The PoC-Library Documentation

Release 1.0.0

Patrick Lehmann, Thomas B. Preusser, Martin Zabel

October 28, 2016

Contents

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 hierachy.

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.

CHAPTER 1

News

1.1 13.05.2016 - PoC 1.0.0 was released.

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

CHAPTER 2

Cite the PoC-Library

The PoC-Library hosted at GitHub.com. Please use the following biblatex 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},
}
```

2.1 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 hierachy.

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 Infrastruture 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.

2.1.1 What is the History of PoC?

In the past years, a lot of "IP cores" were developed at the chair of VLSI design [1] . This lose set of HDL designs was gathered in an old-fashioned CVS repository and grow 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, its 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.

2.1.2 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.
  - **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.

[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-eda

  - **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 documention section for Vivado for more details.

- Simulation Tool Chains:
  - **Aldec Active-HDL** Tested with Active-HDL 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 QuestaSim/ModelSim** Tested with ModelSim Altera Edition 10.3d and ModelSim Altera Starter Edition 10.3d Tested with Mentor QuestaSim 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) 2016.1. 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 0.34dev and GTKWave 3.3.70 Due to ungoing development and bugfixes, we encourage to use the newest GHDL version.

2.1.3 Why should I use PoC?

Here is a brief list of advantages:

- We explicitly use the wording *PoC-Library* rather then *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 varous synthesis tools.

Compare hardware usage for different target platfroms.

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

---

2.1.4 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.

2.2 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

2.2.1 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 synthezise IP cores.
- A supported simulator too 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.
- OS-VVM Open Source 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-Root\lib directory. See Third Party Libraries for more details on these libraries.

2.2.2 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	`git clone --recursive https://github.com/VLSI-EDA/PoC.git PoC`
SSH	`git clone --recursive ssh://git@github.com:VLSI-EDA/PoC.git PoC`

2.2.3 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: `Y` to accept, `N` to decline, `P` to skip/pass a step and `Return` to accept a default value displayed in brackets.

2.2.4 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.ps1 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, EP2SGX90
```

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/proj
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.

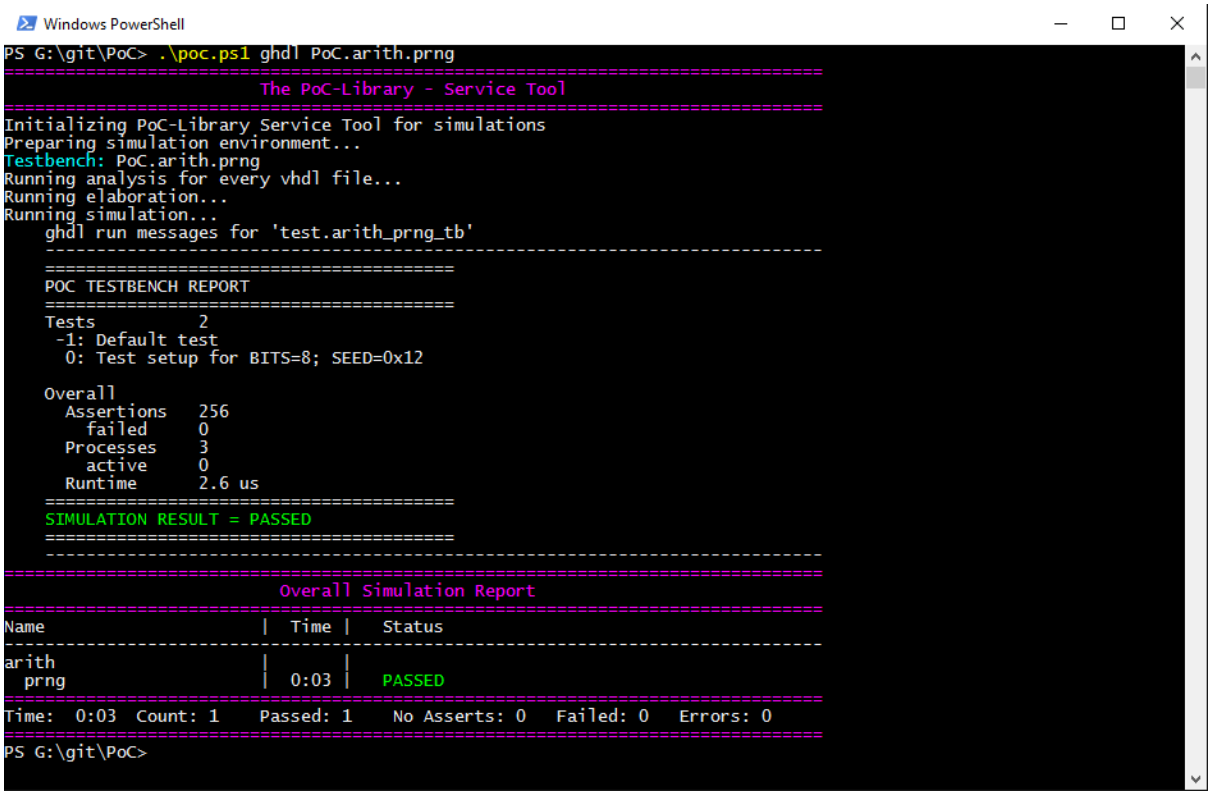
2.2.5 Run a Simulation

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 an 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:



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.

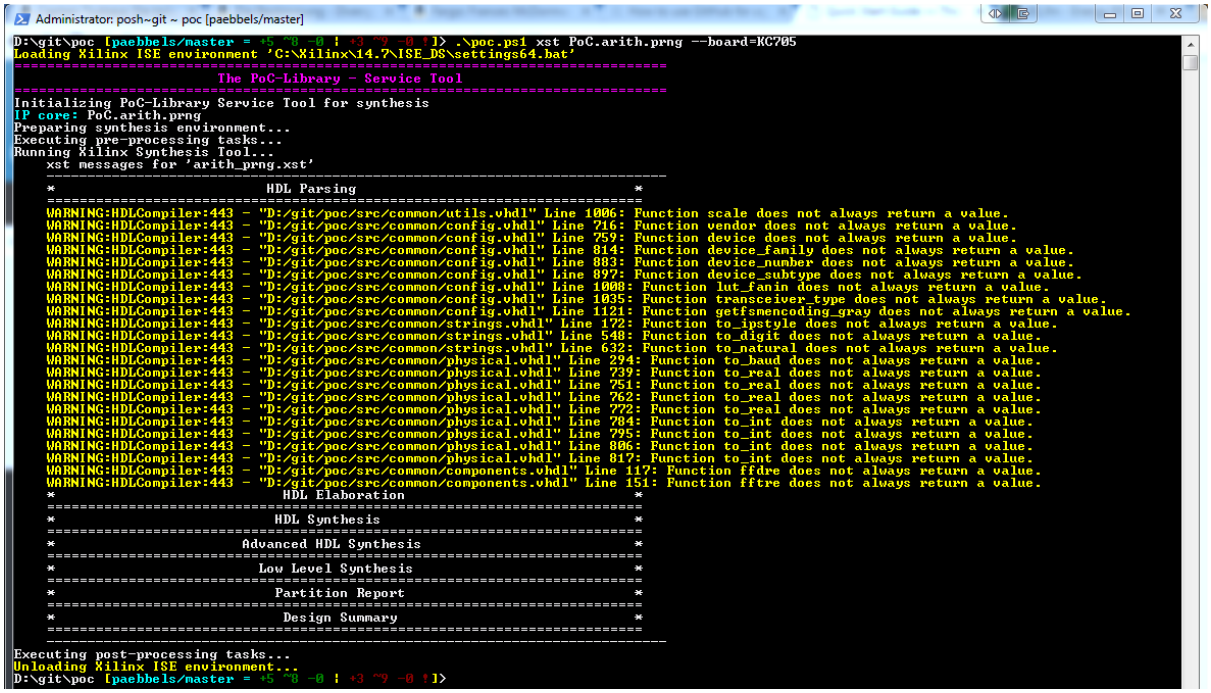
2.2.6 Run a Synthesis

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:

```
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.



2.2.7 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.

2.3 Using PoC

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 copyed 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 copyed 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 copyed 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 copyed 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.

2.3.1 Requirements

Contents of this Page

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.

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 13.0 or
  - Lattice Diamond or
  - Xilinx ISE 14.7 [1] or
  - Xilinx Vivado [2]
- Simulation tool chains
  - Aldec Active-HDL or
  - Mentor Graphics ModelSim Altera Edition or
  - Mentor Graphics QuestaSim or
  - Xilinx ISE Simulator 14.7 or
  - Xilinx Vivado Simulator 2016.1 [3] or
  - GHDL 0.34dev and GTKWave 3.3.70

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.

[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.

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`

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`

Windows specific requirements:

- **PowerShell 4.0** PowerShell shipped with Windows since Vista. It is a part if the Windows Management Framework. If the required version not already included in Windows, it can be downloaded from microsoft.com: WMF 4.0, WMF 5.0 (recommended).
  - Allow local script execution (read more) `Set-ExecutionPolicy RemoteSigned`
  - PowerShell Community Extensions (PSCX) 3.2 The latest PSCX can be downloaded from PowerShellGallery

Optional Tools on Windows:

- **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 Installing posh-git with PsGet package manager: `Install-Module posh-git`

2.3.2 Downloading PoC

Contents of this Page

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

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.

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)."
```

2.3.3 Integrating PoC into Projects

Contents of this page

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

2.3.4 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 setup process is started by invoking PoC's frontend script with the command `configure`. Please follow the instructions on screen. Use the keyboard buttons: `Y` to accept, `N` to decline, `P` to skip/pass a step and `Return` 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)
```

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 infered 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)
```

Git

Note: Setting up Git and Git developer settings, is an advanced feature recommended for all developers interrested 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...
```

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-Editi
```

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\mode
```

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]:
```

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\10.4
```

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]:
```

GHDL

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

GTKWave

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

Hook Files

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

2.3.5 Creating my_config/my_project.vhdl

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

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 projects source directory and renamed into `my_config.vhdl`.

This file should be included into 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 infered from the board information stored in `PoCRoot\src\common\board.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";
```

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 copyed 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/p
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.

2.3.6 Adding IP Cores to a Project

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.

2.3.7 Simulation

Contents of this Page

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 fronend script on Darwin, Linux and Unix platforms.
- **poc.ps1:** `poc.ps1 <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.

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.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 it's outputs are displayed in console:

```
Windows PowerShell
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 vhdl 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>
```

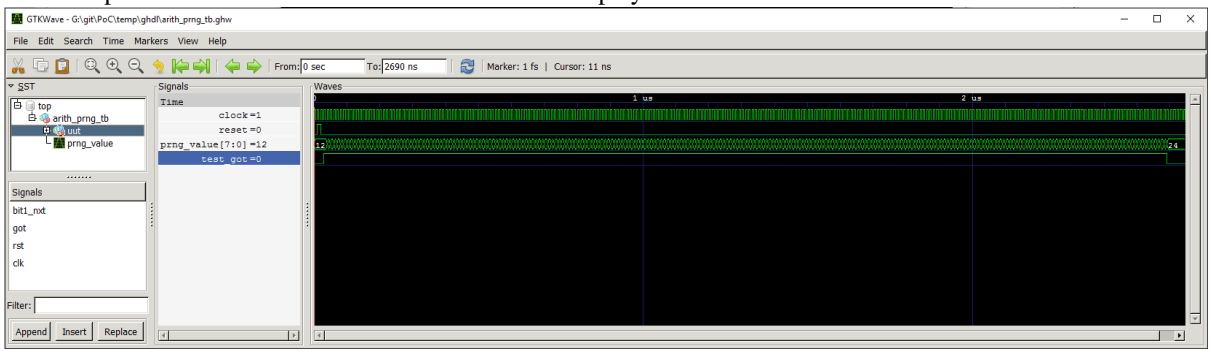
Each testbench uses PoC's simulation helper packages to count asserts and to track active stimuli and checker processes. After a completed simulation run, an report is written to STDOUT or the simulator's console. Note the line `SIMULATION RESULT = PASSED`. For each simulated PoC entity, a line in the overall report is created. It lists the runtime per testbench and the simulation status (`...` `ERROR`, `FAILED`, `NO ASSERTS` or `PASSED`).

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:



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.

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 it's specialties, so a wrapper script is needed as abstraction from the host's operating system. Depending on the choosen 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 choosen, 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 there 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
	–board=<BOARD>	Specify a target board.
	–device=<DEVICE>	Specify a target device.
-g	–gui	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
	–board=<BOARD>	Specify a target board.
	–device=<DEVICE>	Specify a target device.
-g	–gui	Start GTKwave, if installed. Open *.gtkw, if available.
	–std=[87|93|02|08]	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
	–board=<BOARD>	Specify a target board.
	–device=<DEVICE>	Specify a target device.
-g	–gui	Start the simulation in the QuestaSim GUI.
	–std=[87|93|02|08]	Select a VHDL standard. Default: 08

Example:

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

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
	–board=<BOARD>	Specify a target board.
	–device=<DEVICE>	Specify a target device.
-g	–gui	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 (isim) is `xsim` followed by a list of PoC entities. The following options are supported for Vivado Simulator:

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

Example:

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

Running a Group of Testbenches

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

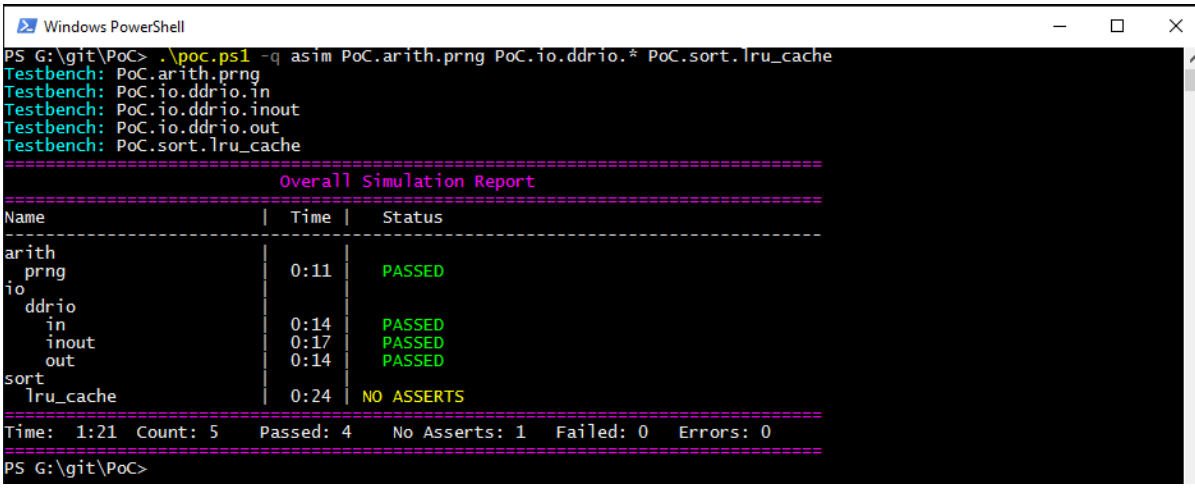
Supported wildcard patterns are `*` and `?`. Question mark refers to all entities in a PoC (sub-)namespace. Asterisk refers to all PoC entiries in the current namespace and all sub-namespaces.

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.cache.* 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
```

