construction.

PoC-Exmaples repository on GitHub.

Part III

References

Command Reference

This is the command line option reference for all provided scripts (Bash, PowerShell, Perl) and programs (Python) shipped with PoC.

12.1 PoC Wrapper Scripts

The PoC main program **PoC.py** requires a prepared environment, which needs to be setup by platform specific wrapper scripts written as shell scripts language (PowerShell/Bash). Moreover, the main program requires a supported Python version, so the wrapper script will search the best matching language environment.

The wrapper script offers the ability to hook in user-defined scripts to prepared (before) and clean up the environment (after) a PoC execution. E.g. it's possible to load the environment variable `LM_LICENSE_FILE` for the FlexLM license manager.

Created Environment Variables

PoCRootDirectory

The path to PoC's root directory.

12.1.1 poc.ps1

`PoC.ps1` is the wrapper for the Windows platform using a PowerShell script. It can be debugged by adding the command line switch `-D`. All parameters are passed to `PoC.py`.

-D

Enabled debug mode in the wrapper script.

Other arguments

All remaining arguments are passed to `PoC.py`.

12.1.2 poc.sh

`PoC.sh` is the wrapper for Linux and Unix platforms using a Bash script. It can be debugged by adding the command line switch `-D`. All parameters are passed to `PoC.py`.

-D

Enabled debug mode in the wrapper script.

Other arguments

All remaining arguments are passed to `PoC.py`.

12.2 Main Program (`PoC.py`)

The main program `PoC.py` expects the environment variable `PoCRootDirectory` to be set.

12.3 Pre-compile Scripts

The following scripts can be used to pre-compile vendor's primitives or third party libraries. Pre-compile vendor primitives are required for vendor specific simulations or if no generic IP core implementation is available. Third party libraries are usually used as simulation helpers and thus needed by many testbenches.

The pre-compiled packages and libraries are stored in the directory `/temp/precompiled/`. Per simulator, one `<simulator>/` sub-directory is created. Each simulator directory in turn contains library directories, which may be grouped by the library vendor's name: `[<vendor>/]<library>/`.

So for example: *OSVVM* pre-compiled with GHDL is stored in `/temp/precompiled/ghdl/osvvm/`. Note OSVVM is a single library and thus no vendor directory is used to group the generated files. GHDL will also create VHDL language revision sub-directories like `v93/` or `v08/`.

Currently the provided scripts support 2 simulator targets and one combined target:

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

The GHDL simulator distinguishes various VHDL language revisions and thus can pre-compile the source for these language revisions into separate output directories. The command line switch `-All/--all` will build the libraries for all major VHDL revisions (93, 2008).

Pre-compile Altera Libraries

12.3.1 compile-altera.sh

This script pre-compiles the Altera primitives. This script will generate all outputs into a `altera` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- help**
Show the embedded help page(s).
- clean**
Clean up directory before analyzing.
- all**
Pre-compile all libraries and packages for all simulators.
- ghdl**
Pre-compile the Altera Quartus libraries for GHDL.
- questa**
Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

- vhd193**
For GHDL only: Set VHDL Standard to '93.
- vhd12008**
For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Altera.

12.3.2 compile-altera.ps1

This script pre-compiles the Altera primitives. This script will generate all outputs into a `altera` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- Help**
Show the embedded help page(s).
- Clean**
Clean up directory before analyzing.

-All

Pre-compile all libraries and packages for all simulators.

-GHDL

Pre-compile the Altera Quartus libraries for GHDL.

-Questa

Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

-VHDL93

For GHDL only: Set VHDL Standard to '93.

-VHDL2008

For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Altera.

Pre-compile Lattice Libraries

12.3.3 compile-lattice.sh

This script pre-compiles the Lattice primitives. This script will generate all outputs into a `lattice` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

--help

Show the embedded help page(s).

--clean

Clean up directory before analyzing.

--all

Pre-compile all libraries and packages for all simulators.

--ghdl

Pre-compile the Altera Quartus libraries for GHDL.

--questa

Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL**--vhd193**

For GHDL only: Set VHDL Standard to '93.

--vhd12008

For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Lattice.

12.3.4 compile-lattice.ps1

This script pre-compiles the Lattice primitives. This script will generate all outputs into a `lattice` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options**-Help**

Show the embedded help page(s).

-Clean

Clean up directory before analyzing.

-All

Pre-compile all libraries and packages for all simulators.

-GHDL

Pre-compile the Altera Quartus libraries for GHDL.

-Questa

Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL**-VHDL93**

For GHDL only: Set VHDL Standard to '93.

-VHDL2008

For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Lattice.

Pre-compile OSVVM Libraries

12.3.5 compile-osvvm.sh

This script pre-compiles the OSVVM packages. This script will generate all outputs into a `osvvm` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

--help

Show the embedded help page(s).

--clean

Clean up directory before analyzing.

--all

Pre-compile all libraries and packages for all simulators.

--ghdl

Pre-compile the Altera Quartus libraries for GHDL.

--questa

Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

--vhd193

For GHDL only: Set VHDL Standard to '93.

--vhd12008

For GHDL only: Set VHDL Standard to '08.

12.3.6 compile-osvvm.ps1

This script pre-compiles the OSVVM packages. This script will generate all outputs into a `osvvm` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

-Help

Show the embedded help page(s).

-Clean

Clean up directory before analyzing.

-All

Pre-compile all libraries and packages for all simulators.

-GHDL

Pre-compile the Altera Quartus libraries for GHDL.

-Questa

Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

-VHDL93

For GHDL only: Set VHDL Standard to '93.

-VHDL2008

For GHDL only: Set VHDL Standard to '08.

Pre-compile UVVM Libraries

12.3.7 compile-uvvm.sh

This script pre-compiles the UVVM framework. This script will generate all outputs into a `uvvm` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

--help

Show the embedded help page(s).

--clean

Clean up directory before analyzing.

- all**
Pre-compile all libraries and packages for all simulators.
- ghdl**
Pre-compile the Altera Quartus libraries for GHDL.
- questa**
Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

- vhd193**
For GHDL only: Set VHDL Standard to '93.
- vhd12008**
For GHDL only: Set VHDL Standard to '08.

12.3.8 compile-uvvm.ps1

This script pre-compiles the UVVM framework. This script will generate all outputs into a `uvvm` directory.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- Help**
Show the embedded help page(s).
- Clean**
Clean up directory before analyzing.
- All**
Pre-compile all libraries and packages for all simulators.
- GHDL**
Pre-compile the Altera Quartus libraries for GHDL.
- Questa**
Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

- VHDL93**
For GHDL only: Set VHDL Standard to '93.
- VHDL2008**
For GHDL only: Set VHDL Standard to '08.

Pre-compile Xilinx ISE Libraries

12.3.9 compile-xilinx-ise.sh

This script pre-compiles the Xilinx primitives. Because Xilinx offers two tool chains (ISE, Vivado), this script will generate all outputs into a `xilinx-ise` directory and a symlink to `xilinx` will be created. This eases the coexistence of pre-compiled primitives from ISE and Vivado.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- help**
Show the embedded help page(s).
- clean**
Clean up directory before analyzing.
- all**
Pre-compile all libraries and packages for all simulators.
- ghdl**
Pre-compile the Altera Quartus libraries for GHDL.
- questa**
Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

- vhd193**
For GHDL only: Set VHDL Standard to '93.
- vhd12008**
For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted SecureIP primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Xilinx.

12.3.10 compile-xilinx-ise.ps1

This script pre-compiles the Xilinx primitives. Because Xilinx offers two tool chains (ISE, Vivado), this script will generate all outputs into a `xilinx-ise` directory and a symlink to `xilinx` will be created. This eases the coexistence of pre-compiled primitives from ISE and Vivado. The symlink can be changed by the user or via `-ReLink`.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- Help**
Show the embedded help page(s).
- Clean**
Clean up directory before analyzing.
- All**
Pre-compile all libraries and packages for all simulators.
- GHDL**
Pre-compile the Altera Quartus libraries for GHDL.
- Questa**
Pre-compile the Altera Quartus libraries for QuestaSim.
- ReLink**
Change the 'xilinx' symlink to 'xilinx-ise'.

Additional Options for GHDL

- VHDL93**
For GHDL only: Set VHDL Standard to '93.
- VHDL2008**
For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted SecureIP primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Xilinx.

Pre-compile Xilinx Vivado Libraries

12.3.11 compile-xilinx-vivado.sh

This script pre-compiles the Xilinx primitives. Because Xilinx offers two tool chains (ISE, Vivado), this script will generate all outputs into a `xilinx-vivado` directory and a symlink to `xilinx` will be created. This eases the coexistence of pre-compiled primitives from ISE and Vivado.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- help**
Show the embedded help page(s).
- clean**
Clean up directory before analyzing.
- all**
Pre-compile all libraries and packages for all simulators.
- ghdl**
Pre-compile the Altera Quartus libraries for GHDL.
- questa**
Pre-compile the Altera Quartus libraries for QuestaSim.

Additional Options for GHDL

- vhd193**
For GHDL only: Set VHDL Standard to '93.
- vhd12008**
For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted SecureIP primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Xilinx.

12.3.12 compile-xilinx-vivado.ps1

This script pre-compiles the Xilinx primitives. Because Xilinx offers two tool chains (ISE, Vivado), this script will generate all outputs into a `xilinx-vivado` directory and a symlink to `xilinx` will be created. This eases the coexistence of pre-compiled primitives from ISE and Vivado. The symlink can be changed by the user or via `-ReLink`.

Supported Simulators

Target	Description
All	pre-compile for all simulators
GHDL	pre-compile for the GHDL simulator
Questa	pre-compile for Metor Graphics QuestaSim

Command Line Options

- Help**
Show the embedded help page(s).
- Clean**
Clean up directory before analyzing.
- All**
Pre-compile all libraries and packages for all simulators.
- GHDL**
Pre-compile the Altera Quartus libraries for GHDL.
- Questa**
Pre-compile the Altera Quartus libraries for QuestaSim.
- ReLink**
Change the 'xilinx' symlink to 'xilinx-vivado'.

Additional Options for GHDL

- VHDL93**
For GHDL only: Set VHDL Standard to '93.
- VHDL2008**
For GHDL only: Set VHDL Standard to '08.

GHDL Notes

Not all primitives and macros are available as plain VHDL source code. Encrypted SecureIP primitives and netlists cannot be pre-compiled by GHDL.

QuestaSim Notes

The pre-compilation for QuestaSim uses a build in program from Xilinx.

Contents of this Page

- *Overview*
- *Database Structure*
 - *Nodes*
 - *References*
 - *Options*
 - *Values*
 - *Value Interpolation*
 - *Node Interpolation*
 - *Root Nodes*
- *Supported Options*
- *Files in detail*
 - *config.structure.ini*
 - *config.entity.ini*
 - *config.boards.ini*
 - *config.private.ini*
- *User Defined Variables*

13.1 Overview

PoC internal IP core database uses INI files and advanced interpolation rules provided by `ExtendedConfigParser`. The database consists of 5 *.ini files:

- **`pyconfig.boards.ini`** This file contains all known *FPGA boards* and *FPGA devices*.

- **pyconfig.defaults.ini** This file contains all default options and values for all supported node types.
- **pyconfig.entity.ini** This file contains all IP cores (entities) and their corresponding testbench or netlist settings.
- **pyconfig.private.ini** This file is created by `.\poc.ps1 configure` and contains settings for the local PoC installation. This file must not be shared with other PoC instances. See [Configuring PoC's Infrastructure](#) on how to configure PoC on a local system.
- **pyconfig.structure.ini** Nodes in this file describe PoC's namespace tree and which IP cores are assigned to which namespace.

Additionally, the database refers to **.files* and **.rules* files. The first file type describes, in an imperative language, which files are needed to compile a simulation or to run a synthesis. The latter file type contains patch instructions per IP core. See [Files Formats](#) for more details.

13.2 Database Structure

The database is stored in multiple *INI files*, which are merged in memory to a single configuration database. Each INI file defines an associative array of *sections* and option lines. The content itself is an associative array of *options* and values. Section names are enclosed in square brackets `[...]` and allow simple case-sensitive strings as names. A section name is followed by its section content, which consists of option lines.

One option is stored per option line and consists of an option name and a value separated by an equal sign `=`. The option name is also a case-sensitive simple string. The value is string, starts after the first non-whitespace character and ends before the newline character at the end of the line. The content can be of any string, except for the newline characters. Support for escape sequences depends on the option usage.

Values containing `${...}` and `%{...}` are raw values, which need to be interpolated by the `ExtendedConfigParser`. See [Value Interpolation](#) and [Node Interpolation](#) for more details.

Sections can have a default section called `DEFAULT`. Options not found in a normal section are looked up in the default section. If found, the value of the matching option name is the lookup result.

Example

```
[section1]
option1 = value1
opt2 =    val ue $2

[section2]
option1 = ${section1:option1}
opt2 =    ${option1}
```

Option lines can be of three kinds: An option, a reference, or a user defined variable. While the syntax is always the same, the meaning is inferred from the context.

Option Line Kind	Distinguishing Characteristic
Reference	The option name is called a (node) reference, if the value of an option is a predefined keyword for the current node class. Because the option's value is a keyword, it cannot be an interpolated value.
Option	The option uses a defined option name valid for the current node class. The value can be a fixed or interpolated string.
User Defined Variable	Otherwise an option line is a user defined variable. It can have fixed or interpolated string values.

```

[PoC]
Name =
Prefix =
arith =      Namespace
bus =      Namespace

[PoC.arith]
addw =      Entity
prng =      Entity

[PoC.bus]
stream =    Namespace
wb =      Namespace
Arbiter =    Entity

[PoC.bus.stream]
Buffer =    Entity
DeMux =     Entity
Mirror =    Entity
Mux =       Entity

[PoC.bus.wb]
fifo_adapter = Entity
ocram_adapter = Entity
uart_wrapper = Entity

```

13.2.1 Nodes

The database is build of nested associative arrays and generated in-memory from 5 *.ini files. This implies that all section names are required to be unique. (Section merging is not allowed.) A fully qualified section name has a prefix and a section name delimited by a dot character. The section name itself can consist of parts also delimited by dot characters. All nodes with the same prefix shape a node class.

The following table lists all used prefixes:

Prefix	Description
INSTALL	A installed tool (chain) or program.
SOLUTION	Registered external solutions / projects.
CONFIG	Configurable PoC settings.
BOARD	A node to describe a known board.
CONST	A node to describe constraint file set for a known board.
PoC	Nodes to describe PoC's namespace structure.
IP	A node describing an IP core.
TB	A node describing testbenches.
COCOTB	A node describing Cocotb testbenches.
CG	A node storing Core Generator settings.
LSE	A node storing settings for LSE based netlist generation.
QMAP	A node storing settings for Quartus based netlist generation.
XST	A node storing settings for XST based netlist generation.
VIVADO	A node storing settings for Vivado based netlist generation.
XCI	A node storing settings for IP Catalog based netlist generation.

The database has 3 special sections without prefixes:

Section Name	Description
PoC	Root node for PoC's namespace hierarchy.
BOARDS	Lists all known boards.
SPECIAL	Section with dummy values. This is needed by synthesis and overwritten at runtime.

Example section names

```
[PoC]
[PoC.arith]
[PoC.bus]
[PoC.bus.stream]
[PoC.bus.wb]
```

The fully qualified section name `PoC.bus.stream` has the prefix `PoC` and the section name `bus.stream`. The section name has two parts: `bus` and `stream`. The dot delimited section name can be considered a path in a hierarchical database. The parent node is `PoC.bus` and its grandparent is `PoC`. (Note this is a special section. See the special sections table from above.)

13.2.2 References

Whatever this is handy to create new field

13.2.3 Options

13.2.4 Values

13.2.5 Value Interpolation

13.2.6 Node Interpolation

13.2.7 Root Nodes

13.3 Supported Options

Note: See `py\config.defaults.ini` for predefined default values (options) and predefined variables, which can be used as a shortcut.