Resulting output:

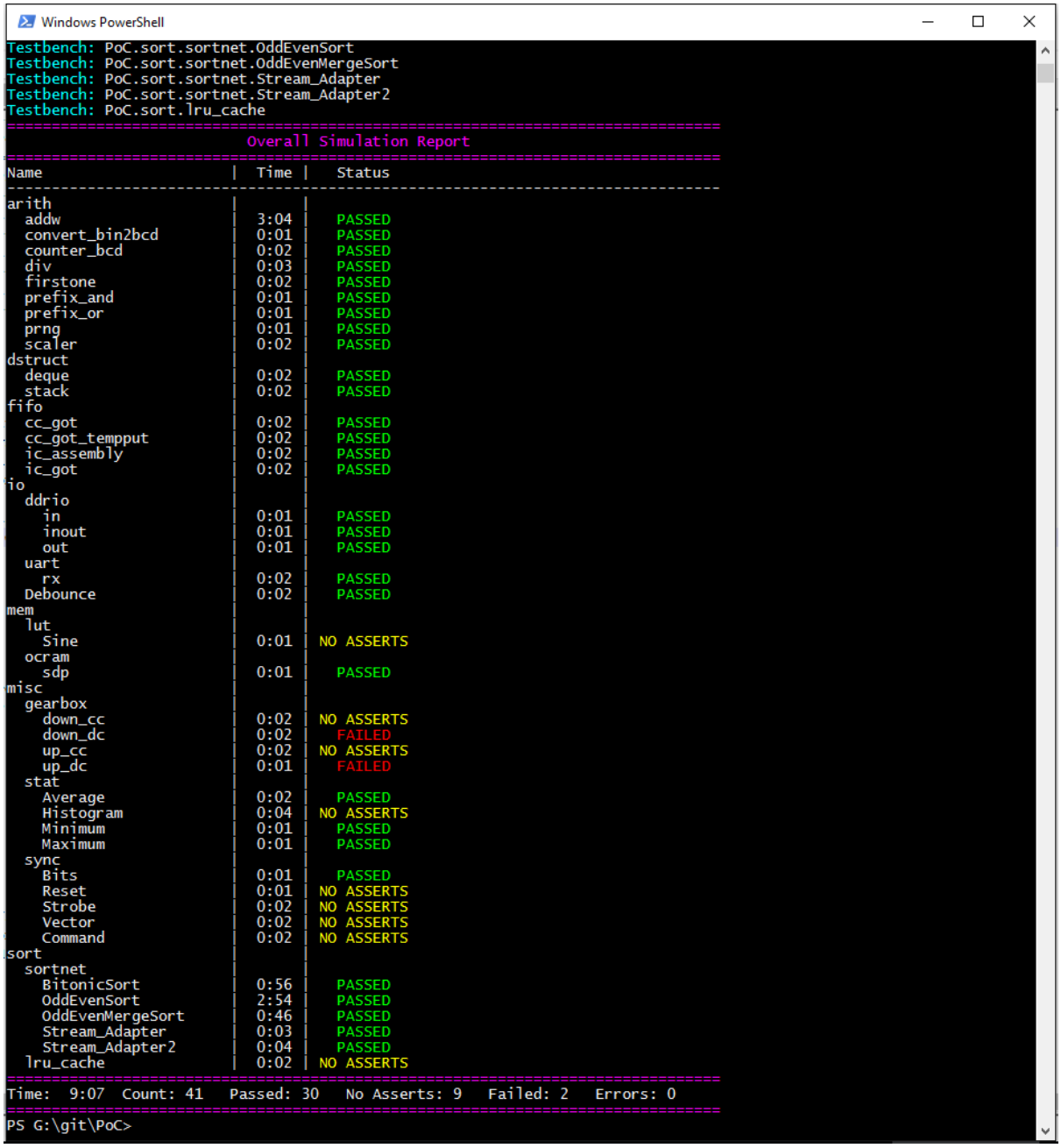


Continuous Integration (CI)

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.*
```



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.

2.3.8 Synthesis

Contents of this Page

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 fronend 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.

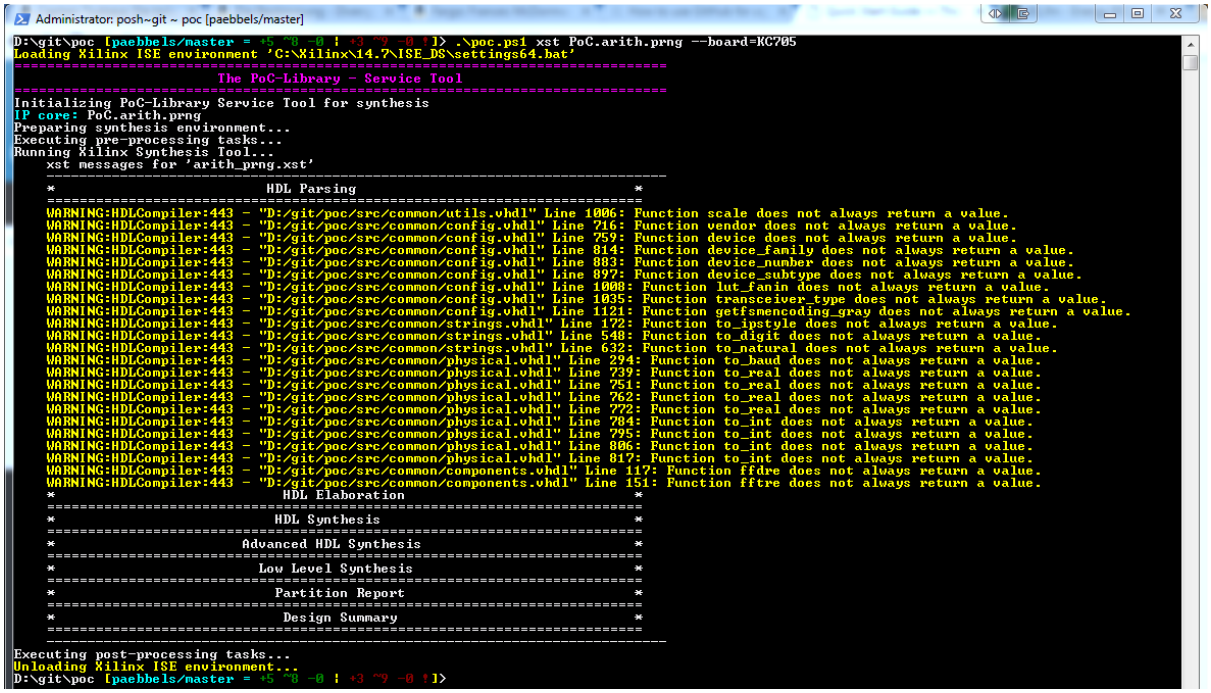
Quick Example

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
```

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.



Running a single Synthesis

A synthesis run is supervised by PoC's `PoCRoot\py\PoC.py` service tool, which offers a consistent interface to all synthesizers. Unfortunately, every platform has it's specialties, so a wrapper script is needed as abstraction from the host's operating system. Depending on the choosen 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> <compiler> <module> <compiler 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 synthesizers can be choosen, if installed and configured in PoC:

Simulator	Description
quartus	Altera Quartus II or Quartus Prime
lse	Lattice Diamond - Lattice Synthesis Engine (LSE)
xst	Xilinx ISE Systhesis Tool (XST)
coregen	Xilinx ISE Core Generator (CoreGen)
vivado	Xilinx Vivado Synthesis

Altera Quartus

The command to invoke a synthesis using Altera Quartus II or Quartus Prime is `quartus` followed by a list of PoC entities. The following options are supported for Quartus:

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 Altera Quartus II or Quartus Prime is `quartus` followed by a list of PoC entities. The following options are supported for Quartus:

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
```

Xilinx ISE Synthesis Tool (XST)

The command to invoke a synthesis using Altera Quartus II or Quartus Prime is `quartus` followed by a list of PoC entities. The following options are supported for Quartus:

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
```

Xilinx ISE Core Generator

The command to invoke a synthesis using Altera Quartus II or Quartus Prime is `quartus` followed by a list of PoC entities. The following options are supported for Quartus:

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
```

Xilinx Vivado Synthesis

The command to invoke a synthesis using Altera Quartus II or Quartus Prime is `quartus` followed by a list of PoC entities. The following options are supported for Quartus:

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
```

2.3.9 Project Management

Overview

Solutions

Projects

2.3.10 Pre-Compiling Vendor Libraries

Contents of this Page

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>`.

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 Active-HDL Lattice Ed. Reviera-PRO	planned planned planned	planned shipped planned	planned planned planned	planned planned planned
Mentor	ModelSim ModelSim Altera Ed. QuestaSim	yes shipped yes	yes yes yes	yes yes yes	yes yes yes
Xilinx	ISE Simulator Vivado Simulator			shipped not supported	not supported shipped

FPGA Vendor's Primitive Libraries

Altera

Note: The Altera Quartus tool chain needs to be configured in PoC. See Configuring PoC's Infrastruture 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		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
	–vhdl93	Compile only for VHDL-93
	–vhdl2008	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		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
	-VHDL93	Compile only for VHDL-93
	-VHDL2008	Compile only for VHDL-2008

Lattice

Note: The Lattice Diamond tool chain needs to be configured in PoC. See Configuring PoC's Infrastruture 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		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
	–vhdl93	Compile only for VHDL-93
	–vhdl2008	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		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
	-VHDL93	Compile only for VHDL-93
	-VHDL2008	Compile only for VHDL-2008

Xilinx ISE

Note: The Xilinx ISE tool chain needs to be configured in PoC. See Configuring PoC's Infrastruture 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 --vhdl2008
```

List of command line arguments:

Common Option		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
	–vhdl93	Compile only for VHDL-93
	–vhdl2008	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		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
	-VHDL93	Compile only for VHDL-93
	-VHDL2008	Compile only for VHDL-2008

Xilinx Vivado

Note: The Xilinx Vivado tool chain needs to be configured in PoC. See Configuring PoC's Infrastruture 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		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
	–vhdl93	Compile only for VHDL-93
	–vhdl2008	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		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
	-VHDL93	Compile only for VHDL-93
	-VHDL2008	Compile only for VHDL-2008

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		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

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		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

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		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

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		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

2.3.11 Miscellaneous

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

GNU Emacs

Todo

No documentation available.

Git

- `git-alias.setup.ps1`/`git-alias.setup.sh` registers new global aliasses 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
    git config --global alias.tree 'log --decorate --pretty=oneline --abbrev-commit --date-order
```

Browse the Git directory.

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.

2.4 IP Core Documentations

Namespace for Packages:

2.4.1 Common Packages

These are common packages....

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

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

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

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

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

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

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

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

2.4.2 Simulation Packages

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

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

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

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

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

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

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 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

sim_waveform

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

Namespaces for Entities:

2.4.3 alt

Todo

This namespace is reserved for Altera specific entities.

2.4.4 arith

These are arithmetic entities....

Package

PoC.arith

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

Package

This package holds all component declarations for this namespace.

Source file: arith/arith.pkg.vhdl

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:

```
entity arith_addw is
  generic (
    N : positive;                    -- Operand Width
    K : positive;                    -- Block Count

    ARCH        : tArch     := AAM;        -- Architecture
    BLOCKING    : tBlocking := DFLT;       -- Blocking Scheme
    SKIPPING    : tSkipping := CCC;        -- Carry Skip Scheme
    P_INCLUSIVE : boolean   := false       -- Use Inclusive Propagate, i.e. c^1
  );
  port (
    a, b : in std_logic_vector(N-1 downto 0);
    cin  : in std_logic;

    s    : out std_logic_vector(N-1 downto 0);
    cout : out std_logic
  );
end entity;
```

Source file: arith/arith_addw.vhdl

arith_carrychain_inc

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

Y <= X + (0...0) & Cin

Entity Declaration:

```
entity arith_carrychain_inc is
  generic (
    BITS      : positive
  );
  port (
    X   : in  std_logic_vector(BITS - 1 downto 0);
    CIn : in  std_logic                           := '1';
    Y   : out std_logic_vector(BITS - 1 downto 0)
  );
end entity;
```

Source file: arith/arith_carrychain_inc.vhdl

arith_convert_bin2bcd

Todo

No documentation available.

Entity Declaration:

```
entity arith_convert_bin2bcd is
  generic (
    BITS          : positive    := 8;
    DIGITS        : positive    := 3;
    RADIX         : positive    := 2
  );
  port (
    Clock         : in  std_logic;
    Reset         : in  std_logic;

    Start         : in  std_logic;
    Busy          : out std_logic;

    Binary        : in  std_logic_vector(BITS - 1 downto 0);
    IsSigned      : in  std_logic                           := '0';
    BCDDigits     : out T_BCD_VECTOR(DIGITS - 1 downto 0);
    Sign          : out std_logic
  );
end entity;
```

Source file: arith/arith_convert_bin2bcd.vhdl

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:

```
entity arith_counter_bcd is
  generic (
    DIGITS : positive                              -- Number of BCD digits
  );
  port (
    clk : in  std_logic;
    rst : in  std_logic;                           -- Reset to 0
    inc : in  std_logic;                           -- Increment
    val : out T_BCD_VECTOR(DIGITS-1 downto 0)      -- Value output
  );
end entity;
```

Source file: arith/arith_counter_bcd.vhdl

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:

```
entity arith_counter_free is
  generic (
    DIVIDER : positive
  );
  port (
    -- Global Control
    clk : in  std_logic;
    rst : in  std_logic;

    inc : in  std_logic;
    stb : out std_logic                  -- End-of-Period Strobe
  );
end entity arith_counter_free;
```

Source file: arith/arith_counter_free.vhdl

arith_counter_gray

Todo

No documentation available.