13.4 Files in detail

13.4.1 config.structure.ini

13.4.2 config.entity.ini

13.4.3 config.boards.ini

13.4.4 config.private.ini

13.5 User Defined Variables

14.1 List of Supported FPGA Devices

Vendor	Family	Device Name
Altera	Max	Max-II, Max 10
	Cyclone	Cyclone III, Cyclone V
	Stratix	Stratix II, Stratix IV, Stratix V, Stratix 10
	Arria	Arria II, Arria V
Lattice	Mach	MachXO
	ECP	ECP3, ECP5
Xilinx	Coolrunner	Coolrunner-II
	Spartan	Spartan-3, Spartan-6
	Artix	Artix-7
	Kintex	Kintex-7, Kintex UltraScale, Kintex UltraScale+
	Virtex	Virtex-II, Virtex-4, Virtex-5, Virtex-7, Virtex UltraScale, Virtex UltraScale+
	Zynq	Zynq-7000

14.2 List of Supported Boards

Board Name	Device String	Device Name
GENERIC	GENERIC	Generic board and device
Altera	⇒ DE4	
DE0	EP3C16F484	Altera Cyclone III
S2GXAV	EP2SGX90FF1508C3	Altera Stratix II
DE4	EP4SGX230KF40C2	Altera Stratix IV
DE5	EP5SGXEA7N2F45C2	Altera Stratix V
Lattice	⇒ ECP5Versa	
ECP5Versa	LFE5UM-45F-6BG381C	Lattice ECP5
Xilinx	⇒ KC705	
S3SK200	XC3S200FT256	Xilinx Spartan-3
S3ESK500	XC3S500EFT256	Xilinx Spartan-3
S3SK1000	XC3S1000FT256	Xilinx Spartan-3
S3ESK1600	XC3S1600EFT256	Xilinx Spartan-3
ATLYS	XC6SLX45-3CSG324	Xilinx Spartan-6
ZC706	XC7Z045-2FFG900	Xilinx Zynq-7000
ZedBoard	XC7Z020-1CLG484	Xilinx Zynq-7000
AC701	XC7A200T-2FBG676C	Xilinx Artix-7
KC705	XC7K325T-2FFG900C	Xilinx Kintex-7
ML505	XC5VLX50T-1FF1136	Xilinx Virtex-5
ML506	XC5VSX50T-1FFG1136	Xilinx Virtex-5
ML507	XC5VFX70T-1FFG1136	Xilinx Virtex-5
XUPV5	XC5VLX110T-1FF1136	Xilinx Virtex-5
ML605	XC6VLX240T-1FF1156	Xilinx Virtex-6
VC707	XC7VX485T-2FFG1761C	Xilinx Virtex-7
VC709	XC7VX690T-2FFG1761C	Xilinx Virtex-7
Custom	<any device>	

14.3 Wrapper Script Hook Files

The shell scripts `poc.ps1` and `poc.sh` can be customized though hook files, which are executed before and after a PoC command is executed. The wrapper scripts support 4 kinds of hook files:

- VendorPreHookFile
- ToolPreHookFile
- VendorPostHookFile
- ToolPostHookFile

The wrapper scans the arguments given to the front-end script and searches for known commands. If one is found, the hook files are scheduled before and after the execution of the wrapped executable. The hook files are sourced into the current execution and need to be located in the `./py/Wrapper/Hooks` directory.

A common use case is the preparation of special vendor or tool chain environments. For example many EDA tools are using FlexLM as a license manager, which needs the environments variable `LM_LICENSE_FILE` to be set. A `PreHookFile` can be used to load/export such an environment variable.

14.3.1 Examples

Mentor QuestaSim on Linux:

The PoC infrastructure is called with this command line:

```
./poc.sh -v vsim PoC.arith.prng
```

The `vsim` command is recognized and the following events are scheduled:

1. `source ./py/Wrapper/Hooks/Mentor.pre.sh`
2. `source ./py/Wrapper/Hooks/Mentor.QuestaSim.pre.sh`
3. `Execute ./py/PoC.py -v vsim PoC.arith.prng`
4. `source ./py/Wrapper/Hooks/Mentor.QuestaSim.post.sh`
5. `source ./py/Wrapper/Hooks/Mentor.post.sh`

If a hook files doesn't exist, it's skipped.

Mentor QuestaSim on Windows:

The PoC infrastructure is called with this command line:

```
.\poc.ps1 -v vsim PoC.arith.prng
```

The `vsim` command is recognized and the following events are scheduled:

1. `.\py\Wrapper\Hooks\Mentor.pre.ps1`
2. `.\py\Wrapper\Hooks\Mentor.QuestaSim.pre.ps1`
3. `Execute .\py\PoC.py -v vsim PoC.arith.prng`
4. `.\py\Wrapper\Hooks\Mentor.QuestaSim.post.ps1`
5. `.\py\Wrapper\Hooks\Mentor.post.ps1`

If a hook files doesn't exist, it's skipped.

14.3.2 FlexLM

Many EDA tools require an environment variable called `LM_LICENSE_FILE`. If no other tool settings are required, a common `FlexLM.sh` can be generated. This file is used as a symlink target for each tool specific hook file.

Content of the 'FlexLM.sh' script:

```
export LM_LICENSE_FILE=1234@flexlm.company.com
```

Create symlinks:

```
ln -s FlexLM.sh Altera.Quartus.pre.sh
ln -s FlexLM.sh Mentor.QuestaSim.pre.sh
```

14.4 File Formats

14.4.1 *.ini Format

Document rule:

DocumentLine rule:

Section rule:

OptionLine rule:

FQSectionName rule:

14.4.2 *.files Format

Contents of this Page

- *Document*
- *Source File Statements*
- *Conditional Statements*
- *Boolean Expressions*
 - *Unary operators*
 - *Binary operators*
 - *Literals*
 - *Pre-defined constants*
- *Path Expressions*

Files files are used to ...

Line comments start with #.

Document

Source File Statements

Bla VHDLStatement blub

- `vhdl Library "<VHDLFile>"` This statement references a VHDL source file.
- `verilog "<VerilogFile>"` This statement references a Verilog source file.
- `cocotb "<PythonFile>"` This statement references a Cocotb testbench file (Python file).
- `ucf "<UCFFile>"` This statement references a Xilinx User Constraint File (UCF).
- `sdc "<SDCFile>"` This statement references a Synopsys Design Constraint file (SDC).
- `xdc "<XDCFile>"` This statement references a Xilinx Design Constraint file (XDC).
- `ldc "<LDCFile>"` This statement references a Lattice Design Constraint file (LDC).

Conditional Statements

- `If (<Expression>) Then ... [ElseIf (<Expression>) Then ...][Else ...]`
`End IF` This allows the user to define conditions, when to load a source file into the file list. The `ElseIf` and `Else` clause of an `If` statement are optional.

Boolean Expressions

Unary operators

- `!` - not
- `[...]` - list construction
- `?` - file exists

Binary operators

- `and` - and
- `or` - or
- `xor` - exclusive or
- `in` - in list
- `=` - equal
- `!=` - unequal
- `<` - less than
- `<=` - less than or equal
- `>` - greater than
- `>=` - greater than or equal

Literals

- `<constant>` - a pre-defined constant
- `"<String>"` - Strings are enclosed in quote signs
- `<Integer>` - Integers as decimal values

Pre-defined constants

- Environment Variables:
 - **Environment** Values:
 - * `"Simulation"`
 - * `"Synthesis"`
 - `ToolChain` - The used tool chain. E.g. `"Xilinx_ISE"`
 - `Tool` - The used tool. E.g. `"Mentor_QuestaSim"` or `"Xilinx_XST"`
 - `VHDL` - The used VHDL version. 1987, 1993, 2002, 2008
- Board Variables:
 - `BoardName` - A string. E.g. `"KC705"`
- Device Variables:
 - `DeviceVendor` - The vendor of the device. E.g. `"Altera"`
 - `DeviceDevice` -
 - `DeviceFamily` -
 - `DeviceGeneration` -
 - `DeviceSeries` -

Path Expressions

- / - sub-directory
- & - string concat

Other Statements

- `include "<FilesFile>"` Include another *.files file.
- `library <VHDLLibrary> "<LibraryPath>"` Reference an existing (pre-compiled) VHDL library, which is passed to the simulator, if external libraries are supported.
- `report "<Message>"` Print a critical warning in the log window. This critical warning is treated as an error.

14.4.3 *.rules Format

Contents of this Page

- **.rules Format*
 - *Headline 1*
 - *Headline 2*
 - *Headline 3*

Headline 1

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

Headline 2

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

Headline 3

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet

14.5 Naming Conventions

Todo: Write an introduction paragraph for this page.

14.5.1 Root Directory Overview (PoCRoot)

The PoC-Library is structured into several sub-directories, naming the purpose of the directory like `src` for sources files or `tb` for testbench files. The structure within these directories is most likely the same and based on PoC's *sub-namespace tree*. PoC's installation directory is also referred to as `PoCRoot`.