Entity Declaration:

```
entity arith_counter_gray is
  generic (
    BITS : positive;                                -- Bit width of the counter
    INIT : natural        := 0                      -- Initial/reset counter value
  );
  port (
    clk : in  std_logic;
    rst : in  std_logic;                            -- Reset to INIT value
    inc : in  std_logic;                            -- Increment
    dec : in  std_logic   := '0';                   -- Decrement
    val : out std_logic_vector(BITS-1 downto 0);    -- Value output
    cry : out std_logic                             -- Carry output
  );
end entity arith_counter_gray;
```

Source file: arith/arith_counter_gray.vhdl

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:

```
entity arith_counter_ring is
  generic (
    BITS            : positive;
    INVERT_FEEDBACK : boolean   := FALSE                                    -- FALSE -> ring counte
  );
  port (
    Clock    : in  std_logic;                                               -- Clock
    Reset    : in  std_logic;                                               -- Reset
    seed     : in  std_logic_vector(BITS - 1 downto 0) := (others => '0'); -- initial counter vecto
    inc      : in  std_logic                           := '0';             -- increment counter
    dec      : in  std_logic                           := '0';             -- decrement counter
    value    : out std_logic_vector(BITS - 1 downto 0)                     -- counter value
  );
end entity;
```

Source file: arith/arith_counter_ring.vhdl

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:

```
entity arith_div is
  generic (
    A_BITS              : positive;          -- Dividend Width
    D_BITS              : positive;          -- Divisor Width
    RAPOW               : positive := 1;     -- Power of Compute Radix (2**RAPOW)
```

```
    PIPELINED          : boolean  := false  -- Computation Pipeline
  );
  port (
    -- Global Reset/Clock
    clk : in std_logic;
    rst : in std_logic;

    -- Ready / Start
    start : in  std_logic;
    ready : out std_logic;

    -- Arguments / Result (2's complement)
    A : in  std_logic_vector(A_BITS-1 downto 0);  -- Dividend
    D : in  std_logic_vector(D_BITS-1 downto 0);  -- Divisor
    Q : out std_logic_vector(A_BITS-1 downto 0);  -- Quotient
    R : out std_logic_vector(D_BITS-1 downto 0);  -- Remainder
    Z : out std_logic  -- Division by Zero
  );
end entity arith_div;
```

Source file: arith/arith_div.vhdl

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 <= '0' (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:

```
entity arith_firstone is
  generic (
    N : positive                                -- Length of Token Chain
  );
  port (
    tin  : in  std_logic := '1';                -- Enable:   Fed Token
    rqst : in  std_logic_vector(N-1 downto 0);  -- Request:  Token Requests
    grnt : out std_logic_vector(N-1 downto 0);  -- Grant:    Token Output
    tout : out std_logic;                       -- Inactive: Unused Token
    bin  : out std_logic_vector(log2ceil(N)-1 downto 0)  -- Binary Grant Index
  );
end entity arith_firstone;
```

Source file: arith/arith_firstone.vhdl

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

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:

```
entity arith_prefix_and is
  generic (
    N : positive
  );
  port (
    x : in  std_logic_vector(N-1 downto 0);
    y : out std_logic_vector(N-1 downto 0)
  );
end entity;
```

Source file: arith/arith_prefix_and.vhdl

arith_prefix_or

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

Entity Declaration:

```
entity arith_prefix_or is
  generic (
    N : positive
  );
  port (
    x : in  std_logic_vector(N-1 downto 0);
    y : out std_logic_vector(N-1 downto 0)
  );
end entity;
```

Source file: arith/arith_prefix_or.vhdl

arith_prng

This module implementes 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:

```
entity arith_prng is
  generic (
    BITS : positive          := 32;
    SEED : std_logic_vector := "0"
  );
  port (
    clk  : in std_logic;
    rst  : in std_logic;                                -- reset value to initial seed
    got  : in std_logic;                                -- the current value has been got, and a new
    val  : out std_logic_vector(BITS - 1 downto 0)   -- the pseudo-random number
  );
end entity;
```

Source file: arith/arith_prng.vhdl

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:

```
entity arith_same is
  generic (
    N : positive                              -- Input width
  );
  port (
    g : in  std_logic := '1';                -- Guard Input (!g => !y)
    x : in  std_logic_vector(N-1 downto 0); -- Input Vector
    y : out std_logic                         -- All-same Output
  );
end entity;
```

Source file: arith/arith_same.vhdl

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: `unsigned(arg) * MULS(msel) / DIVS(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:

```
entity arith_scaler is
  generic (
    MULS : T_POSVEC := (0 => 1);   -- The set of multipliers to choose from in scaling operations.
    DIVS : T_POSVEC := (0 => 1)    -- The set of divisors to choose from in scaling operations.
  );
  port (
    clk   : in std_logic;
    rst   : in std_logic;

    start : in  std_logic;           -- Start of Computation
    arg   : in std_logic_vector;     -- Fixed-point value to be scaled
    msel  : in  std_logic_vector(log2ceil(MULS'length)-1 downto 0) := (others => '0');
    dsel  : in  std_logic_vector(log2ceil(DIVS'length)-1 downto 0) := (others => '0');

    done  : out std_logic;           -- Completion
    res   : out std_logic_vector     -- Result
  );
end entity arith_scaler;
```

Source file: arith/arith_scaler.vhdl

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:

```
entity arith_shifter_barrel is
  generic (
    BITS        : positive    := 32
  );
  port (
    Input           : in  std_logic_vector(BITS - 1 downto 0);
    ShiftAmount     : in  std_logic_vector(log2ceilnz(BITS) - 1 downto 0);
    ShiftRotate     : in  std_logic;
    LeftRight       : in  std_logic;
    ArithmeticLogic : in  std_logic;
    Output          : out std_logic_vector(BITS - 1 downto 0)
  );
end entity;
```

Source file: arith/arith_shifter_barrel.vhdl

arith_sqrt

Iterative Square Root Extractor.

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

Entity Declaration:

```
entity arith_sqrt is
  generic (
    N : positive -- := 8                    -- Bit Width of Argument
  );
  port (
    -- Global Control
    rst : in std_logic;                 -- Reset (synchronous)
    clk : in std_logic;                 -- Clock

    -- Inputs
    arg  : in std_logic_vector(N-1 downto 0); -- Radicand
    start : in std_logic;                       -- Start Strobe

    -- Outputs
    sqrt : out std_logic_vector((N-1)/2 downto 0);  -- Result
    rdy : out std_logic                             -- Ready / Done
  );
end entity arith_sqrt;
```

Source file: arith/arith_sqrt.vhdl

2.4.5 bus

These are bus entities....

Sub-namespaces

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

Entities

- PoC.bus.Arbiter

stream

PoC.Stream modules ...

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:

```
entity stream_Buffer is
  generic (
    FRAMES            : positive                                    := 2;
    DATA_BITS         : positive                                    := 8;
    DATA_FIFO_DEPTH   : positive                                    := 8;
    META_BITS         : T_POSVEC                                    := (0 => 8);
    META_FIFO_DEPTH   : T_POSVEC                                    := (0 => 16)
  );
  port (
    Clock             : in  std_logic;
    Reset             : in  std_logic;
```

```
    -- IN Port
    In_Valid          : in  std_logic;
    In_Data           : in  std_logic_vector(DATA_BITS - 1 downto 0);
    In_SOF            : in  std_logic;
    In_EOF            : in  std_logic;
    In_Ack            : out std_logic;
    In_Meta_rst       : out std_logic;
    In_Meta_nxt       : out std_logic_vector(META_BITS'length - 1 downto 0);
    In_Meta_Data      : in  std_logic_vector(isum(META_BITS) - 1 downto 0);
    -- OUT Port
    Out_Valid         : out std_logic;
    Out_Data          : out std_logic_vector(DATA_BITS - 1 downto 0);
    Out_SOF           : out std_logic;
    Out_EOF           : out std_logic;
    Out_Ack           : in  std_logic;
    Out_Meta_rst      : in  std_logic;
    Out_Meta_nxt      : in  std_logic_vector(META_BITS'length - 1 downto 0);
    Out_Meta_Data     : out std_logic_vector(isum(META_BITS) - 1 downto 0)
  );
end entity;
```

Source file: bus/stream/stream_Buffer.vhdl

stream_DeMux

Todo

No documentation available.

Entity Declaration:

```
entity stream_DeMux is
  generic (
    PORTS             : positive                  := 2;
    DATA_BITS         : positive                  := 8;
    META_BITS         : natural                   := 8;
    META_REV_BITS     : natural                   := 2
  );
  port (
    Clock             : in  std_logic;
    Reset             : in  std_logic;
    -- Control interface
    DeMuxControl      : in  std_logic_vector(PORTS - 1 downto 0);
    -- IN Port
    In_Valid          : in  std_logic;
    In_Data           : in  std_logic_vector(DATA_BITS - 1 downto 0);
    In_Meta           : in  std_logic_vector(META_BITS - 1 downto 0);
    In_Meta_rev       : out std_logic_vector(META_REV_BITS - 1 downto 0);
    In_SOF            : in  std_logic;
    In_EOF            : in  std_logic;
    In_Ack            : out std_logic;
    -- OUT Ports
    Out_Valid         : out std_logic_vector(PORTS - 1 downto 0);
    Out_Data          : out T_SLM(PORTS - 1 downto 0, DATA_BITS - 1 downto 0);
    Out_Meta          : out T_SLM(PORTS - 1 downto 0, META_BITS - 1 downto 0);
    Out_Meta_rev      : in  T_SLM(PORTS - 1 downto 0, META_REV_BITS - 1 downto 0);
    Out_SOF           : out std_logic_vector(PORTS - 1 downto 0);
    Out_EOF           : out std_logic_vector(PORTS - 1 downto 0);
```

```
    Out_Ack             : in  std_logic_vector(PORTS - 1 downto 0)
  );
end entity;
```

Source file: bus/stream/stream_DeMux.vhdl

stream_Mux

Todo

No documentation available.

Entity Declaration:

```
entity stream_Mux is
  generic (
    PORTS               : positive                  := 2;
    DATA_BITS           : positive                  := 8;
    META_BITS           : natural                   := 8;
    META_REV_BITS       : natural                   := 2--;
--    WEIGHTS             : T_INTVEC                    := (1, 1)
  );
  port (
    Clock               : in  std_logic;
    Reset               : in  std_logic;
    -- IN Ports
    In_Valid            : in  std_logic_vector(PORTS - 1 downto 0);
    In_Data             : in  T_SLM(PORTS - 1 downto 0, DATA_BITS - 1 downto 0);
    In_Meta             : in  T_SLM(PORTS - 1 downto 0, META_BITS - 1 downto 0);
    In_Meta_rev         : out T_SLM(PORTS - 1 downto 0, META_REV_BITS - 1 downto 0);
    In_SOF              : in  std_logic_vector(PORTS - 1 downto 0);
    In_EOF              : in  std_logic_vector(PORTS - 1 downto 0);
    In_Ack              : out std_logic_vector(PORTS - 1 downto 0);
    -- OUT Port
    Out_Valid           : out std_logic;
    Out_Data            : out std_logic_vector(DATA_BITS - 1 downto 0);
    Out_Meta            : out std_logic_vector(META_BITS - 1 downto 0);
    Out_Meta_rev        : in  std_logic_vector(META_REV_BITS - 1 downto 0);
    Out_SOF             : out std_logic;
    Out_EOF             : out std_logic;
    Out_Ack             : in  std_logic
  );
end entity;
```

Source file: bus/stream/stream_Mux.vhdl

stream_Mirror

Todo

No documentation available.

Entity Declaration:

```
entity stream_Mirror is
  generic (
    PORTS             : positive                    := 2;
    DATA_BITS         : positive                    := 8;
    META_BITS         : T_POSVEC                    := (0 => 8);
    META_LENGTH       : T_POSVEC                    := (0 => 16)
  );
  port (
    Clock             : in  std_logic;
    Reset             : in  std_logic;
    -- IN Port
    In_Valid          : in  std_logic;
    In_Data           : in  std_logic_vector(DATA_BITS - 1 downto 0);
    In_SOF            : in  std_logic;
    In_EOF            : in  std_logic;
    In_Ack            : out std_logic;
    In_Meta_rst       : out std_logic;
    In_Meta_nxt       : out std_logic_vector(META_BITS'length - 1 downto 0);
    In_Meta_Data      : in  std_logic_vector(isum(META_BITS) - 1 downto 0);
    -- OUT Port
    Out_Valid         : out std_logic_vector(PORTS - 1 downto 0);
    Out_Data          : out T_SLM(PORTS - 1 downto 0, DATA_BITS - 1 downto 0);
    Out_SOF           : out std_logic_vector(PORTS - 1 downto 0);
    Out_EOF           : out std_logic_vector(PORTS - 1 downto 0);
    Out_Ack           : in  std_logic_vector(PORTS - 1 downto 0);
    Out_Meta_rst      : in  std_logic_vector(PORTS - 1 downto 0);
    Out_Meta_nxt      : in  T_SLM(PORTS - 1 downto 0, META_BITS'length - 1 downto 0);
    Out_Meta_Data     : out T_SLM(PORTS - 1 downto 0, isum(META_BITS) - 1 downto 0)
  );
end entity;
```

Source file: bus/stream/stream_Mirror.vhdl

stream_Sink

Todo

No documentation available.

Entity Declaration:

Source file: bus/stream/stream_Sink.vhdl

stream_Source

Todo

No documentation available.

Entity Declaration:

```
entity stream_Source is
  generic (
    TESTCASES          : T_SIM_STREAM_FRAMEGROUP_VECTOR_8
  );
  port (
    Clock              : in  std_logic;
    Reset              : in  std_logic;
    -- Control interface
    Enable             : in  std_logic;
    -- OUT Port
    Out_Valid          : out std_logic;
    Out_Data           : out T_SLV_8;
    Out_SOF            : out std_logic;
    Out_EOF            : out std_logic;
    Out_Ack            : in  std_logic
  );
end entity;
```

Source file: bus/stream/stream_Source.vhdl

stream_FrameGenerator

Todo

No documentation available.

Entity Declaration:

```
entity stream_FrameGenerator is
  generic (
    DATA_BITS          : positive                          := 8;
    WORD_BITS          : positive                          := 16;
    APPEND             : T_FRAMEGEN_APPEND                 := FRAMEGEN_APP_NONE;
    FRAMEGROUPS        : T_FRAMEGEN_FRAMEGROUP_VECTOR_8    := (0 => C_FRAMEGEN_FRAMEGROUP_EMPTY)
  );
  port (
    Clock              : in  std_logic;
    Reset              : in  std_logic;
    -- CSE interface
    Command            : in  T_FRAMEGEN_COMMAND;
    Status             : out T_FRAMEGEN_STATUS;
    -- Control interface
    Pause              : in  T_SLV_16;
    FrameGroupIndex    : in  T_SLV_8;
    FrameIndex         : in  T_SLV_8;
    Sequences          : in  T_SLV_16;
    FrameLength        : in  T_SLV_16;
    -- OUT Port
    Out_Valid          : out std_logic;
    Out_Data           : out std_logic_vector(DATA_BITS - 1 downto 0);
    Out_SOF            : out std_logic;
    Out_EOF            : out std_logic;
    Out_Ack            : in  std_logic
  );
end entity;
```

Source file: bus/stream/stream_FrameGenerator.vhdl

wb

WishBone modules ...

Entities:

ocram_wb

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 cyle 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

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

uart_wb

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

bus_Arbiter

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

- Round Robin (RR)

Entity Declaration:

```
entity bus_Arbiter is
  generic (
    STRATEGY                : string                        := "RR";       -- RR, LOT
    PORTS                   : positive                      := 1;
    WEIGHTS                 : T_INTVEC                      := (0 => 1);
    OUTPUT_REG              : boolean                       := TRUE
  );
  port (
    Clock                   : in  std_logic;
    Reset                   : in  std_logic;

    Arbitrate               : in  std_logic;
    Request_Vector          : in  std_logic_vector(PORTS - 1 downto 0);

    Arbitrated              : out std_logic;
    Grant_Vector            : out std_logic_vector(PORTS - 1 downto 0);
    Grant_Index             : out std_logic_vector(log2ceilnz(PORTS) - 1 downto 0)
  );
end entity;
```

Source file: bus/bus_Arbiter.vhdl

2.4.6 cache

These are cache entities....

Entities

- PoC.cache.par
- PoC.cache.tagunit_par
- PoC.cache.tagunit_seq

cache_par

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 outputed 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 outputed on `CacheLineOut` and the old tag on `OldAddress`, both with a latency of one clock cycle.

Entity Declaration:

```
entity cache_par is
  generic (
    REPLACEMENT_POLICY : string   := "LRU";
    CACHE_LINES        : positive := 32;--1024;
    ASSOCIATIVITY      : positive := 32;--4;
    ADDRESS_BITS       : positive := 8; --32-6;
    DATA_BITS          : positive := 8  --64*8
  );
  port (
    Clock : in std_logic;
    Reset : in std_logic;

    Request     : in std_logic;
    ReadWrite   : in std_logic;
    Invalidate  : in std_logic;
    Replace     : in std_logic;
    Address     : in std_logic_vector(ADDRESS_BITS - 1 downto 0);

    CacheLineIn  : in  std_logic_vector(DATA_BITS - 1 downto 0);
    CacheLineOut : out std_logic_vector(DATA_BITS - 1 downto 0);
    CacheHit     : out std_logic := '0';
    CacheMiss    : out std_logic := '0';
    OldAddress   : out std_logic_vector(ADDRESS_BITS - 1 downto 0)
  );
end entity;
```

Source file: cache/cache_par.vhdl

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:

```
entity cache_replacement_policy is
  generic (
    REPLACEMENT_POLICY : string   := "LRU";
    CACHE_WAYS         : positive := 32
  );
  port (
    Clock : in std_logic;
    Reset : in std_logic;

    -- replacement interface
    Replace    : in  std_logic;
    ReplaceWay : out std_logic_vector(log2ceilnz(CACHE_WAYS) - 1 downto 0);

    -- cacheline usage update interface
    TagAccess  : in std_logic;
    ReadWrite  : in std_logic;
    Invalidate : in std_logic;
    HitWay     : in std_logic_vector(log2ceilnz(CACHE_WAYS) - 1 downto 0)
  );
end entity;
```

Source file: cache/cache_replacement_policy.vhdl

cache_tagunit_par

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

Command thruth 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. 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 `NewAddress` 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.

Entity Declaration:

```
entity cache_tagunit_par is
  generic (
    REPLACEMENT_POLICY : string   := "LRU";
    CACHE_LINES        : positive := 32;
    ASSOCIATIVITY      : positive := 32;
    ADDRESS_BITS          : positive := 8
  );
  port (
    Clock : in std_logic;
    Reset : in std_logic;

    Replace          : in  std_logic;
    ReplaceLineIndex : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto 0);
    NewAddress       : in  std_logic_vector(ADDRESS_BITS - 1 downto 0);
    OldAddress       : out std_logic_vector(ADDRESS_BITS - 1 downto 0);

    Request    : in  std_logic;
    ReadWrite  : in  std_logic;
    Invalidate : in  std_logic;
    Address    : in  std_logic_vector(ADDRESS_BITS - 1 downto 0);
    LineIndex  : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto 0);
    TagHit     : out std_logic;
    TagMiss    : out std_logic
  );
end entity;
```

Source file: cache/cache_tagunit_par.vhdl

cache_tagunit_seq

Todo

No documentation available.

Entity Declaration:

```
entity cache_tagunit_seq is
  generic (
    REPLACEMENT_POLICY : string       := "LRU";
    CACHE_LINES        : positive     := 32;
    ASSOCIATIVITY      : positive     := 32;
    TAG_BITS           : positive     := 128;
    CHUNK_BITS         : positive     := 8;
    TAG_BYTE_ORDER     : T_BYTE_ORDER := LITTLE_ENDIAN;
    USE_INITIAL_TAGS   : boolean      := false;
    INITIAL_TAGS       : T_SLM        := (0 downto 0 => (127 downto 0 => '0'))
  );
  port (
    Clock : in std_logic;
    Reset : in std_logic;

    Replace             : in  std_logic;
    Replaced            : out std_logic;
    Replace_NewTag_rst  : out std_logic;
    Replace_NewTag_rev  : out std_logic;
    Replace_NewTag_nxt  : out std_logic;
    Replace_NewTag_Data : in  std_logic_vector(CHUNK_BITS - 1 downto 0);
    Replace_NewIndex    : out std_logic_vector(log2ceilnz(CACHE_LINES) - 1 downto 0);
```

```
23
24      Request             : in  std_logic;
25      Request_ReadWrite   : in  std_logic;
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

2.4.7 comm

These are communication entities....

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

Source file: comm/comm_crc.vhdl

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:

```
entity comm_scramble is
  generic (
    GEN  : bit_vector;        -- Generator Polynomial (little endian)
    BITS : positive           -- Width of Mask Bits to be computed in parallel in each step
  );
  port (
    clk  : in  std_logic;     -- Clock

    set  : in  std_logic;     -- Set LFSR to value provided on din
    din  : in  std_logic_vector(GEN'length-2 downto 0) := (others => '0');

    step : in  std_logic;     -- Compute a Mask Output
    mask : out std_logic_vector(BITS-1 downto 0)
  );
end entity comm_scramble;
```

Source file: comm/comm_scramble.vhdl

2.4.8 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, acknoledge 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).
- `PoC.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)

Package

This package holds all component declarations for this namespace.

Source file: fifo/fifo.pkg.vhdl

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:

```
entity fifo_cc_got is
  generic (
    D_BITS          : positive;          -- Data Width
    MIN_DEPTH       : positive;          -- Minimum FIFO Depth
    DATA_REG        : boolean := false;  -- Store Data Content in Registers
    STATE_REG       : boolean := false;  -- Registered Full/Empty Indicators
    OUTPUT_REG      : boolean := false;  -- Registered FIFO Output
    ESTATE_WR_BITS  : natural := 0;      -- Empty State Bits
    FSTATE_RD_BITS  : natural := 0       -- Full State Bits
  );
  port (
    -- Global Reset and Clock
    rst, clk : in  std_logic;

    -- Writing Interface
    put      : in  std_logic;                                -- Write Request
    din      : in  std_logic_vector(D_BITS-1 downto 0);      -- Input Data
    full     : out std_logic;
    estate_wr : out std_logic_vector(imax(0, ESTATE_WR_BITS-1) downto 0);

    -- Reading Interface
    got      : in  std_logic;                                -- Read Completed
    dout     : out std_logic_vector(D_BITS-1 downto 0);      -- Output Data
    valid    : out std_logic;
```

```
    fstate_rd : out std_logic_vector(imax(0, FSTATE_RD_BITS-1) downto 0)
  );
end entity fifo_cc_got;
```

Source file: fifo/fifo_cc_got.vhdl

See also:

PoC.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.

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:

```
entity fifo_cc_got_tempgot is
  generic (
    D_BITS          : positive;           -- Data Width
    MIN_DEPTH       : positive;           -- Minimum FIFO Depth
    DATA_REG        : boolean := false;   -- Store Data Content in Registers
    STATE_REG       : boolean := false;   -- Registered Full/Empty Indicators
    OUTPUT_REG      : boolean := false;   -- Registered FIFO Output
    ESTATE_WR_BITS  : natural := 0;       -- Empty State Bits
    FSTATE_RD_BITS  : natural := 0        -- Full State Bits
  );
  port (
    -- Global Reset and Clock
    rst, clk : in  std_logic;

    -- Writing Interface
    put        : in  std_logic;                             -- Write Request
    din        : in  std_logic_vector(D_BITS-1 downto 0);   -- Input Data
```

```
    full       : out std_logic;
    estate_wr  : out std_logic_vector(imax(0, ESTATE_WR_BITS-1) downto 0);

    -- Reading Interface
    got        : in  std_logic;                              -- Read Completed
    dout       : out std_logic_vector(D_BITS-1 downto 0);    -- Output Data
    valid      : out std_logic;
    fstate_rd  : out std_logic_vector(imax(0, FSTATE_RD_BITS-1) downto 0);

    commit     : in  std_logic;
    rollback   : in  std_logic
  );
end entity fifo_cc_got_tempgot;
```

Source file: fifo/fifo_cc_got_tempgot.vhdl

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:

```
entity fifo_cc_got_tempput is
  generic (
    D_BITS          : positive;           -- Data Width
    MIN_DEPTH       : positive;           -- Minimum FIFO Depth
    DATA_REG        : boolean := false;   -- Store Data Content in Registers
    STATE_REG       : boolean := false;   -- Registered Full/Empty Indicators
    OUTPUT_REG      : boolean := false;   -- Registered FIFO Output
    ESTATE_WR_BITS  : natural := 0;       -- Empty State Bits
    FSTATE_RD_BITS  : natural := 0        -- Full State Bits
  );
  port (
    -- Global Reset and Clock
    rst, clk : in  std_logic;

```

```
    -- Writing Interface
    put       : in  std_logic;                              -- Write Request
    din       : in  std_logic_vector(D_BITS-1 downto 0);    -- Input Data
    full      : out std_logic;
    estate_wr : out std_logic_vector(imax(0, ESTATE_WR_BITS-1) downto 0);

    commit    : in  std_logic;
    rollback  : in  std_logic;

    -- Reading Interface
    got       : in  std_logic;                              -- Read Completed
    dout      : out std_logic_vector(D_BITS-1 downto 0);    -- Output Data
    valid     : out std_logic;
    fstate_rd : out std_logic_vector(imax(0, FSTATE_RD_BITS-1) downto 0)
  );
end entity fifo_cc_got_tempput;
```

Source file: fifo/fifo_cc_got_tempput.vhdl

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:

```
entity fifo_glue is
  generic (
    D_BITS : positive                                 -- Data Width
  );
  port (
    -- Control
    clk : in std_logic;                               -- Clock
    rst : in std_logic;                               -- Synchronous Reset

    -- Input
    put : in  std_logic;                              -- Put Value
    di  : in  std_logic_vector(D_BITS-1 downto 0);    -- Data Input
    ful : out std_logic;                              -- Full

    -- Output
    vld : out std_logic;                              -- Data Available
    do  : out std_logic_vector(D_BITS-1 downto 0);    -- Data Output
    got : in  std_logic                               -- Data Consumed
  );
end entity fifo_glue;
```

Source file: fifo/fifo_glue.vhdl

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:

```
entity fifo_ic_assembly is
  generic (
    D_BITS : positive;                  -- Data Width
    A_BITS : positive;                  -- Address Bits
    G_BITS : positive                   -- Generation Guard Bits
  );
  port (
    -- Write Interface
    clk_wr : in std_logic;
    rst_wr : in std_logic;

    -- Only write addresses in the range [base, base+2**(A_BITS-G_BITS)) are
    -- acceptable. This is equivalent to the test
    --   tmp(A_BITS-1 downto A_BITS-G_BITS) = 0 where tmp = addr - base.
    -- Writes performed outside the allowable range will assert the failure
    -- indicator, which will stick until the next reset.
    -- No write is to be performed before base turns zero (0) for the first
    -- time.
    base   : out std_logic_vector(A_BITS-1 downto 0);
    failed : out std_logic;

    addr : in  std_logic_vector(A_BITS-1 downto 0);
    din  : in  std_logic_vector(D_BITS-1 downto 0);
    put  : in  std_logic;

    -- Read Interface
    clk_rd : in std_logic;
    rst_rd : in std_logic;

    dout : out std_logic_vector(D_BITS-1 downto 0);
    vld  : out std_logic;
    got  : in  std_logic
  );
end entity fifo_ic_assembly;
```

Source file: fifo/fifo_ic_assembly.vhdl

fifo_ic_got

Independent clocks meens 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:

```
entity fifo_ic_got is
  generic (
    D_BITS         : positive;           -- Data Width
    MIN_DEPTH      : positive;           -- Minimum FIFO Depth
    DATA_REG       : boolean := false;   -- Store Data Content in Registers
    OUTPUT_REG     : boolean := false;   -- Registered FIFO Output
    ESTATE_WR_BITS : natural := 0;       -- Empty State Bits
    FSTATE_RD_BITS : natural := 0        -- Full State Bits
  );
  port (
    -- Write Interface
    clk_wr    : in  std_logic;
    rst_wr    : in  std_logic;
    put       : in  std_logic;
    din       : in  std_logic_vector(D_BITS-1 downto 0);
    full      : out std_logic;
    estate_wr : out std_logic_vector(imax(ESTATE_WR_BITS-1, 0) downto 0);

    -- Read Interface
    clk_rd    : in  std_logic;
    rst_rd    : in  std_logic;
    got       : in  std_logic;
    valid     : out std_logic;
    dout      : out std_logic_vector(D_BITS-1 downto 0);
    fstate_rd : out std_logic_vector(imax(FSTATE_RD_BITS-1, 0) downto 0)
  );
end entity fifo_ic_got;
```

Source file: fifo/fifo_ic_got.vhdl

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:

```
entity fifo_shift is
  generic (
    D_BITS    : positive;              -- Data Width
    MIN_DEPTH : positive               -- Minimum FIFO Size in Words
  );
  port (
    -- Global Control
    clk : in std_logic;
    rst : in std_logic;

    -- 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

2.4.9 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

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

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.

```
            __      _____      _____      __
Clock         |_____|     |_____|     |_____|
Pad         < 0 >< 1 >< 2 >< 3 >< 4 >< 5 >
DataIn_low    ... >< 0      >< 2      ><
DataIn_high   ... >< 1      >< 3      ><

< i > is the value of the i-th data bit on the line.
```

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"FFFFFFFF"
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

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.

```
            __      _____      _____      __
Clock         |_____|     |_____|     |_____|
Pad         < 0 >< 1 >< 2 >< 3 >< 4 >< 5 >
DataIn_low    ... >< 0      >< 2      ><
DataIn_high   ... >< 1      >< 3      ><
```

```
< i > is the value of the i-th data bit on the line.
```

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

Entity Declaration:

```
entity ddrio_inout is
  generic (
    BITS          : positive
  );
  port (
    ClockOut        : in    std_logic;
    ClockOutEnable  : in    std_logic;
    OutputEnable    : in    std_logic;
    DataOut_high    : in    std_logic_vector(BITS - 1 downto 0);
    DataOut_low     : in    std_logic_vector(BITS - 1 downto 0);

    ClockIn         : in    std_logic;
    ClockInEnable   : in    std_logic;
    DataIn_high     : out   std_logic_vector(BITS - 1 downto 0);
    DataIn_low      : out   std_logic_vector(BITS - 1 downto 0);

    Pad             : inout std_logic_vector(BITS - 1 downto 0)
  );
end entity;
```

Source file: io/ddrio/ddrio_inout.vhdl

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:

```
entity ddrio_out is
  generic (
    NO_OUTPUT_ENABLE    : boolean     := false;
    BITS                : positive;
    INIT_VALUE          : bit_vector  := x"FFFFFFFF"
  );
  port (
    Clock         : in  std_logic;
    ClockEnable   : in  std_logic := '1';
    OutputEnable  : in  std_logic := '1';
    DataOut_high  : in  std_logic_vector(BITS - 1 downto 0);
    DataOut_low   : in  std_logic_vector(BITS - 1 downto 0);
    Pad           : out std_logic_vector(BITS - 1 downto 0)
```

```
14    );
15  end entity;
```

Source file: io/ddrio/ddrio_out.vhdl

iic

These are I2C entities....

iic_BusController

The I2C BusController transmitts 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

iic_Controller

The I2C Controller transmitts 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

iic_Switch_PCA9548A

Todo

No documentation available. TODO

Entity Declaration:

Source file: io/iic/iic_Switch_PCA9548A.vhdl

jtag

These are JTAG entities....

lcd

These are LCD entities....

lcd_LCDBuffer

Todo

No documentation available.

Entity Declaration:

Source file: io/lcd/lcd_LCDBuffer.vhdl

lcd_LCDBusController

Todo

No documentation available.