- **lib** Third party libraries like Coctb, OSVVM or VUnit are shipped in this folder. The external library is stored in a sub directory named like the library. If a library is available as a Git submodule, then it is linked as a submodule for better version tracking.
- **netlist** This is the output directory for pre-configured netlists, synthesized by PoC. Netlists and related constraint files are the result of IP core synthesis flows, either from PoC's source files or from vendor specific IP core files like *.xco files from Xilinx Core Generator. Generated IP cores are stored in device sub-directories, because most netlists formats are device specific. For example the IP core `PoC.arith.prng` created from source file `src\arith\arith_prng.vhdl` generated for a Kintex-7 325T mounted on a KC705 board will be copied to `netlist\XC7K325T-2FFG900\arith\arith_prng.ngc` if Xilinx ISE XST is used for synthesis.
- **py** The supporting Python infrastructure, the configuration files and the IP core 'database' is stored in this directory.
- **sim** Some of PoC's testbenches are shipped with pre-configured waveform views/ waveform configuration files for selected simulators or waveform viewers. If a testbench is launched in GUI mode (`--gui`) and a waveform view for the chosen simulator is found, it's loaded as the default view.
- **src** The source files of PoC's IP cores are stored in this directory. The IP cores are grouped by their sub-namespace into sub-directories according to the *sub-namespace tree*. See the paragraph below, for how IP cores are named and how PoC core names map to the sub-namespace hierarchy and the resulting sub-namespace directory structure.
- **tb** PoC is shipped with testbenches. All testbenches are categorized and stored in sub-directories like the IP core, which is tested.
- **tcl** Supporting Tcl files.
- **temp** A pre-created temporary directors for various tool's intermediate outputs. In case of errors in a used vendor tool or in PoC's infrastructure, this directory contains intermediate files, log files and report files, which can be used to analyze the error.
- **tools** This directory contains miscellaneous files or scripts for external tools like emacs, git or text editor syntax highlighting files.
- **ucf** Pre-configured constraint files (*.ucf, *.xdc, *.sdc) for many FPGA boards, containing physical (pin, placement) and timing constraints.
- **xst** Configuration files to synthesize PoC modules with Xilinx XST into a netlist.

14.5.2 Namespaces and Modules

Namespaces

PoC uses namespaces and sub-namespaces to categorize all VHDL and Verilog modules. Despite VHDL doesn't support sub-namespaces yet, PoC already uses sub-namespaces enforced by a strict naming schema.

Rules: 1. Namespace names are lower-case, underscore free, valid VHDL identifiers. 2. A namespace name is unique, but can be part of a entity name.

Module Names

Module names are prefixed with its parents namespace name. A module name can contain underscores to denote implementation variants of a module.

Rules: 3. Modul names are valid VHDL identifiers prefixed with its parent namespace's name. 4. The first part of module name must not contain the parents namespace name.

Example 1 - PoC.fifo.cc_got

For example a FIFO module with a common clock interface and a *got* semantic is named `PoC.fifo.cc_got` (fully qualified name). This name can be split at every dot and underscore sign, resulting in the following table of name parts:

PoC	fifo	cc	got
Root Namespace	Sub-Namespace	Common Clock Interface	Got Semantic

Because `PoC.fifo.cc_got` refers to an IP core, the source file is located in the `<PoCRoot>\src` directory. The (sub-)namespace of the PoC entity is `fifo`, so it's stored in the sub-directory `fifo`. The file name `cc_got` FIFO is prefixed with the last sub-namespace: In this case `fifo_`. This is summarized in the following table:

Property	Value
Fully Qualified Name	<code>PoC.fifo.cc_got</code>
VHDL entity name	<code>fifo_cc_got</code>
File name	<code>fifo_cc_got.vhdl</code>
IP Core Description File	<code>\src\fifo\fifo_cc_got.files</code>
Source File Location	<code>\src\fifo\fifo_cc_got.vhdl</code>
Testbench Location	<code>\tb\fifo\fifo_cc_got_tb.vhdl</code>
Testbench Description File	<code>\tb\fifo\fifo_cc_got_tb.files</code>
Waveform Description Files	<code>\sim\fifo\fifo_cc_got_tb.*</code>

Other implementation variants are:

- `_dc` – dependent clock / related clock
- `_ic` – independent clock / cross clock
- `_got_tempgot` – got interface extended by a temporary got interface
- `_got_tempput` – got interface extended by a temporary put interface

Example 2 - PoC.mem.ocram.tdp

PoC	mem	ocram	tdp
Root Namespace	Sub-Namespace	Sub-Namespace	True-Dual-Port

Property	Value
Fully Qualified Name	PoC.mem.ocram.tdp
VHDL entity name	ocram_tdp
File name	ocram_tdp.vhdl
IP Core Description File	\src\mem\ocram\ocram_tdp.files
Source File Location	\src\mem\ocram\ocram_tdp.vhdl
Testbench Location	\tb\mem\ocram\ocram_tdp_tb.vhdl
Testbench Description File	\tb\mem\ocram\ocram_tdp_tb.files
Waveform Description Files	\sim\mem\ocram\ocram_tdp_tb.*

Note: Not all sub-namespace parts are include as a prefix in the name, only the last one.

14.5.3 Signal Names

Todo: No documentation available.

14.6 Known Issues

14.6.1 General

Synthesis of tri-state signals

Tri-state signals should be only used when they are connected (through the hierarchy) to top-level bidirectional or output pins.

Descriptions which infer a tri-state driver like:

```
pin <= data when tri = '0' else 'Z';
```

should not be included in any IP core description because these hinder or even inhibit block-based design flows. If a netlist is generated from such an IP core, the netlist may contain only a simple internal (on-chip) tri-state buffer instead of the correct tri-state I/O block primitive because I/O buffers are not automatically added for netlist generation. If the netlist is then used in another design, the mapper, e.g. Xilinx ISE Map, may fail to merge the internal tri-state buffer of the IP core netlist with the I/O buffer automatically created for the top-level netlist. This failing behavior is not considered as a tool bug.

Thus, if tri-state drivers should be included in an IP core, then the IP core description must instantiate the appropriate I/O block primitive of the target architecture like it is done by the Xilinx MIG.

Synthesis of bidirectional records

Records are useful to group several signals of an IP core interface. But the corresponding port of this record type should not be of mode `inout` to pass data in both direction. This restriction holds even if a record member will be driven only by one source in the real hardware and even if all the drivers (one for each record member) are visible to the current synthesis run. The following observations have been made:

- An IP core (entity or procedure) must drive all record members with value 'Z' which are only used as an input in the IP core. If this is missed, then the respective record member will be driven by 'U' and the effective value after resolution will be 'U' as well, see IEEE Std. 1076-2008 para. 12.6.1. Thus simulation will fail.

But these 'Z' drivers will flood the RTL / Netlist view of Altera Quartus-II, Intel Quartus Prime and Lattice Diamond with always tri-stated drivers and make this view unusable.

Note: Simulation with ModelSim shows correct output even when the ‘Z’ driver is missing, but a warning is reported that the behavior is not VHDL Standard compliant.

- Altera Quartus-II and Intel Quartus Prime report warnings about this meaningless ‘Z’ drivers. Synthesis result is as expected if each record member is only driven by one source in real hardware.
- The synthesis result of the Lattice Synthesis Engine (3.7.0 / 3.8.0) is not optimal. It seems that the synthesizer tries to implement the internal (on-chip) tristate bus using AND-OR logic but failed to optimize it away because there was only one real source. Test case was a simple SRAM controller which used the record type `T_IO_TRISTATE` to bring-out the data-bus so that the tri-state driver could be instantiated on the top-level.

Use separate records for the input and output data flow instead.

14.6.2 Aldec Active-HDL

- Aliases to functions and protected type methods

14.6.3 Altera Quartus-II / Intel Quartus Prime

- Generic types of type strings filled with NUL

14.6.4 GHDL

- Aliases to protected type methods

14.6.5 Xilinx ISE

- Shared Variables in Simulation (VHDL-93)

14.6.6 Xilinx Vivado

- Physical types in synthesis
- VHDL-2008 mode in simulation
- Shared variables in simulation (VHDL-93 and VHDL-2008))

14.7 Local License Copies

This documentation contains local copies of all used licenses by either PoC itself or embedded libraries or IP cores. Each formatted¹ license text can be downloaded from the original source. Therefor see the top-most INFO box, which contains a link to the original license file source.

Note: This is a local copy of the [Apache License Version 2.0](#).

¹ Formatted means, that reStructured Text (reST) markup is used to highlight headlines, to display lists as unordered or ordered lists and to emphasis special wording in bold. There is no change in words, spelling or paragraph numbering. The formatting is needed to allow a proper rendering with Sphinx and to create a correct toctree (Table of Content Tree).

14.7.1 Apache License 2.0

Version 2.0, January 2004

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

“License” shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

“Licensor” shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

“Legal Entity” shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, **“control”** means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

“You” (or **“Your”**) shall mean an individual or Legal Entity exercising permissions granted by this License.

“Source” form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

“Object” form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

“Work” shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

“Derivative Works” shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

“Contribution” shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, **“submitted”** means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as **“Not a Contribution.”**

“Contributor” shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those

patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- You must give any other recipients of the Work or Derivative Works a copy of this License; and
- You must cause any modified files to carry prominent notices stating that You changed the files; and
- You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- If the Work includes a “**NOTICE**” text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

Appendix: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets “[]” replaced with your own identifying information. (Don’t include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same “printed page” as the copyright notice for easier identification within third-party archives.

```
Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```



14.7.2 Modified Apache Contributor License Agreement v2.0

Thank you for your interest in the **Chair for VLSI Design, Diagnostics and Architecture** - Faculty of Computer Science, Technische Universität Dresden, Germany (the “Chair”). In order to clarify the intellectual property license granted with Contributions from any person or entity, the Chair must have a Contributor License Agreement (“CLA”) on file that has been signed by each Contributor, indicating agreement to the license terms below. This license is for your protection as a Contributor as well as the protection of the Chair and its users; it does not change your rights to use your own Contributions for any other purpose.

The following CLA is an adaption of the [Apache Contributor License Agreement v2.0](#)

You accept and agree to the following terms and conditions for Your present and future Contributions submitted to the Chair. In return, the Chair shall not use Your Contributions in a way that is contrary to the public benefit or

inconsistent with its nonprofit status and bylaws in effect at the time of the Contribution. Except for the license granted herein to the Chair and recipients of software distributed by the Chair, You reserve all right, title, and interest in and to Your Contributions.

1. **Definitions.** “*You*” (or “*Your*”) shall mean the copyright owner or legal entity authorized by the copyright owner that is making this Agreement with the Chair. For legal entities, the entity making a Contribution and all other entities that control, are controlled by, or are under common control with that entity are considered to be a single Contributor. For the purposes of this definition, “*control*” means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

“*Contribution*” shall mean any original work of authorship, including any modifications or additions to an existing work, that is intentionally submitted by You to the Chair for inclusion in, or documentation of, any of the products owned or managed by the Chair (the “*Work*”). For the purposes of this definition, “submitted” means any form of electronic, verbal, or written communication sent to the Chair or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Chair for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by You as “**Not a Contribution.**”

2. **Grant of Copyright License.** Subject to the terms and conditions of this Agreement, You hereby grant to the Chair and to recipients of software distributed by the Chair a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, sublicense, and distribute Your Contributions and such derivative works.
3. **Grant of Patent License.** Subject to the terms and conditions of this Agreement, You hereby grant to the Chair and to recipients of software distributed by the Chair a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by You that are necessarily infringed by Your Contribution(s) alone or by combination of Your Contribution(s) with the Work to which such Contribution(s) was submitted. If any entity institutes patent litigation against You or any other entity (including a cross-claim or counterclaim in a lawsuit) alleging that your Contribution, or the Work to which you have contributed, constitutes direct or contributory patent infringement, then any patent licenses granted to that entity under this Agreement for that Contribution or Work shall terminate as of the date such litigation is filed.
4. You represent that you are legally entitled to grant the above license. If your employer(s) has rights to intellectual property that you create that includes your Contributions, you represent that you have received permission to make Contributions on behalf of that employer, that your employer has waived such rights for your Contributions to the Chair, or that your employer has executed a separate *Corporate CLA* with the Chair.
5. You represent that each of Your Contributions is Your original creation (see section 7 for submissions on behalf of others). You represent that Your Contribution submissions include complete details of any third-party license or other restriction (including, but not limited to, related patents and trademarks) of which you are personally aware and which are associated with any part of Your Contributions.
6. You are not expected to provide support for Your Contributions, except to the extent You desire to provide support. You may provide support for free, for a fee, or not at all. Unless required by applicable law or agreed to in writing, You provide Your Contributions on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON- INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.
7. Should You wish to submit work that is not Your original creation, You may submit it to the Chair separately from any Contribution, identifying the complete details of its source and of any license or other restriction (including, but not limited to, related patents, trademarks, and license agreements) of which you are personally aware, and conspicuously marking the work as “**Submitted on behalf of a third-party: [named here]**”.
8. You agree to notify the Chair of any facts or circumstances of which you become aware that would make these representations inaccurate in any respect.

14.7.3 Modified Apache Corporate Contributor License Agreement v2.0

Thank you for your interest in the **Chair for VLSI Design, Diagnostics and Architecture** - Faculty of Computer Science, Technische Universität Dresden, Germany (the “Chair”). In order to clarify the intellectual property license granted with Contributions from any person or entity, the Chair must have a Contributor License Agreement (CLA) on file that has been signed by each Contributor, indicating agreement to the license terms below. This license is for your protection as a Contributor as well as the protection of the Chair and its users; it does not change your rights to use your own Contributions for any other purpose.

This version of the Agreement allows an entity (the “Corporation”) to submit Contributions to the Chair, to authorize Contributions submitted by its designated employees to the Chair, and to grant copyright and patent licenses thereto.

The following CLA is an adaption of the [Apache Corporate Contributor License Agreement v2.0](#)

You accept and agree to the following terms and conditions for Your present and future Contributions submitted to the Chair. In return, the Chair shall not use Your Contributions in a way that is contrary to the public benefit or inconsistent with its nonprofit status and bylaws in effect at the time of the Contribution. Except for the license granted herein to the Chair and recipients of software distributed by the Chair, You reserve all right, title, and interest in and to Your Contributions.

1. **Definitions.** “*You*” (or “*Your*”) shall mean the copyright owner or legal entity authorized by the copyright owner that is making this Agreement with the Chair. For legal entities, the entity making a Contribution and all other entities that control, are controlled by, or are under common control with that entity are considered to be a single Contributor. For the purposes of this definition, “control” means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

“*Contribution*” shall mean the code, documentation or other original works of authorship expressly identified in Schedule B, as well as any original work of authorship, including any modifications or additions to an existing work, that is intentionally submitted by You to the Chair for inclusion in, or documentation of, any of the products owned or managed by the Chair (the “*Work*”). For the purposes of this definition, “submitted” means any form of electronic, verbal, or written communication sent to the Chair or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Chair for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by You as “**Not a Contribution.**”

2. **Grant of Copyright License.** Subject to the terms and conditions of this Agreement, You hereby grant to the Chair and to recipients of software distributed by the Chair a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, sublicense, and distribute Your Contributions and such derivative works.
3. **Grant of Patent License.** Subject to the terms and conditions of this Agreement, You hereby grant to the Chair and to recipients of software distributed by the Chair a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by You that are necessarily infringed by Your Contribution(s) alone or by combination of Your Contribution(s) with the Work to which such Contribution(s) were submitted. If any entity institutes patent litigation against You or any other entity (including a cross-claim or counterclaim in a lawsuit) alleging that your Contribution, or the Work to which you have contributed, constitutes direct or contributory patent infringement, then any patent licenses granted to that entity under this Agreement for that Contribution or Work shall terminate as of the date such litigation is filed.

4. You represent that You are legally entitled to grant the above license. You represent further that each employee of the Corporation designated on Schedule A below (or in a subsequent written modification to that Schedule) is authorized to submit Contributions on behalf of the Corporation.
5. You represent that each of Your Contributions is Your original creation (see section 7 for submissions on behalf of others).
6. You are not expected to provide support for Your Contributions, except to the extent You desire to provide support. You may provide support for free, for a fee, or not at all. Unless required by applicable law or agreed to in writing, You provide Your Contributions on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.
7. Should You wish to submit work that is not Your original creation, You may submit it to the Chair separately from any Contribution, identifying the complete details of its source and of any license or other restriction (including, but not limited to, related patents, trademarks, and license agreements) of which you are personally aware, and conspicuously marking the work as “**Submitted on behalf of a third-party: [named here]**”.
8. It is your responsibility to notify the Chair when any change is required to the list of designated employees authorized to submit Contributions on behalf of the Corporation, or to the Corporation’s Point of Contact with the Chair.

Note: This is a local copy of the [Artistic License 2.0](#).

14.7.4 Artistic License 2.0

Copyright © 2000-2006, The Perl Foundation.

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

This license establishes the terms under which a given free software Package may be copied, modified, distributed, and/or redistributed. The intent is that the Copyright Holder maintains some artistic control over the development of that Package while still keeping the Package available as open source and free software. You are always permitted to make arrangements wholly outside of this license directly with the Copyright Holder of a given Package. If the terms of this license do not permit the full use that you propose to make of the Package, you should contact the Copyright Holder and seek a different licensing arrangement.

Definitions

“**Copyright Holder**” means the individual(s) or organization(s) named in the copyright notice for the entire Package.

“**Contributor**” means any party that has contributed code or other material to the Package, in accordance with the Copyright Holder’s procedures.

“**You**” and “**your**” means any person who would like to copy, distribute, or modify the Package.

“**Package**” means the collection of files distributed by the Copyright Holder, and derivatives of that collection and/or of those files. A given Package may consist of either the Standard Version, or a Modified Version.

“**Distribute**” means providing a copy of the Package or making it accessible to anyone else, or in the case of a company or organization, to others outside of your company or organization.

“Distributor Fee” means any fee that you charge for Distributing this Package or providing support for this Package to another party. It does not mean licensing fees.

“Standard Version” refers to the Package if it has not been modified, or has been modified only in ways explicitly requested by the Copyright Holder.

“Modified Version” means the Package, if it has been changed, and such changes were not explicitly requested by the Copyright Holder.

“Original License” means this Artistic License as Distributed with the Standard Version of the Package, in its current version or as it may be modified by The Perl Foundation in the future.

“Source” form means the source code, documentation source, and configuration files for the Package.

“Compiled” form means the compiled bytecode, object code, binary, or any other form resulting from mechanical transformation or translation of the Source form.

Permission for Use and Modification Without Distribution

(1) You are permitted to use the Standard Version and create and use Modified Versions for any purpose without restriction, provided that you do not Distribute the Modified Version.

Permissions for Redistribution of the Standard Version

(2) You may Distribute verbatim copies of the Source form of the Standard Version of this Package in any medium without restriction, either gratis or for a Distributor Fee, provided that you duplicate all of the original copyright notices and associated disclaimers. At your discretion, such verbatim copies may or may not include a Compiled form of the Package.

(3) You may apply any bug fixes, portability changes, and other modifications made available from the Copyright Holder. The resulting Package will still be considered the Standard Version, and as such will be subject to the Original License.

Distribution of Modified Versions of the Package as Source

(4) You may Distribute your Modified Version as Source (either gratis or for a Distributor Fee, and with or without a Compiled form of the Modified Version) provided that you clearly document how it differs from the Standard Version, including, but not limited to, documenting any non-standard features, executables, or modules, and provided that you do at least ONE of the following:

- (a) make the Modified Version available to the Copyright Holder of the Standard Version, under the Original License, so that the Copyright Holder may include your modifications in the Standard Version.
- (b) ensure that installation of your Modified Version does not prevent the user installing or running the Standard Version. In addition, the Modified Version must bear a name that is different from the name of the Standard Version.
- (c) allow anyone who receives a copy of the Modified Version to make the Source form of the Modified Version available to others under (i) the Original License or (ii) a license that permits the licensee to freely copy, modify and redistribute the Modified Version using the same licensing terms that apply to the copy that the licensee received, and requires that the Source form of the Modified Version, and of any works derived from it, be made freely available in that license fees are prohibited but Distributor Fees are allowed.

Distribution of Compiled Forms of the Standard Version or Modified Versions without the Source

(5) You may Distribute Compiled forms of the Standard Version without the Source, provided that you include complete instructions on how to get the Source of the Standard Version. Such instructions must be valid at the time of your distribution. If these instructions, at any time while you are carrying out such distribution, become invalid, you must provide new instructions on demand or cease further distribution. If you provide valid instructions or

cease distribution within thirty days after you become aware that the instructions are invalid, then you do not forfeit any of your rights under this license.

(6) You may Distribute a Modified Version in Compiled form without the Source, provided that you comply with Section 4 with respect to the Source of the Modified Version.

Aggregating or Linking the Package

(7) You may aggregate the Package (either the Standard Version or Modified Version) with other packages and Distribute the resulting aggregation provided that you do not charge a licensing fee for the Package. Distributor Fees are permitted, and licensing fees for other components in the aggregation are permitted. The terms of this license apply to the use and Distribution of the Standard or Modified Versions as included in the aggregation.

(8) You are permitted to link Modified and Standard Versions with other works, to embed the Package in a larger work of your own, or to build stand-alone binary or bytecode versions of applications that include the Package, and Distribute the result without restriction, provided the result does not expose a direct interface to the Package.

Items That are Not Considered Part of a Modified Version

(9) Works (including, but not limited to, modules and scripts) that merely extend or make use of the Package, do not, by themselves, cause the Package to be a Modified Version. In addition, such works are not considered parts of the Package itself, and are not subject to the terms of this license.

General Provisions

(10) Any use, modification, and distribution of the Standard or Modified Versions is governed by this Artistic License. By using, modifying or distributing the Package, you accept this license. Do not use, modify, or distribute the Package, if you do not accept this license.

(11) If your Modified Version has been derived from a Modified Version made by someone other than you, you are nevertheless required to ensure that your Modified Version complies with the requirements of this license.

(12) This license does not grant you the right to use any trademark, service mark, tradename, or logo of the Copyright Holder.

(13) This license includes the non-exclusive, worldwide, free-of-charge patent license to make, have made, use, offer to sell, sell, import and otherwise transfer the Package with respect to any patent claims licensable by the Copyright Holder that are necessarily infringed by the Package. If you institute patent litigation (including a cross-claim or counterclaim) against any party alleging that the Package constitutes direct or contributory patent infringement, then this Artistic License to you shall terminate on the date that such litigation is filed.

14. Disclaimer of Warranty:

THE PACKAGE IS PROVIDED BY THE COPYRIGHT HOLDER AND CONTRIBUTORS “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES. THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT ARE DISCLAIMED TO THE EXTENT PERMITTED BY YOUR LOCAL LAW. UNLESS REQUIRED BY LAW, NO COPYRIGHT HOLDER OR CONTRIBUTOR WILL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING IN ANY WAY OUT OF THE USE OF THE PACKAGE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Note: This is a local copy of the (Revised) BSD License used in the Cocotb project. The original can be found in file LICENSE in the Cocotb source tree.

Todo: Check link to the lib/cocotb/LICENSE file.

14.7.5 BSD License for Cocotb

Cocotb is licensed under the Revised BSD License. Full license text below.

Copyright © 2013 Potential Ventures Ltd Copyright © 2013 SolarFlare Communications Inc All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Potential Ventures Ltd, SolarFlare Communications Inc nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL POTENTIAL VENTURES LTD BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Note: This is a local copy of [The MIT License \(MIT\)](#) used in the UVVM library. The original can be found in file LICENSE in the UVVM_All source tree.

Todo: Check link to the lib/uvvm/LICENSE file.

14.7.6 The MIT License (MIT)

Copyright © 2016 by Bitvis AS

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Note: This is a local copy of the [Mozilla Public License, Version 2.0](#).

14.7.7 Mozilla Public License, v. 2.0

1. Definitions

- 1.1. “Contributor”** means each individual or legal entity that creates, contributes to the creation of, or owns Covered Software.
- 1.2. “Contributor Version”** means the combination of the Contributions of others (if any) used by a Contributor and that particular Contributor’s Contribution.
- 1.3. “Contribution”** means Covered Software of a particular Contributor.
- 1.4. “Covered Software”** means Source Code Form to which the initial Contributor has attached the notice in Exhibit A, the Executable Form of such Source Code Form, and Modifications of such Source Code Form, in each case including portions thereof.
- 1.5. “Incompatible With Secondary Licenses”** means
- that the initial Contributor has attached the notice described in Exhibit B to the Covered Software; or
 - that the Covered Software was made available under the terms of version 1.1 or earlier of the License, but not also under the terms of a Secondary License.
- 1.6. “Executable Form”** means any form of the work other than Source Code Form.
- 1.7. “Larger Work”** means a work that combines Covered Software with other material, in a separate file or files, that is not Covered Software.
- 1.8. “License”** means this document.
- 1.9. “Licensable”** means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently, any and all of the rights conveyed by this License.
- 1.10. “Modifications”** means any of the following:
- any file in Source Code Form that results from an addition to, deletion from, or modification of the contents of Covered Software; or
 - any new file in Source Code Form that contains any Covered Software.
- 1.11. “Patent Claims” of a Contributor** means any patent claim(s), including without limitation, method, process, and apparatus claims, in any patent Licensable by such Contributor that would be infringed, but for the grant of the License, by the making, using, selling, offering for sale, having made, import, or transfer of either its Contributions or its Contributor Version.
- 1.12. “Secondary License”** means either the GNU General Public License, Version 2.0, the GNU Lesser General Public License, Version 2.1, the GNU Affero General Public License, Version 3.0, or any later versions of those licenses.
- 1.13. “Source Code Form”** means the form of the work preferred for making modifications.
- 1.14. “You” (or “Your”)** means an individual or a legal entity exercising rights under this License. For legal entities, “You” includes any entity that controls, is controlled by, or is under common control with You. For purposes of this definition, “control” means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. License Grants and Conditions

2.1. Grants

Each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license:

- under intellectual property rights (other than patent or trademark) Licensable by such Contributor to use, reproduce, make available, modify, display, perform, distribute, and otherwise exploit its Contributions, either on an unmodified basis, with Modifications, or as part of a Larger Work; and

- under Patent Claims of such Contributor to make, use, sell, offer for sale, have made, import, and otherwise transfer either its Contributions or its Contributor Version.

2.2. Effective Date

The licenses granted in Section 2.1 with respect to any Contribution become effective for each Contribution on the date the Contributor first distributes such Contribution.

2.3. Limitations on Grant Scope

The licenses granted in this Section 2 are the only rights granted under this License. No additional rights or licenses will be implied from the distribution or licensing of Covered Software under this License. Notwithstanding Section 2.1(b) above, no patent license is granted by a Contributor:

- for any code that a Contributor has removed from Covered Software; or
- for infringements caused by: (i) Your and any other third party's modifications of Covered Software, or (ii) the combination of its Contributions with other software (except as part of its Contributor Version); or
- under Patent Claims infringed by Covered Software in the absence of its Contributions.

This License does not grant any rights in the trademarks, service marks, or logos of any Contributor (except as may be necessary to comply with the notice requirements in Section 3.4).

2.4. Subsequent Licenses

No Contributor makes additional grants as a result of Your choice to distribute the Covered Software under a subsequent version of this License (see Section 10.2) or under the terms of a Secondary License (if permitted under the terms of Section 3.3).

2.5. Representation

Each Contributor represents that the Contributor believes its Contributions are its original creation(s) or it has sufficient rights to grant the rights to its Contributions conveyed by this License.

2.6. Fair Use

This License is not intended to limit any rights You have under applicable copyright doctrines of fair use, fair dealing, or other equivalents.

2.7. Conditions

Sections 3.1, 3.2, 3.3, and 3.4 are conditions of the licenses granted in Section 2.1.

3. Responsibilities

3.1. Distribution of Source Form

All distribution of Covered Software in Source Code Form, including any Modifications that You create or to which You contribute, must be under the terms of this License. You must inform recipients that the Source Code Form of the Covered Software is governed by the terms of this License, and how they can obtain a copy of this License. You may not attempt to alter or restrict the recipients' rights in the Source Code Form.

3.2. Distribution of Executable Form

If You distribute Covered Software in Executable Form then:

- such Covered Software must also be made available in Source Code Form, as described in Section 3.1, and You must inform recipients of the Executable Form how they can obtain a copy of such Source Code Form by reasonable means in a timely manner, at a charge no more than the cost of distribution to the recipient; and
- You may distribute such Executable Form under the terms of this License, or sublicense it under different terms, provided that the license for the Executable Form does not attempt to limit or alter the recipients' rights in the Source Code Form under this License.

3.3. Distribution of a Larger Work

You may create and distribute a Larger Work under terms of Your choice, provided that You also comply with the requirements of this License for the Covered Software. If the Larger Work is a combination of Covered Software with a work governed by one or more Secondary Licenses, and the Covered Software is not Incompatible With Secondary Licenses, this License permits You to additionally distribute such Covered Software under the terms of such Secondary License(s), so that the recipient of the Larger Work may, at their option, further distribute the Covered Software under the terms of either this License or such Secondary License(s).

3.4. Notices

You may not remove or alter the substance of any license notices (including copyright notices, patent notices, disclaimers of warranty, or limitations of liability) contained within the Source Code Form of the Covered Software, except that You may alter any license notices to the extent required to remedy known factual inaccuracies.

3.5. Application of Additional Terms

You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, You may do so only on Your own behalf, and not on behalf of any Contributor. You must make it absolutely clear that any such warranty, support, indemnity, or liability obligation is offered by You alone, and You hereby agree to indemnify every Contributor for any liability incurred by such Contributor as a result of warranty, support, indemnity or liability terms You offer. You may include additional disclaimers of warranty and limitations of liability specific to any jurisdiction.

4. Inability to Comply Due to Statute or Regulation

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Software due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be placed in a text file included with all distributions of the Covered Software under this License. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Termination

5.1. The rights granted under this License will terminate automatically if You fail to comply with any of its terms. However, if You become compliant, then the rights granted under this License from a particular Contributor are reinstated (a) provisionally, unless and until such Contributor explicitly and finally terminates Your grants, and (b) on an ongoing basis, if such Contributor fails to notify You of the non-compliance by some reasonable means prior to 60 days after You have come back into compliance. Moreover, Your grants from a particular Contributor are reinstated on an ongoing basis if such Contributor notifies You of the non-compliance by some reasonable

means, this is the first time You have received notice of non-compliance with this License from such Contributor, and You become compliant prior to 30 days after Your receipt of the notice.

5.2. If You initiate litigation against any entity by asserting a patent infringement claim (excluding declaratory judgment actions, counter-claims, and cross-claims) alleging that a Contributor Version directly or indirectly infringes any patent, then the rights granted to You by any and all Contributors for the Covered Software under Section 2.1 of this License shall terminate.

5.3. In the event of termination under Sections 5.1 or 5.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or Your distributors under this License prior to termination shall survive termination.

6. Disclaimer of Warranty

Covered Software is provided under this License on an “as is” basis, without warranty of any kind, either expressed, implied, or statutory, including, without limitation, warranties that the Covered Software is free of defects, merchantable, fit for a particular purpose or non-infringing. The entire risk as to the quality and performance of the Covered Software is with You. Should any Covered Software prove defective in any respect, You (not any Contributor) assume the cost of any necessary servicing, repair, or correction. This disclaimer of warranty constitutes an essential part of this License. No use of any Covered Software is authorized under this License except under this disclaimer.

7. Limitation of Liability

Under no circumstances and under no legal theory, whether tort (including negligence), contract, or otherwise, shall any Contributor, or anyone who distributes Covered Software as permitted above, be liable to You for any direct, indirect, special, incidental, or consequential damages of any character including, without limitation, damages for lost profits, loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses, even if such party shall have been informed of the possibility of such damages. This limitation of liability shall not apply to liability for death or personal injury resulting from such party’s negligence to the extent applicable law prohibits such limitation. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so this exclusion and limitation may not apply to You.

8. Litigation

Any litigation relating to this License may be brought only in the courts of a jurisdiction where the defendant maintains its principal place of business and such litigation shall be governed by laws of that jurisdiction, without reference to its conflict-of-law provisions. Nothing in this Section shall prevent a party’s ability to bring cross-claims or counter-claims.

9. Miscellaneous

This License represents the complete agreement concerning the subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not be used to construe this License against a Contributor.

10. Versions of the License

10.1. New Versions

Mozilla Foundation is the license steward. Except as provided in Section 10.3, no one other than the license steward has the right to modify or publish new versions of this License. Each version will be given a distinguishing version number.

10.2. Effect of New Versions

You may distribute the Covered Software under the terms of the version of the License under which You originally received the Covered Software, or under the terms of any subsequent version published by the license steward.

10.3. Modified Versions

If you create software not governed by this License, and you want to create a new license for such software, you may create and use a modified version of this License if you rename the license and remove any references to the name of the license steward (except to note that such modified license differs from this License).

10.4. Distributing Source Code Form that is Incompatible With Secondary Licenses

If You choose to distribute Source Code Form that is Incompatible With Secondary Licenses under the terms of this version of the License, the notice described in Exhibit B of this License must be attached.

Exhibit A - Source Code Form License Notice

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at <http://mozilla.org/MPL/2.0/>.

If it is not possible or desirable to put the notice in a particular file, then You may include the notice in a location (such as a LICENSE file in a relevant directory) where a recipient would be likely to look for such a notice.

You may add additional accurate notices of copyright ownership.

Exhibit B - "Incompatible With Secondary Licenses" Notice

This Source Code Form is "Incompatible With Secondary Licenses", as defined by the Mozilla Public License, v. 2.0.

Part IV

Appendix

CHAPTER 15

Change Log

CHAPTER 16

Index

Symbols

–all

compile-altera.sh command line option, 221
 compile-lattice.sh command line option, 222
 compile-osvvm.sh command line option, 224
 compile-uvvm.sh command line option, 225
 compile-xilinx-ise.sh command line option, 227
 compile-xilinx-vivado.sh command line option, 229

–clean

compile-altera.sh command line option, 221
 compile-lattice.sh command line option, 222
 compile-osvvm.sh command line option, 224
 compile-uvvm.sh command line option, 225
 compile-xilinx-ise.sh command line option, 227
 compile-xilinx-vivado.sh command line option, 229

–ghdl

compile-altera.sh command line option, 221
 compile-lattice.sh command line option, 222
 compile-osvvm.sh command line option, 224
 compile-uvvm.sh command line option, 226
 compile-xilinx-ise.sh command line option, 227
 compile-xilinx-vivado.sh command line option, 229

–help

compile-altera.sh command line option, 221
 compile-lattice.sh command line option, 222
 compile-osvvm.sh command line option, 224
 compile-uvvm.sh command line option, 225
 compile-xilinx-ise.sh command line option, 227
 compile-xilinx-vivado.sh command line option, 229

–questa

compile-altera.sh command line option, 221
 compile-lattice.sh command line option, 222
 compile-osvvm.sh command line option, 224
 compile-uvvm.sh command line option, 226
 compile-xilinx-ise.sh command line option, 227
 compile-xilinx-vivado.sh command line option, 229

–vhdl2008

compile-altera.sh command line option, 221

compile-lattice.sh command line option, 223

compile-osvvm.sh command line option, 224

compile-uvvm.sh command line option, 226

compile-xilinx-ise.sh command line option, 227

compile-xilinx-vivado.sh command line option, 229

–vhdl93

compile-altera.sh command line option, 221

compile-lattice.sh command line option, 223

compile-osvvm.sh command line option, 224

compile-uvvm.sh command line option, 226

compile-xilinx-ise.sh command line option, 227

compile-xilinx-vivado.sh command line option, 229

–All

compile-altera.ps1 command line option, 221

compile-lattice.ps1 command line option, 223

compile-osvvm.ps1 command line option, 225

compile-uvvm.ps1 command line option, 226

compile-xilinx-ise.ps1 command line option, 228

compile-xilinx-vivado.ps1 command line option, 230

–Clean

compile-altera.ps1 command line option, 221

compile-lattice.ps1 command line option, 223

compile-osvvm.ps1 command line option, 225

compile-uvvm.ps1 command line option, 226

compile-xilinx-ise.ps1 command line option, 228

compile-xilinx-vivado.ps1 command line option, 230

–D

poc.ps1 command line option, 219

poc.sh command line option, 220

–GHDL

compile-altera.ps1 command line option, 222

compile-lattice.ps1 command line option, 223

compile-osvvm.ps1 command line option, 225

compile-uvvm.ps1 command line option, 226

compile-xilinx-ise.ps1 command line option, 228

compile-xilinx-vivado.ps1 command line option, 230

–Help

compile-altera.ps1 command line option, 221

compile-lattice.ps1 command line option, 223

- compile-osvvm.ps1 command line option, [225](#)
 - compile-uvvm.ps1 command line option, [226](#)
 - compile-xilinx-ise.ps1 command line option, [228](#)
 - compile-xilinx-vivado.ps1 command line option, [230](#)
 - Questa
 - compile-altera.ps1 command line option, [222](#)
 - compile-lattice.ps1 command line option, [223](#)
 - compile-osvvm.ps1 command line option, [225](#)
 - compile-uvvm.ps1 command line option, [226](#)
 - compile-xilinx-ise.ps1 command line option, [228](#)
 - compile-xilinx-vivado.ps1 command line option, [230](#)
 - ReLink
 - compile-xilinx-ise.ps1 command line option, [228](#)
 - compile-xilinx-vivado.ps1 command line option, [230](#)
 - VHDL2008
 - compile-altera.ps1 command line option, [222](#)
 - compile-lattice.ps1 command line option, [223](#)
 - compile-osvvm.ps1 command line option, [225](#)
 - compile-uvvm.ps1 command line option, [226](#)
 - compile-xilinx-ise.ps1 command line option, [228](#)
 - compile-xilinx-vivado.ps1 command line option, [230](#)
 - VHDL93
 - compile-altera.ps1 command line option, [222](#)
 - compile-lattice.ps1 command line option, [223](#)
 - compile-osvvm.ps1 command line option, [225](#)
 - compile-uvvm.ps1 command line option, [226](#)
 - compile-xilinx-ise.ps1 command line option, [228](#)
 - compile-xilinx-vivado.ps1 command line option, [230](#)
- ## A
- Altera
- Pre-compilation, [50](#)
- ## C
- Cocotb
- Pre-compilation, [55](#)
 - Third-Party Libraries, [203](#)
- compile-altera.ps1 command line option
- All, [221](#)
 - Clean, [221](#)
 - GHDL, [222](#)
 - Help, [221](#)
 - Questa, [222](#)
 - VHDL2008, [222](#)
 - VHDL93, [222](#)
- compile-altera.sh command line option
- all, [221](#)
 - clean, [221](#)
 - ghdl, [221](#)
 - help, [221](#)
 - questa, [221](#)
 - vhdl2008, [221](#)
 - vhdl93, [221](#)
- compile-lattice.ps1 command line option
- All, [223](#)
 - Clean, [223](#)
 - GHDL, [223](#)
 - Help, [223](#)
 - Questa, [223](#)
 - VHDL2008, [223](#)
 - VHDL93, [223](#)
- compile-lattice.sh command line option
- all, [222](#)
 - clean, [222](#)
 - ghdl, [222](#)
 - help, [222](#)
 - questa, [222](#)
 - vhdl2008, [223](#)
 - vhdl93, [223](#)
- compile-osvvm.ps1 command line option
- All, [225](#)
 - Clean, [225](#)
 - GHDL, [225](#)
 - Help, [225](#)
 - Questa, [225](#)
 - VHDL2008, [225](#)
 - VHDL93, [225](#)
- compile-osvvm.sh command line option
- all, [224](#)
 - clean, [224](#)
 - ghdl, [224](#)
 - help, [224](#)
 - questa, [224](#)
 - vhdl2008, [224](#)
 - vhdl93, [224](#)
- compile-uvvm.ps1 command line option
- All, [226](#)
 - Clean, [226](#)
 - GHDL, [226](#)
 - Help, [226](#)
 - Questa, [226](#)
 - VHDL2008, [226](#)
 - VHDL93, [226](#)
- compile-uvvm.sh command line option
- all, [225](#)
 - clean, [225](#)
 - ghdl, [226](#)
 - help, [225](#)
 - questa, [226](#)
 - vhdl2008, [226](#)
 - vhdl93, [226](#)
- compile-xilinx-ise.ps1 command line option
- All, [228](#)
 - Clean, [228](#)
 - GHDL, [228](#)
 - Help, [228](#)
 - Questa, [228](#)
 - ReLink, [228](#)
 - VHDL2008, [228](#)
 - VHDL93, [228](#)
- compile-xilinx-ise.sh command line option

- all, 227
- clean, 227
- ghdl, 227
- help, 227
- questa, 227
- vhdl2008, 227
- vhdl93, 227

compile-xilinx-vivado.ps1 command line option

- All, 230
- Clean, 230
- GHDL, 230
- Help, 230
- Questa, 230
- ReLink, 230
- VHDL2008, 230
- VHDL93, 230

compile-xilinx-vivado.sh command line option

- all, 229
- clean, 229
- ghdl, 229
- help, 229
- questa, 229
- vhdl2008, 229
- vhdl93, 229

E

environment variable

- LM_LICENSE_FILE, 219
- PoCRootDirectory, 219, 220

L

Lattice

- Pre-compilation, 51

LM_LICENSE_FILE, 219

O

OSVVM

- Pre-compilation, 54
- Third-Party Libraries, 203

P

poc.ps1 command line option

- D, 219

poc.sh command line option

- D, 220

PoCRootDirectory, 220

Pre-compilation, 49

- Altera, 50
- Cocotb, 55
- Lattice, 51
- OSVVM, 54
- Simulator Adapters, 55
- Supported Simulators, 50
- Third-Party Libraries, 53
- UVVM, 54
- Vendor Primitives, 50
- Xilinx ISE, 52
- Xilinx Vivado, 53

S

Simulator Adapters

- Pre-compilation, 55

Supported Simulators

- Pre-compilation, 50

T

T_SORTNET_IMPL (C type), 188

Third-Party Libraries, 202

- Cocotb, 203
- OSVVM, 203
- Pre-compilation, 53
- UVVM, 203
- VUnit, 204

U

UVVM

- Pre-compilation, 54
- Third-Party Libraries, 203

V

Vendor Primitives

- Pre-compilation, 50

VUnit

- Third-Party Libraries, 204

X

Xilinx ISE

- Pre-compilation, 52

Xilinx Vivado

- Pre-compilation, 53