Entity Declaration:

Source file: io/lcd/lcd_LCDBusController.vhdl

lcd_LCDController_KS0066U

Todo

No documentation available.

Entity Declaration:

Source file: io/lcd/lcd_LCDController_KS0066U.vhdl

lcd_LCDSynchronizer

Todo

No documentation available.

Entity Declaration:

Source file: io/lcd/lcd_LCDSynchronizer.vhdl

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

mdio_Controller

Todo

No documentation available.

Entity Declaration:

Source file: io/mdio/mdio_Controller.vhdl

mdio_IIC_Adapter

Todo

No documentation available.

Entity Declaration:

Source file: io/mdio/mdio_IIC_Adapter.vhdl

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

pio

These are Pmod entities....

pio_in

Entity Declaration:

Source file: io/pio/pio_in.vhdl

pio_out

Entity Declaration:

Source file: io/pio/pio_out.vhdl

pio_fifo_in

Entity Declaration:

Source file: io/pio/pio_fifo_in.vhdl

pio_fifo_out

Entity Declaration:

Source file: io/pio/pio_fifo_out.vhdl

pmod

These are Pmod entities....

Entities

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

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:

```
entity pmod_KYPD is
  generic (
    CLOCK_FREQ     : FREQ         := 100 MHz;
    SCAN_FREQ      : FREQ         := 1 kHz;
    BOUNCE_TIME    : time         := 10 ms
  );
  port (
    Clock          : in  std_logic;
    Reset          : in  std_logic;
    -- Matrix interface
    Keys           : out T_PMOD_KYPD_KEYPAD;
    -- KeyPad interface
    Columns_n      : out std_logic_vector(3 downto 0);
    Rows_n         : in  std_logic_vector(3 downto 0)
  );
end entity;
```

Source file: io/pmod/pmod_KYPD.vhdl

pmod_SSD

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

– code-block:. none

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:

```
entity pmod_SSD is
  generic (
    CLOCK_FREQ     : FREQ     := 100 MHz;
    REFRESH_RATE   : FREQ     := 1 kHz
  );
  port (
    Clock      : in  std_logic;

    Digit0     : in  std_logic_vector(3 downto 0);
    Digit1     : in  std_logic_vector(3 downto 0);

    SSD        : out T_PMOD_SSD_PINS
  );
end entity;
```

Source file: io/pmod/pmod_SSD.vhdl

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:

```
entity pmod_USBUART is
  generic (
    CLOCK_FREQ     : FREQ     := 100 MHz;
    BAUDRATE       : BAUD     := 115200 Bd
  );
  port (
    Clock      : in  std_logic;
    Reset      : in  std_logic;

    TX_put     : in  std_logic;
    TX_Data    : in  std_logic_vector(7 downto 0);
    TX_Full    : out std_logic;

    RX_Valid   : out std_logic;
    RX_Data    : out std_logic_vector(7 downto 0);
    RX_got     : in  std_logic;

    UART_TX    : out std_logic;
    UART_RX    : in  std_logic;
    UART_RTS   : out std_logic;
    UART_CTS   : in  std_logic
  );
end entity;
```

Source file: io/pmod/pmod_USBUART.vhdl

ps2

These are PS/2 entities....

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

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:

```
entity uart_bclk is
  generic (
    CLOCK_FREQ     : FREQ       := 100 MHz;
    BAUDRATE       : BAUD       := 115200 Bd
  );
  port (
    clk        : in  std_logic;
    rst        : in  std_logic;
    bclk       : out std_logic;
    bclk_x8    : out std_logic
  );
end entity;
```

Source file: io/uart/uart_bclk.vhdl

uart_rx

UART Receiver: 1 Start + 8 Data + 1 Stop

Entity Declaration:

```
entity uart_rx is
  generic (
    SYNC_DEPTH : natural := 2  -- use zero for already clock-synchronous rx
  );
  port (
    -- Global Control
    clk : in std_logic;
    rst : in std_logic;

    -- Bit Clock and RX Line
    bclk_x8 : in std_logic;   -- bit clock, eight strobes per bit length
    rx      : in std_logic;

    -- Byte Stream Output
    do  : out std_logic_vector(7 downto 0);
    stb : out std_logic
  );
end entity;
```

Source file: io/uart/uart_rx.vhdl

uart_tx

UART Transmitter: 1 Start + 8 Data + 1 Stop

Entity Declaration:

```
entity uart_tx is
  port (
    -- Global Control
    clk : in std_logic;
```

```
    rst : in std_logic;

    -- Bit Clock and TX Line
    bclk : in  std_logic;  -- bit clock, one strobe each bit length
    tx   : out std_logic;

    -- Byte Stream Input
    di  : in  std_logic_vector(7 downto 0);
    put : in  std_logic;
    ful : out std_logic
  );
end entity;
```

Source file: io/uart/uart_tx.vhdl

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:

```
entity uart_fifo is
  generic (
    -- Communication Parameters
    CLOCK_FREQ              : FREQ;
    BAUDRATE                : BAUD;
    ADD_INPUT_SYNCHRONIZERS : boolean                       := TRUE;

    -- Buffer Dimensioning
    TX_MIN_DEPTH            : positive                      := 16;
    TX_ESTATE_BITS          : natural                       :=  0;
    RX_MIN_DEPTH            : positive                      := 16;
    RX_FSTATE_BITS          : natural                       :=  0;

    -- Flow Control
    FLOWCONTROL             : T_IO_UART_FLOWCONTROL_KIND    := UART_FLOWCONTROL_NONE;
    SWFC_XON_CHAR           : std_logic_vector(7 downto 0)  := x"11"; -- ^Q
    SWFC_XON_TRIGGER        : real                          := 0.0625;
    SWFC_XOFF_CHAR          : std_logic_vector(7 downto 0)  := x"13"; -- ^S
    SWFC_XOFF_TRIGGER       : real                          := 0.75
  );
  port (
    Clock         : in  std_logic;
    Reset         : in  std_logic;

    -- FIFO interface
    TX_put        : in  std_logic;
    TX_Data       : in  std_logic_vector(7 downto 0);
    TX_Full       : out std_logic;
    TX_EmptyState : out std_logic_vector(imax(0, TX_ESTATE_BITS-1) downto 0);

    RX_Valid      : out std_logic;
    RX_Data       : out std_logic_vector(7 downto 0);
    RX_got        : in  std_logic;
    RX_FullState  : out std_logic_vector(imax(0, RX_FSTATE_BITS-1) downto 0);
    RX_Overflow   : out std_logic;

```

```
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

vga

These are VGA entities....

vga_phy

The clock frequency must be the same as used for the timing module.

The number of color-bits per pixel can be configured with the generic "COLOR_BITS". The format of the pixel data is defined the picture generator in use.

Entity Declaration:

Source file: io/vga/vga_phy.vhdl

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.

0x49 PM 0xC0 Enable DVI, RGB bypass off or 0xD0 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 when using SMT (VS0, HS0)

or 0x90 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

vga_timing

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)

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 respectivly 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

Package

This package holds all component declarations for this namespace.

Source file: io/io.pkg.vhdl

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:

```
entity io_7SegmentMux_BCD is
  generic (
    CLOCK_FREQ       : FREQ         := 100 MHz;
    REFRESH_RATE     : FREQ         := 1 kHz;
    DIGITS           : positive     := 4
  );
  port (
    Clock            : in  std_logic;

    BCDDigits        : in  T_BCD_VECTOR(DIGITS - 1 downto 0);
    BCDDots          : in  std_logic_vector(DIGITS - 1 downto 0);

    SegmentControl   : out std_logic_vector(7 downto 0);
    DigitControl     : out std_logic_vector(DIGITS - 1 downto 0)
```

```
  );
end entity;
```

Source file: io/io_7SegmentMux_BCD.vhdl

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:

```
entity io_7SegmentMux_HEX is
  generic (
    CLOCK_FREQ       : FREQ         := 100 MHz;
    REFRESH_RATE     : FREQ         := 1 kHz;
    DIGITS           : positive     := 4
  );
  port (
    Clock            : in  std_logic;

    HexDigits        : in  T_SLVV_4(DIGITS - 1 downto 0);
    HexDots          : in  std_logic_vector(DIGITS - 1 downto 0);

    SegmentControl   : out std_logic_vector(7 downto 0);
    DigitControl     : out std_logic_vector(DIGITS - 1 downto 0)
  );
end entity;
```

Source file: io/io_7SegmentMux_HEX.vhdl

io_Debounce

This module debounces several input pins preventing input changes following a previous one within the configured `BOUNCE_TIME` to pass. Internally, the forwarded state is locked for, at least, this `BOUNCE_TIME`. As the backing timer is restarted on every input fluctuation, the next passing input update must have seen a stabilized input.

The parameter `COMMON_LOCK` uses a single internal timer for all processed inputs. Thus, all inputs must stabilize before any one may pass changed. This option is usually fully acceptable for user inputs such as push buttons.

The parameter `ADD_INPUT_SYNCHRONIZERS` triggers the optional instantiation of a two-FF input synchronizer on each input bit.

Entity Declaration:

```
entity io_Debounce is
  generic (
    CLOCK_FREQ               : FREQ;
    BOUNCE_TIME              : time;
    BITS                     : positive := 1;
    INIT                     : std_logic_vector    := x"00000000"; -- initial state of Output
    ADD_INPUT_SYNCHRONIZERS  : boolean  := true;
    COMMON_LOCK              : boolean  := false
  );
  port (
    Clock    : in  std_logic;
    Reset    : in  std_logic             := '0';
    Input    : in  std_logic_vector(BITS-1 downto 0);
```

```
    Output   : out std_logic_vector(BITS-1 downto 0) := resize(descend(INIT), BITS)
  );
end entity;
```

Source file: io/io_Debounce.vhdl

io_FanControl

```
This module generates a PWM signal for a 3-pin (transistor controlled) or
4-pin fan header. The FPGAs temperature is read from device specific system
monitors (normal, user temperature, over temperature).

For example the Xilinx System Monitors are configured as follows:

                                                                |
Temp_ov  on=80  |       -       -       -       -       -       -       /-------/
                                                                |
Temp_ov off=60  |       -       -       -       -       -       /       -       -       -       -
                                                                |
                                                                |
Temp_us  on=35  |       -        /---/                                            |
Temp_us off=30  |       -       /       -       -|-     -       -       -       -       -       |
                                                                |  /              |
----------------|--------|------------|--------------|----------|---------
pwm =                                       |               min      |      medium          |
```

Entity Declaration:

```
entity io_FanControl is
  generic (
    CLOCK_FREQ               : FREQ;
    ADD_INPUT_SYNCHRONIZERS  : boolean     := TRUE;
    ENABLE_TACHO             : boolean     := FALSE
  );
  port (
    -- Global Control
    Clock                    : in  std_logic;
    Reset                    : in  std_logic;

    -- Fan Control derived from internal System Health Monitor
    Fan_PWM                  : out std_logic;

    -- Decoding of Speed Sensor (Requires ENABLE_TACHO)
    Fan_Tacho      : in  std_logic := 'X';
    TachoFrequency : out std_logic_vector(15 downto 0)
  );
end entity;
```

Source file: io/io_FanControl.vhdl

io_FrequencyCounter

Todo

No documentation available.

Entity Declaration:

```
entity io_FrequencyCounter is
  generic (
    CLOCK_FREQ                : FREQ                    := 100 MHz;
    TIMEBASE                  : time                    := 1 sec;
    RESOLUTION                : positive                := 8
  );
  port (
    Clock       : in  std_logic;
    Reset       : in  std_logic;
    FreqIn      : in  std_logic;
    FreqOut     : out std_logic_vector(RESOLUTION - 1 downto 0)
  );
end entity;
```

Source file: io/io_FrequencyCounter.vhdl

io_GlitchFilter

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

Entity Declaration:

```
entity io_GlitchFilter is
  generic (
    HIGH_SPIKE_SUPPRESSION_CYCLES     : natural       := 5;
    LOW_SPIKE_SUPPRESSION_CYCLES      : natural       := 5
  );
  port (
    Clock    : in  std_logic;
    Input    : in  std_logic;
    Output   : out std_logic
  );
end entity;
```

Source file: io/io_GlitchFilter.vhdl

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:

```
entity io_KeyPadScanner is
  generic (
    CLOCK_FREQ               : FREQ        := 100 MHz;
    SCAN_FREQ                : FREQ        := 1 kHz;
    ROWS                     : positive    := 4;
    COLUMNS                  : positive    := 4;
    ADD_INPUT_SYNCHRONIZERS  : boolean     := TRUE
  );
  port (
    Clock          : in  std_logic;
    Reset          : in  std_logic;
```

```
    -- Matrix interface
    KeyPadMatrix  : out T_SLM(COLUMNS - 1 downto 0, ROWS - 1 downto 0);
    -- KeyPad interface
    ColumnVector  : out std_logic_vector(COLUMNS - 1 downto 0);
    RowVector     : in  std_logic_vector(ROWS - 1 downto 0)
  );
end entity;
```

Source file: io/io_KeyPadScanner.vhdl

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:

```
entity io_PulseWidthModulation is
  generic (
    CLOCK_FREQ                : FREQ                    := 100 MHz;
    PWM_FREQ                  : FREQ                    := 1 kHz;
    PWM_RESOLUTION            : positive                := 8
  );
  port (
    Clock       : in  std_logic;
    Reset       : in  std_logic;
    PWMIn       : in  std_logic_vector(PWM_RESOLUTION - 1 downto 0);
    PWMOut      : out std_logic
  );
end entity;
```

Source file: io/io_PulseWidthModulation.vhdl

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 `log2ceilnz(imax(TIMING_TABLE)) + 1` 6-input LUTs.

Entity Declaration:

```
entity io_TimingCounter is
  generic (
    TIMING_TABLE  : T_NATVEC                                          -- timing table
  );
  port (
    Clock         : in  std_logic;                                    -- clock
    Enable        : in  std_logic;                                    -- enable counter
    Load          : in  std_logic;                                    -- load Timing Value from T
    Slot          : in  natural range 0 to (TIMING_TABLE'length - 1); --
    Timeout       : out std_logic                                     -- timing reached
  );
end entity;
```

Source file: io/io_TimingCounter.vhdl

2.4.10 mem

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

Sub-Namespaces

- `PoC.mem.ddr3` - DDR3 memory controllers
- PoC.mem.is61lv - ISSI - IS61LV SRAM controller
- PoC.mem.is61nlp - ISSI - IS61NLP SRAM controller
- 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

is61lv

These are IS61LV entities....

is61nlp

These are IS61NLP entities....

lut

These are Lookup-Table entities....

lut_Sine

Todo

No documentation available.

Entity Declaration:

Source file: mem/lut/lut_Sine.vhdl

ocram

These are On-Chip RAM (OCRAM) entities...

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.tdp - An 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.

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:

```
entity ocram_sp is
  generic (
    A_BITS    : positive;                          -- number of address bits
    D_BITS    : positive;                          -- number of data bits
    FILENAME  : string    := ""                    -- file-name for RAM initialization
  );
  port (
    clk : in  std_logic;                           -- clock
    ce  : in  std_logic;                           -- clock enable
    we  : in  std_logic;                           -- write enable
    a   : in unsigned(A_BITS-1 downto 0);          -- address
    d   : in std_logic_vector(D_BITS-1 downto 0);  -- write data
    q   : out std_logic_vector(D_BITS-1 downto 0)  -- read output
  );
end entity;
```

Source file: mem/ocram/ocram_sp.vhdl

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).

Note: This component is **deprecated**. Please use PoC.mem.ocram.tdp for new designs. This component has been provided because older FPGA compilers where 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". 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.

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.

Warning: The simulated behavior on RT-level is too optimistic. When reading at the write address always the new data will be returned.

Entity Declaration:

```
entity ocram_esdp is
  generic (
    A_BITS    : positive;                          -- number of address bits
    D_BITS    : positive;                          -- number of data bits
    FILENAME  : string    := ""                    -- file-name for RAM initialization
  );
  port (
    clk1 : in std_logic;                           -- clock for 1st port
    clk2 : in std_logic;                           -- clock for 2nd port
    ce1 : in  std_logic;                           -- clock-enable for 1st port
    ce2 : in  std_logic;                           -- clock-enable for 2nd port
    we1 : in  std_logic;                           -- write-enable for 1st port
    a1   : in unsigned(A_BITS-1 downto 0);         -- address for 1st port
    a2   : in unsigned(A_BITS-1 downto 0);         -- address for 2nd port
    d1   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 1st port
    q1   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 1st port
    q2   : out std_logic_vector(D_BITS-1 downto 0)  -- read-data from 2nd port
  );
end entity;
```

Source file: mem/ocram/ocram_esdp.vhdl

ocram_sdp

Inferring / instantiating simple dual-port memory, with:

- dual clock, clock enable,
- 1 read port plus 1 write port.

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.

> **Warning:** The simulated behavior on RT-level is too optimistic. The mixed-port read-during-write behavior is only valid if the read and write clock are in phase. Otherwise, simulation will always show known data.

Todo

Implement correct behavior for RT-level simulation.

Entity Declaration:

```
entity ocram_sdp is
  generic (
    A_BITS     : positive;                                  -- number of address bits
    D_BITS     : positive;                                  -- number of data bits
    FILENAME   : string    := ""                            -- file-name for RAM initialization
  );
  port (
    rclk   : in  std_logic;                                 -- read clock
    rce    : in  std_logic;                                 -- read clock-enable
    wclk   : in  std_logic;                                 -- write clock
    wce    : in  std_logic;                                 -- write clock-enable
    we     : in  std_logic;                                 -- write enable
    ra     : in  unsigned(A_BITS-1 downto 0);               -- read address
    wa     : in  unsigned(A_BITS-1 downto 0);               -- write address
    d      : in  std_logic_vector(D_BITS-1 downto 0);       -- data in
    q      : out std_logic_vector(D_BITS-1 downto 0)        -- data out
  );
end entity;
```

Source file: mem/ocram/ocram_sdp.vhdl

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

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". 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.

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.

Warning: The simulated behavior on RT-level is too optimistic. When reading at the write address always the new data will be returned.

Todo

Implement correct behavior for RT-level simulation.

Entity Declaration:

```
entity ocram_tdp is
  generic (
    A_BITS    : positive;                          -- number of address bits
    D_BITS    : positive;                          -- number of data bits
    FILENAME  : string    := ""                    -- file-name for RAM initialization
  );
  port (
    clk1 : in std_logic;                           -- clock for 1st port
    clk2 : in std_logic;                           -- clock for 2nd port
    ce1 : in  std_logic;                           -- clock-enable for 1st port
    ce2 : in  std_logic;                           -- clock-enable for 2nd port
    we1 : in  std_logic;                           -- write-enable for 1st port
    we2 : in  std_logic;                           -- write-enable for 2nd port
    a1   : in unsigned(A_BITS-1 downto 0);         -- address for 1st port
    a2   : in unsigned(A_BITS-1 downto 0);         -- address for 2nd port
    d1   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 1st port
    d2   : in std_logic_vector(D_BITS-1 downto 0); -- write-data for 2nd port
    q1   : out std_logic_vector(D_BITS-1 downto 0); -- read-data from 1st port
    q2   : out std_logic_vector(D_BITS-1 downto 0)  -- read-data from 2nd port
  );
end entity;
```

Source file: mem/ocram/ocram_tdp.vhdl

ocrom

These are On-Chip ROM (OCROM) entities....

Namespace *PoC.mem.ocrom*

The namespace *PoC.mem.ocrom* offers different on-chip ROM abstractions.

Package(s)

The package [*ocrom*][ocrom.pkg] holds all component declarations for this namespace.

```
`VHDL library PoC; use PoC.ocrom.all; `
```

Entities

- [*ocrom_sp*][ocrom_sp] is a on-chip RAM with a single port interface.
- [*ocrom_dp*][ocrom_dp] is a on-chip RAM with a dual port interface.

[ocrom.pkg]: https://github.com/VLSI-EDA/PoC/blob/master/src/mem/ocrom/ocrom.pkg.vhdl
[ocrom_sp]: https://github.com/VLSI-EDA/PoC/blob/master/src/mem/ocrom/ocrom_sp.vhdl
[ocrom_dp]: https://github.com/VLSI-EDA/PoC/blob/master/src/mem/ocrom/ocrom_dp.vhdl

ocrom_sp

Inferring / instantiating single-port read-only memory

- single clock, clock enable
- 1 read port

Entity Declaration:

```
entity ocrom_sp is
  generic (
    A_BITS    : positive;
    D_BITS    : positive;
    FILENAME  : string    := ""
  );
  port (
    clk : in  std_logic;
    ce  : in  std_logic;
    a   : in  unsigned(A_BITS-1 downto 0);
    q   : out std_logic_vector(D_BITS-1 downto 0)
  );
end entity;
```

Source file: mem/ocrom/ocrom_sp.vhdl

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:

```
entity ocrom_dp is
  generic (
    A_BITS    : positive;
    D_BITS    : positive;
    FILENAME  : string    := ""
  );
  port (
    clk1 : in std_logic;
```

```
    clk2 : in std_logic;
    ce1 : in  std_logic;
    ce2 : in  std_logic;
    a1   : in unsigned(A_BITS-1 downto 0);
    a2   : in unsigned(A_BITS-1 downto 0);
    q1   : out std_logic_vector(D_BITS-1 downto 0);
    q2   : out std_logic_vector(D_BITS-1 downto 0)
  );
end entity;
```

Source file: mem/ocrom/ocrom_dp.vhdl

sdram

These are SDRAM entities....

Package

This package holds all component declarations for this namespace.

Source file: mem/mem.pkg.vhdl

2.4.11 misc

The namespace `PoC.misc` offers different yet uncathegorized entities.

Sub-Namespaces

- 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
- PoC.misc.WordAligner

filter

These are filter entities....

Entities

- PoC.misc.filter.and
- PoC.misc.filter.mean
- PoC.misc.filter.or

filter_and

Todo

No documentation available.

Entity Declaration:

```
entity filter_and is
  generic (
    TAPS             : positive        := 4;        --
    INIT             : std_logic       := '0';      --
    ADD_OUTPUT_REG   : boolean         := FALSE     --
  );
  port (
    Clock            : in  std_logic;               -- clock
    DataIn           : in  std_logic;               -- data to filter
    DataOut          : out std_logic                -- filtered signal
  );
end entity;
```

Source file: misc/filter/filter_and.vhdl

filter_mean

Todo

No documentation available.

Entity Declaration:

```
entity filter_mean is
  generic (
    TAPS             : positive        := 4;        --
    INIT             : std_logic       := '1';      --
    ADD_OUTPUT_REG   : boolean         := FALSE     --
  );
  port (
    Clock            : in  std_logic;               -- clock
    DataIn           : in  std_logic;               -- data to filter
    DataOut          : out std_logic                -- filtered signal
  );
end entity;
```

Source file: misc/filter/filter_mean.vhdl

filter_or

Todo

No documentation available.

Entity Declaration:

```
entity filter_or is
  generic (
    TAPS            : positive        := 4;       --
    INIT            : std_logic       := '1';     --
    ADD_OUTPUT_REG  : boolean         := FALSE    --
  );
  port (
    Clock           : in  std_logic;              -- clock
    DataIn          : in  std_logic;              -- data to filter
    DataOut         : out std_logic               -- filtered signal
  );
end entity;
```

Source file: misc/filter/filter_or.vhdl

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

gearbox_down_cc

This module provides a downscaling gearbox with a common clock (cc) interface. It perfoems 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:

```
entity gearbox_down_cc is
  generic (
    INPUT_BITS            : positive  := 32;
    OUTPUT_BITS           : positive  := 24;
    META_BITS             : natural   := 0;
    ADD_INPUT_REGISTERS   : boolean   := FALSE;
    ADD_OUTPUT_REGISTERS  : boolean   := FALSE
  );
  port (
    Clock       : in  std_logic;

    In_Sync     : in  std_logic;
    In_Valid    : in  std_logic;
    In_Next     : out std_logic;
    In_Data     : in  std_logic_vector(INPUT_BITS - 1 downto 0);
    In_Meta     : in  std_logic_vector(META_BITS - 1 downto 0);

    Out_Sync    : out std_logic;
    Out_Valid   : out std_logic;
    Out_Data    : out std_logic_vector(OUTPUT_BITS - 1 downto 0);
```

```
    Out_Meta     : out std_logic_vector(META_BITS - 1 downto 0);
    Out_First    : out std_logic;
    Out_Last     : out std_logic
  );
end entity;
```

Source file: misc/gearbox/gearbox_down_cc.vhdl

gearbox_down_dc

This module provides a downscaling gearbox with a dependent clock (dc) interface. It perfoems 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:

```
entity gearbox_down_dc is
  generic (
    INPUT_BITS             : positive        := 32;                          -- input bits ('words
    OUTPUT_BITS            : positive        := 8;                           -- output bits ('byte
    OUTPUT_ORDER           : T_BIT_ORDER     := LSB_FIRST;                   -- LSB_FIRST: start a
    ADD_INPUT_REGISTERS    : boolean         := FALSE;                       -- add input register
    ADD_OUTPUT_REGISTERS   : boolean         := FALSE                        -- add output registe
  );
  port (
    Clock1                 : in  std_logic;                                  -- input clock domain
    Clock2                 : in  std_logic;                                  -- output clock domai
    In_Data                : in  std_logic_vector(INPUT_BITS - 1 downto 0);  -- input word
    Out_Data               : out std_logic_vector(OUTPUT_BITS - 1 downto 0)  -- output word
  );
end entity;
```

Source file: misc/gearbox/gearbox_down_dc.vhdl

gearbox_up_cc

This module provides a downscaling gearbox with a common clock (cc) interface. It perfoems 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:

```
entity gearbox_up_cc is
  generic (
    INPUT_BITS             : positive  := 24;
    OUTPUT_BITS            : positive  := 32;
    META_BITS              : natural   := 0;
    ADD_INPUT_REGISTERS    : boolean   := FALSE;
```

```
    ADD_OUTPUT_REGISTERS  : boolean   := FALSE
  );
  port (
    Clock       : in  std_logic;

    In_Sync     : in  std_logic;
    In_Valid    : in  std_logic;
    In_Data     : in  std_logic_vector(INPUT_BITS - 1 downto 0);
    In_Meta     : in  std_logic_vector(META_BITS - 1 downto 0);

    Out_Sync    : out std_logic;
    Out_Valid   : out std_logic;
    Out_Data    : out std_logic_vector(OUTPUT_BITS - 1 downto 0);
    Out_Meta    : out std_logic_vector(META_BITS - 1 downto 0);
    Out_First   : out std_logic;
    Out_Last    : out std_logic
  );
end entity;
```

Source file: misc/gearbox/gearbox_up_cc.vhdl

gearbox_up_dc

This module provides a upscaling gearbox with a dependent clock (dc) interface. It perfoems 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:

```
entity gearbox_up_dc is
  generic (
    INPUT_BITS            : positive        := 8;                                 -- input bit width
    INPUT_ORDER           : T_BIT_ORDER     := LSB_FIRST;                         -- LSB_FIRST: start a
    OUTPUT_BITS           : positive        := 32;                                -- output bit width
    ADD_INPUT_REGISTERS   : boolean         := FALSE                              -- add input register
  );
  port (
    Clock1                : in  std_logic;                                        -- input clock domain
    Clock2                : in  std_logic;                                        -- output clock domai
    In_Align              : in  std_logic;                                        -- align word (one cy
    In_Data               : in  std_logic_vector(INPUT_BITS - 1 downto 0);  -- input word
    Out_Data              : out std_logic_vector(OUTPUT_BITS - 1 downto 0); -- output word
    Out_Valid             : out std_logic                                         -- output is valid
  );
end entity;
```

Source file: misc/gearbox/gearbox_up_dc.vhdl

stat

These are stat entities....

Entities

- PoC.misc.stat.Average
- PoC.misc.stat.Histogram
- PoC.misc.stat.Maximum
- PoC.misc.stat.Minimum

stat_Average

Todo

No documentation available.

Entity Declaration:

```
entity stat_Average is
  generic (
    DATA_BITS      : positive     := 8;
    COUNTER_BITS   : positive     := 16
  );
  port (
    Clock          : in  std_logic;
    Reset          : in  std_logic;

    Enable         : in  std_logic;
    DataIn         : in  std_logic_vector(DATA_BITS - 1 downto 0);

    Count          : out std_logic_vector(COUNTER_BITS - 1 downto 0);
    Sum            : out std_logic_vector(COUNTER_BITS - 1 downto 0);
    Average        : out std_logic_vector(COUNTER_BITS - 1 downto 0);
    Valid          : out std_logic
  );
end entity;
```

Source file: misc/stat/stat_Average.vhdl

stat_Histogram

Todo

No documentation available.

Entity Declaration:

```
entity stat_Histogram is
  generic (
    DATA_BITS      : positive     := 16;
    COUNTER_BITS   : positive     := 16
  );
  port (
    Clock          : in  std_logic;
    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  );
15end entity;
```

Source file: misc/stat/stat_Histogram.vhdl

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    Valids         : 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  );
18end entity;
```

Source file: misc/stat/stat_Maximum.vhdl

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 (
```

```
    Clock          : in  std_logic;
    Reset          : in  std_logic;

    Enable         : in  std_logic;
    DataIn         : in  std_logic_vector(DATA_BITS - 1 downto 0);

    Valids         : out std_logic_vector(DEPTH - 1 downto 0);
    Minimums       : out T_SLM(DEPTH - 1 downto 0, DATA_BITS - 1 downto 0);
    Counts         : out T_SLM(DEPTH - 1 downto 0, COUNTER_BITS - 1 downto 0)
  );
end entity;
```

Source file: misc/stat/stat_Minimum.vhdl

sync

The namespace `PoC.misc.sync` offers different clock-domain-crossing (CDC) synchronizer circuits. All synchronizers are based on the basic 2 flip-flop synchonizer 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	sync_Bits	sync_Strobe	fifo_ic_got [5]	sync_Reset	sync_Pulse
n Bit	sync_Vector	sync_Command	fifo_ic_got [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` respectivily.

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` respectivily.

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` respectivily.

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.

[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 documented
[3] A *pulse* signal is a very short event.
[5] See the `PoC.fifo` namespace for cross-clock capable FIFOs.

- 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.

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 choosen.

> **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:

```
entity sync_Bits is
  generic (
    BITS           : positive            := 1;                  -- number of bit to be synchronized
    INIT           : std_logic_vector    := x"00000000";        -- initialitation bits
    SYNC_DEPTH     : T_MISC_SYNC_DEPTH   := 2                   -- generate SYNC_DEPTH many stages,
  );
  port (
    Clock          : in  std_logic;                             -- <Clock>  output clock domain
    Input          : in  std_logic_vector(BITS - 1 downto 0);   -- @async:  input bits
    Output         : out std_logic_vector(BITS - 1 downto 0)    -- @Clock:  output bits
  );
end entity;
```

Source file: misc/sync/sync_Bits.vhdl

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.

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:

```
entity sync_Command is
  generic (
    BITS                : positive          := 8;                       -- number of bit to be sync
    INIT                : std_logic_vector  := x"00000000"              --
  );
  port (
    Clock1              : in  std_logic;                                -- <Clock>  input clock
    Clock2              : in  std_logic;                                -- <Clock>  output clock
    Input               : in  std_logic_vector(BITS - 1 downto 0);      -- @Clock1: input vector
    Output              : out std_logic_vector(BITS - 1 downto 0);      -- @Clock2: output vector
    Busy                : out std_logic;                                -- @Clock1: busy bit
    Changed             : out std_logic                                 -- @Clock2: changed bit
  );
end entity;
```

Source file: misc/sync/sync_Command.vhdl

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 choosen.

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:

```
entity sync_Pulse is
  generic (
    BITS            : positive              := 1;                   -- number of bit to be synchronized
    SYNC_DEPTH      : T_MISC_SYNC_DEPTH     := 2                    -- generate SYNC_DEPTH many stages,
  );
  port (
    Clock           : in  std_logic;                                -- <Clock>  output clock domain
    Input           : in  std_logic_vector(BITS - 1 downto 0);      -- @async:  input bits
    Output          : out std_logic_vector(BITS - 1 downto 0)       -- @Clock:  output bits
  );
end entity;
```

Source file: misc/sync/sync_Pulse.vhdl

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.

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:

```
entity sync_Reset is
  generic (
    SYNC_DEPTH     : T_MISC_SYNC_DEPTH    := 2  -- generate SYNC_DEPTH many stages, at least 2
  );
  port (
    Clock          : in  std_logic;             -- <Clock>  output clock domain
    Input          : in  std_logic;             -- @async:  reset input
    Output         : out std_logic              -- @Clock:  reset output
  );
end entity;
```

Source file: misc/sync/sync_Reset.vhdl

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:

```
entity sync_Strobe is
  generic (
    BITS                : positive    := 1;                          -- number of bit to be sync
    GATED_INPUT_BY_BUSY : boolean     := TRUE                        -- use gated input (by busy
  );
  port (
    Clock1              : in  std_logic;                             -- <Clock>  input clock dom
```

```
    Clock2              : in  std_logic;                                    -- <Clock>  output clock do
    Input               : in  std_logic_vector(BITS - 1 downto 0);          -- @Clock1: input bits
    Output              : out std_logic_vector(BITS - 1 downto 0);          -- @Clock2: output bits
    Busy                : out std_logic_vector(BITS - 1 downto 0)           -- @Clock1: busy bits
  );
end entity;
```

Source file: misc/sync/sync_Strobe.vhdl

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 constrainted. Please attend to the notes of the instantiated sub modules.

Entity Declaration:

```
entity sync_Vector is
  generic (
    MASTER_BITS         : positive          := 8;                       -- number of bit to be sync
    SLAVE_BITS          : natural           := 0;
    INIT                : std_logic_vector  := x"00000000"              --
  );
  port (
    Clock1              : in  std_logic;                                                        -- <Clo
    Clock2              : in  std_logic;                                                        -- <Clo
    Input               : in  std_logic_vector((MASTER_BITS + SLAVE_BITS) - 1 downto 0);        -- @Clo
    Output              : out std_logic_vector((MASTER_BITS + SLAVE_BITS) - 1 downto 0);        -- @Clo
    Busy                : out std_logic;                                                        -- @Clo
    Changed             : out std_logic                                                         -- @Clo
  );
end entity;
```

Source file: misc/sync/sync_Vector.vhdl

Package

This package holds all component declarations for this namespace.

Source file: misc/misc.pkg.vhdl

misc_Delay

Todo

No documentation available.

Entity Declaration:

```
entity misc_Delay is
  generic (
    BITS          : positive;
    TAPS          : T_NATVEC          -- select one or multiple delay tap points
  );
  port (
    Clock         : in  std_logic;                                          -- clock
    Reset         : in  std_logic   := '0';                                 -- reset; avoid reset
    Enable        : in  std_logic   := '1';                                 -- enable
    DataIn        : in  std_logic_vector(BITS - 1 downto 0);                -- data to delay
    DataOut       : out T_SLM(TAPS'length - 1 downto 0, BITS - 1 downto 0)  -- delayed ouputs, ta
  );
end entity;
```

Source file: misc/misc_Delay.vhdl

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:

```
entity misc_FrequencyMeasurement is
  generic (
    REFERENCE_CLOCK_FREQ  : FREQ      := 100 MHz
  );
  port (
    Reference_Clock   : in  std_logic;
    Input_Clock       : in  std_logic;

    Start             : in  std_logic;
    Done              : out std_logic;
    Result            : out T_SLV_32
  );
end entity;
```

Source file: misc/misc_FrequencyMeasurement.vhdl

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

misc_Sequencer

Todo

No documentation available.

Entity Declaration:

Source file: misc/misc_Sequencer.vhdl

misc_StrobeGenerator

Todo

No documentation available.

Entity Declaration:

Source file: misc/misc_StrobeGenerator.vhdl

misc_StrobeLimiter

Todo

No documentation available.

Entity Declaration:

Source file: misc/misc_StrobeLimiter.vhdl

WordAligner

Todo

No documentation available.

Entity Declaration:

Source file: misc/misc_WordAligner.vhdl

2.4.12 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
- PoC.net.FrameLoopback

arp

These are ARP entities....

arp_BroadCast_Receiver

Todo

No documentation available.

Entity Declaration:

```
entity arp_BroadCast_Receiver is
  generic (
    ALLOWED_PROTOCOL_IPV4       : boolean                       := TRUE;
    ALLOWED_PROTOCOL_IPV6       : boolean                       := FALSE
  );
  port (
    Clock                       : in  std_logic;                                --
    Reset                       : in  std_logic;                                --

    RX_Valid                    : in  std_logic;
    RX_Data                     : in  T_SLV_8;
    RX_SOF                      : in  std_logic;
    RX_EOF                      : in  std_logic;
    RX_Ack                      : out std_logic;
    RX_Meta_rst                 : out std_logic;
    RX_Meta_SrcMACAddress_nxt   : out std_logic;
    RX_Meta_SrcMACAddress_Data  : in  T_SLV_8;
    RX_Meta_DestMACAddress_nxt  : out std_logic;
    RX_Meta_DestMACAddress_Data : in  T_SLV_8;

    Clear                       : in  std_logic;
    Error                       : out std_logic;

    RequestReceived             : out std_logic;
    Address_rst                 : in  std_logic;
    SenderMACAddress_nxt        : in  std_logic;
    SenderMACAddress_Data       : out T_SLV_8;
    SenderIPAddress_nxt         : in  std_logic;
    SenderIPAddress_Data        : out T_SLV_8;
    TargetIPAddress_nxt         : in  std_logic;
    TargetIPAddress_Data        : out T_SLV_8
  );
end entity;
```

Source file: net/arp/arp_BroadCast_Receiver.vhdl

arp_BroadCast_Requester

Todo

No documentation available.

Entity Declaration:

```
entity arp_BroadCast_Requester is
  generic (
    ALLOWED_PROTOCOL_IPV4         : boolean                       := TRUE;
    ALLOWED_PROTOCOL_IPV6         : boolean                       := FALSE
  );
  port (
    Clock                         : in  std_logic;                --
    Reset                         : in  std_logic;                --

    SendRequest                   : in  std_logic;
    Complete                      : out std_logic;

    Address_rst                   : out std_logic;
    SenderMACAddress_nxt          : out std_logic;
    SenderMACAddress_Data         : in  T_SLV_8;
    SenderIPv4Address_nxt         : out std_logic;
    SenderIPv4Address_Data        : in  T_SLV_8;
    TargetMACAddress_nxt          : out std_logic;
    TargetMACAddress_Data         : in  T_SLV_8;
    TargetIPv4Address_nxt         : out std_logic;
    TargetIPv4Address_Data        : in  T_SLV_8;

    TX_Valid                      : out std_logic;
    TX_Data                       : out T_SLV_8;
    TX_SOF                        : out std_logic;
    TX_EOF                        : out std_logic;
    TX_Ack                        : in  std_logic;
    TX_Meta_DestMACAddress_rst    : in  std_logic;
    TX_Meta_DestMACAddress_nxt    : in  std_logic;
    TX_Meta_DestMACAddress_Data   : out T_SLV_8
  );
end entity;
```

Source file: net/arp/arp_BroadCast_Requester.vhdl

arp_Cache

Todo

No documentation available.

Entity Declaration:

```
entity arp_Cache is
  generic (
    CLOCK_FREQ                : FREQ                                  := 125 MHz;
    REPLACEMENT_POLICY        : string                                := "LRU";
    TAG_BYTE_ORDER            : T_BYTE_ORDER                          := BIG_ENDIAN;
    DATA_BYTE_ORDER           : T_BYTE_ORDER                          := BIG_ENDIAN;
    INITIAL_CACHE_CONTENT     : T_NET_ARP_ARPCACHE_VECTOR
  );
  port (
    Clock                     : in  std_logic;                                    --
    Reset                     : in  std_logic;                                    --

    Command                   : in  T_NET_ARP_ARPCACHE_COMMAND;
    Status                    : out T_NET_ARP_ARPCACHE_STATUS;
    NewIPv4Address_rst        : out std_logic;
    NewIPv4Address_nxt        : out std_logic;
    NewIPv4Address_Data       : in  T_SLV_8;
    NewMACAddress_rst         : out std_logic;
    NewMACAddress_nxt         : out std_logic;
    NewMACAddress_Data        : in  T_SLV_8;

    Lookup                    : in  std_logic;
    IPv4Address_rst           : out std_logic;
    IPv4Address_nxt           : out std_logic;
    IPv4Address_Data          : in  T_SLV_8;

    CacheResult               : out T_CACHE_RESULT;
    MACAddress_rst            : in  std_logic;
    MACAddress_nxt            : in  std_logic;
    MACAddress_Data           : out T_SLV_8
  );
end entity;
```

Source file: net/arp/arp_Cache.vhdl

arp_IPPool

Todo

No documentation available.

Entity Declaration:

```
entity arp_IPPool is
  generic (
    IPPOOL_SIZE                 : positive;
    INITIAL_IPV4ADDRESSES       : T_NET_IPV4_ADDRESS_VECTOR     := (0 to 7 => C_NET_IPV4_ADDRESS
  );
  port (
    Clock                       : in  std_logic;                                --
    Reset                       : in  std_logic;                                --

--    Command                     : in  T_ETHERNET_ARP_IPPOOL_COMMAND;
--    IPv4Address                 : in  T_NET_IPV4_ADDRESS;
--    MACAddress                  : in  T_ETHERNET_MAC_ADDRESS;

```

```
    Lookup                          : in  std_logic;
    IPv4Address_rst                 : out std_logic;
    IPv4Address_nxt                 : out std_logic;
    IPv4Address_Data                : in  T_SLV_8;

    PoolResult                      : out T_CACHE_RESULT
  );
end entity;
```

Source file: net/arp/arp_IPPool.vhdl

arp_Tester

Todo

No documentation available.

Entity Declaration:

Source file: net/arp/arp_Tester.vhdl

arp_UniCast_Receiver

Todo

No documentation available.

Entity Declaration:

```
entity arp_UniCast_Receiver is
  generic (
    ALLOWED_PROTOCOL_IPV4       : boolean                       := TRUE;
    ALLOWED_PROTOCOL_IPV6       : boolean                       := FALSE
  );
  port (
    Clock                       : in  std_logic;                                --
    Reset                       : in  std_logic;                                --

    RX_Valid                    : in  std_logic;
    RX_Data                     : in  T_SLV_8;
    RX_SOF                      : in  std_logic;
    RX_EOF                      : in  std_logic;
    RX_Ack                      : out std_logic;
    RX_Meta_rst                 : out std_logic;
    RX_Meta_SrcMACAddress_nxt   : out std_logic;
    RX_Meta_SrcMACAddress_Data  : in  T_SLV_8;
    RX_Meta_DestMACAddress_nxt  : out std_logic;
    RX_Meta_DestMACAddress_Data : in  T_SLV_8;

    Clear                       : in  std_logic;
    Error                       : out std_logic;

    ResponseReceived            : out std_logic;
    Address_rst                 : in  std_logic;
```

```
    SenderMACAddress_nxt          : in  std_logic;
    SenderMACAddress_Data         : out T_SLV_8;
    SenderIPAddress_nxt           : in  std_logic;
    SenderIPAddress_Data          : out T_SLV_8;
    TargetIPAddress_nxt           : in  std_logic;
    TargetIPAddress_Data          : out T_SLV_8;
    TargetMACAddress_nxt          : in  std_logic;
    TargetMACAddress_Data         : out T_SLV_8
  );
end entity;
```

Source file: net/arp/arp_UniCast_Receiver.vhdl

arp_UniCast_Responder

Todo

No documentation available.

Entity Declaration:

```
entity arp_UniCast_Responder is
  generic (
    ALLOWED_PROTOCOL_IPV4         : boolean                       := TRUE;
    ALLOWED_PROTOCOL_IPV6         : boolean                       := FALSE
  );
  port (
    Clock                         : in  std_logic;                --
    Reset                         : in  std_logic;                --

    SendResponse                  : in  std_logic;
    Complete                      : out std_logic;

    Address_rst                   : out std_logic;
    SenderMACAddress_nxt          : out std_logic;
    SenderMACAddress_Data         : in  T_SLV_8;
    SenderIPv4Address_nxt         : out std_logic;
    SenderIPv4Address_Data        : in  T_SLV_8;
    TargetMACAddress_nxt          : out std_logic;
    TargetMACAddress_Data         : in  T_SLV_8;
    TargetIPv4Address_nxt         : out std_logic;
    TargetIPv4Address_Data        : in  T_SLV_8;

    TX_Valid                      : out std_logic;
    TX_Data                       : out T_SLV_8;
    TX_SOF                        : out std_logic;
    TX_EOF                        : out std_logic;
    TX_Ack                        : in  std_logic;
    TX_Meta_DestMACAddress_rst    : in  std_logic;
    TX_Meta_DestMACAddress_nxt    : in  std_logic;
    TX_Meta_DestMACAddress_Data   : out T_SLV_8
  );
end entity;
```

Source file: net/arp/arp_UniCast_Responder.vhdl

arp_Wrapper

Todo

No documentation available.

Entity Declaration:

```
entity arp_Wrapper is
  generic (
    CLOCK_FREQ                          : FREQ                                  := 125 MHz;
    INTERFACE_MACADDRESS                : T_NET_MAC_ADDRESS                     := C_NET_MAC_ADDR
    INITIAL_IPV4ADDRESSES               : T_NET_IPV4_ADDRESS_VECTOR             := (0 => C_NET_IP
    INITIAL_ARPCACHE_CONTENT            : T_NET_ARP_ARPCACHE_VECTOR             := (0 => (Tag => 
    APR_REQUEST_TIMEOUT                 : time                                  := 100 ms
  );
  port (
    Clock                               : in  std_logic;
    Reset                               : in  std_logic;

    IPPool_Announce                     : in  std_logic;
    IPPool_Announced                    : out std_logic;

    IPCache_Lookup                      : in  std_logic;
    IPCache_IPv4Address_rst             : out std_logic;
    IPCache_IPv4Address_nxt             : out std_logic;
    IPCache_IPv4Address_Data            : in  T_SLV_8;

    IPCache_Valid                       : out std_logic;
    IPCache_MACAddress_rst              : in  std_logic;
    IPCache_MACAddress_nxt              : in  std_logic;
    IPCache_MACAddress_Data             : out T_SLV_8;

    Eth_UC_TX_Valid                     : out std_logic;
    Eth_UC_TX_Data                      : out T_SLV_8;
    Eth_UC_TX_SOF                       : out std_logic;
    Eth_UC_TX_EOF                       : out std_logic;
    Eth_UC_TX_Ack                       : in  std_logic;
    Eth_UC_TX_Meta_rst                  : in  std_logic;
    Eth_UC_TX_Meta_DestMACAddress_nxt   : in  std_logic;
    Eth_UC_TX_Meta_DestMACAddress_Data  : out T_SLV_8;

    Eth_UC_RX_Valid                     : in  std_logic;
    Eth_UC_RX_Data                      : in  T_SLV_8;
    Eth_UC_RX_SOF                       : in  std_logic;
    Eth_UC_RX_EOF                       : in  std_logic;
    Eth_UC_RX_Ack                       : out std_logic;
    Eth_UC_RX_Meta_rst                  : out std_logic;
    Eth_UC_RX_Meta_SrcMACAddress_nxt    : out std_logic;
    Eth_UC_RX_Meta_SrcMACAddress_Data   : in  T_SLV_8;
    Eth_UC_RX_Meta_DestMACAddress_nxt   : out std_logic;
    Eth_UC_RX_Meta_DestMACAddress_Data  : in  T_SLV_8;

    Eth_BC_RX_Valid                     : in  std_logic;
    Eth_BC_RX_Data                      : in  T_SLV_8;
    Eth_BC_RX_SOF                       : in  std_logic;
    Eth_BC_RX_EOF                       : in  std_logic;
    Eth_BC_RX_Ack                       : out std_logic;
    Eth_BC_RX_Meta_rst                  : out std_logic;
    Eth_BC_RX_Meta_SrcMACAddress_nxt    : out std_logic;
    Eth_BC_RX_Meta_SrcMACAddress_Data   : in  T_SLV_8;
    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

eth

These are eth entities....

eth_GEMAC_GMII

Todo

No documentation available.

Entity Declaration:

Source file: net/eth/eth_GEMAC_GMII.vhdl

Eth_GEMAC_RX

Todo

No documentation available.

Entity Declaration:

Source file: net/eth/eth_GEMAC_RX.vhdl

Eth_GEMAC_TX

Todo

No documentation available.

Entity Declaration:

Source file: net/eth/eth_GEMAC_TX.vhdl

Eth_PHYController

Todo

No documentation available.

Entity Declaration:

Source file: net/eth/eth_PHYController.vhdl

Eth_PHYController_Marvell_88E1111

Todo

No documentation available.

Entity Declaration:

Source file: net/eth/eth_PHYController_Marvell_88E1111.vhdl

Eth_Wrapper

Todo

No documentation available.

Entity Declaration:

Source file: net/eth/eth_Wrapper.vhdl

icmpv4

These are icmpv4 entities....

icmpv4_RX

Todo

No documentation available.

Entity Declaration:

```
entity icmpv4_RX is
  generic (
    DEBUG                       : boolean                       := FALSE
  );
  port (
    Clock                       : in  std_logic;                                --
    Reset                       : in  std_logic;                                --
    -- CSE interface
    Command                     : in  T_NET_ICMPV4_RX_COMMAND;
    Status                      : out T_NET_ICMPV4_RX_STATUS;
    Error                       : out T_NET_ICMPV4_RX_ERROR;
    -- IN port
```

```
    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_SrcMACAddress_nxt       : out std_logic;
    In_Meta_SrcMACAddress_Data      : in  T_SLV_8;
    In_Meta_DestMACAddress_nxt      : out std_logic;
    In_Meta_DestMACAddress_Data     : in  T_SLV_8;
    In_Meta_SrcIPv4Address_nxt      : out std_logic;
    In_Meta_SrcIPv4Address_Data     : in  T_SLV_8;
    In_Meta_DestIPv4Address_nxt     : out std_logic;
    In_Meta_DestIPv4Address_Data    : in  T_SLV_8;
    In_Meta_Length                  : in  T_SLV_16;
    -- OUT Port
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_SrcMACAddress_nxt      : in  std_logic;
    Out_Meta_SrcMACAddress_Data     : out T_SLV_8;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8;
    Out_Meta_SrcIPv4Address_nxt     : in  std_logic;
    Out_Meta_SrcIPv4Address_Data    : out T_SLV_8;
    Out_Meta_DestIPv4Address_nxt    : in  std_logic;
    Out_Meta_DestIPv4Address_Data   : out T_SLV_8;
    Out_Meta_Length                 : out T_SLV_16;
    Out_Meta_Type                   : out T_SLV_8;
    Out_Meta_Code                   : out T_SLV_8;
    Out_Meta_Identification         : out T_SLV_16;
    Out_Meta_SequenceNumber         : out T_SLV_16;
    Out_Meta_Payload_nxt            : in  std_logic;
    Out_Meta_Payload_last           : out std_logic;
    Out_Meta_Payload_Data           : out T_SLV_8
  );
end entity;
```

Source file: net/icmpv4/icmpv4_RX.vhdl

icmpv4_TX

Todo

No documentation available.

Entity Declaration:

```
entity icmpv4_TX is
  generic (
    DEBUG                           : boolean                       := FALSE;
    SOURCE_IPV4ADDRESS              : T_NET_IPV4_ADDRESS            := C_NET_IPV4_ADDRESS_EMPTY
  );
  port (
    Clock                           : in  std_logic;                                    --
    Reset                           : in  std_logic;                                    --
    -- CSE interface
    Command                         : in  T_NET_ICMPV4_TX_COMMAND;
    Status                          : out T_NET_ICMPV4_TX_STATUS;
    Error                           : out T_NET_ICMPV4_TX_ERROR;
```

```
    -- OUT port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_SrcIPv4Address_nxt     : in  std_logic;
    Out_Meta_SrcIPv4Address_Data    : out T_SLV_8;
    Out_Meta_DestIPv4Address_nxt    : in  std_logic;
    Out_Meta_DestIPv4Address_Data   : out T_SLV_8;
    Out_Meta_Length                 : out T_SLV_16;
    -- IN port
    In_Meta_rst                     : out std_logic;
    In_Meta_IPv4Address_nxt         : out std_logic;
    In_Meta_IPv4Address_Data        : in  T_SLV_8;
    In_Meta_Type                    : in  T_SLV_8;
    In_Meta_Code                    : in  T_SLV_8;
    In_Meta_Identification          : in  T_SLV_16;
    In_Meta_SequenceNumber          : in  T_SLV_16;
    In_Meta_Payload_nxt             : out std_logic;
    In_Meta_Payload_last            : in  std_logic;
    In_Meta_Payload_Data            : in  T_SLV_8
  );
end entity;
```

Source file: net/icmpv4/icmpv4_TX.vhdl

icmpv4_Wrapper

Todo

No documentation available.

Entity Declaration:

```
entity icmpv4_Wrapper is
  generic (
    DEBUG                           : boolean               := FALSE;
    SOURCE_IPV4ADDRESS              : T_NET_IPV4_ADDRESS    := C_NET_IPV4_ADDRESS_EMPTY
  );
  port (
    Clock                           : in  std_logic;
    Reset                           : in  std_logic;
    -- CSE interface
    Command                         : in  T_NET_ICMPV4_COMMAND;
    Status                          : out T_NET_ICMPV4_STATUS;
    Error                           : out T_NET_ICMPV4_ERROR;
    -- Echo-Request destination address
    IPv4Address_rst                 : out std_logic;
    IPv4Address_nxt                 : out std_logic;
    IPv4Address_Data                : in  T_SLV_8;
    -- to IPv4 layer
    IP_TX_Valid                     : out std_logic;
    IP_TX_Data                      : out T_SLV_8;
    IP_TX_SOF                       : out std_logic;
    IP_TX_EOF                       : out std_logic;
    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

icmpv6

These are icmpv6 entities....

icmpv6_RX

Todo

No documentation available.

Entity Declaration:

Source file: net/icmpv6/icmpv6_RX.vhdl

icmpv6_TX

Todo

No documentation available.

Entity Declaration:

Source file: net/icmpv6/icmpv6_TX.vhdl

icmpv6_Wrapper

Todo

No documentation available.

Entity Declaration:

Source file: net/icmpv6/icmpv6_Wrapper.vhdl

ipv4

These are ipv4 entities....

ipv4_RX

Todo

No documentation available.

Entity Declaration:

```
entity ipv4_RX is
  generic (
    DEBUG                           : boolean             := FALSE
  );
  port (
    Clock                           : in  std_logic;                  --
    Reset                           : in  std_logic;                  --
    -- STATUS port
    Error                           : out std_logic;
    -- IN port
    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_SrcMACAddress_nxt       : out std_logic;
    In_Meta_SrcMACAddress_Data      : in  T_SLV_8;
    In_Meta_DestMACAddress_nxt      : out std_logic;
    In_Meta_DestMACAddress_Data     : in  T_SLV_8;
    In_Meta_EthType                 : in  T_SLV_16;
    -- OUT port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
```

```
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_SrcMACAddress_nxt      : in  std_logic;
    Out_Meta_SrcMACAddress_Data     : out T_SLV_8;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8;
    Out_Meta_EthType                : out T_SLV_16;
    Out_Meta_SrcIPv4Address_nxt     : in  std_logic;
    Out_Meta_SrcIPv4Address_Data    : out T_SLV_8;
    Out_Meta_DestIPv4Address_nxt    : in  std_logic;
    Out_Meta_DestIPv4Address_Data   : out T_SLV_8;
    Out_Meta_Length                 : out T_SLV_16;
    Out_Meta_Protocol               : out T_SLV_8
  );
end entity;
```

Source file: net/ipv4/ipv4_RX.vhdl

ipv4_TX

Todo

No documentation available.

Entity Declaration:

```
entity ipv4_TX is
  generic (
    DEBUG                           : boolean             := FALSE
  );
  port (
    Clock                           : in  std_logic;                  --
    Reset                           : in  std_logic;                  --
    -- IN port
    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_SrcIPv4Address_nxt      : out std_logic;
    In_Meta_SrcIPv4Address_Data     : in  T_SLV_8;
    In_Meta_DestIPv4Address_nxt     : out std_logic;
    In_Meta_DestIPv4Address_Data    : in  T_SLV_8;
    In_Meta_Length                  : in  T_SLV_16;
    In_Meta_Protocol                : in  T_SLV_8;
    -- ARP port
    ARP_IPCache_Query               : out std_logic;
    ARP_IPCache_IPv4Address_rst     : in  std_logic;
    ARP_IPCache_IPv4Address_nxt     : in  std_logic;
    ARP_IPCache_IPv4Address_Data    : out T_SLV_8;
    ARP_IPCache_Valid               : in  std_logic;
    ARP_IPCache_MACAddress_rst      : out std_logic;
    ARP_IPCache_MACAddress_nxt      : out std_logic;
    ARP_IPCache_MACAddress_Data     : in  T_SLV_8;
    -- OUT port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
```

```
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8
  );
end entity;
```

Source file: net/ipv4/ipv4_TX.vhdl

ipv4_FrameLoopback

Todo

No documentation available.

Entity Declaration:

```
entity ipv4_FrameLoopback is
  generic (
    MAX_FRAMES                    : positive            := 4
  );
  port (
    Clock                         : in  std_logic;
    Reset                         : in  std_logic;
    -- IN port
    In_Valid                      : in  std_logic;
    In_Data                       : in  T_SLV_8;
    In_SOF                        : in  std_logic;
    In_EOF                        : in  std_logic;
    In_Ack                        : out std_logic;
    In_Meta_rst                   : out std_logic;
    In_Meta_SrcIPv4Address_nxt    : out std_logic;
    In_Meta_SrcIPv4Address_Data   : in  T_SLV_8;
    In_Meta_DestIPv4Address_nxt   : out std_logic;
    In_Meta_DestIPv4Address_Data  : in  T_SLV_8;
    In_Meta_Length                : in  T_SLV_16;
    -- OUT port
    Out_Valid                     : out std_logic;
    Out_Data                      : out T_SLV_8;
    Out_SOF                       : out std_logic;
    Out_EOF                       : out std_logic;
    Out_Ack                       : in  std_logic;
    Out_Meta_rst                  : in  std_logic;
    Out_Meta_SrcIPv4Address_nxt   : in  std_logic;
    Out_Meta_SrcIPv4Address_Data  : out T_SLV_8;
    Out_Meta_DestIPv4Address_nxt  : in  std_logic;
    Out_Meta_DestIPv4Address_Data : out T_SLV_8;
    Out_Meta_Length               : out T_SLV_16
  );
end entity;
```

Source file: net/ipv4/ipv4_FrameLoopback.vhdl

ipv4_Wrapper

Todo

No documentation available.

Entity Declaration:

```
entity ipv4_Wrapper is
  generic (
    DEBUG                           : boolean                           := FALSE;
    PACKET_TYPES                    : T_NET_IPV4_PROTOCOL_VECTOR        := (0 => x"00")
  );
  port (
    Clock                           : in  std_logic;
    Reset                           : in  std_logic;
    -- to MAC layer
    MAC_TX_Valid                    : out std_logic;
    MAC_TX_Data                     : out T_SLV_8;
    MAC_TX_SOF                      : out std_logic;
    MAC_TX_EOF                      : out std_logic;
    MAC_TX_Ack                      : in  std_logic;
    MAC_TX_Meta_rst                 : in  std_logic;
    MAC_TX_Meta_DestMACAddress_nxt  : in  std_logic;
    MAC_TX_Meta_DestMACAddress_Data : out T_SLV_8;
    -- from MAC layer
    MAC_RX_Valid                    : in  std_logic;
    MAC_RX_Data                     : in  T_SLV_8;
    MAC_RX_SOF                      : in  std_logic;
    MAC_RX_EOF                      : in  std_logic;
    MAC_RX_Ack                      : out std_logic;
    MAC_RX_Meta_rst                 : out std_logic;
    MAC_RX_Meta_SrcMACAddress_nxt   : out std_logic;
    MAC_RX_Meta_SrcMACAddress_Data  : in  T_SLV_8;
    MAC_RX_Meta_DestMACAddress_nxt  : out std_logic;
    MAC_RX_Meta_DestMACAddress_Data : in  T_SLV_8;
    MAC_RX_Meta_EthType             : in  T_SLV_16;
    -- to ARP
    ARP_IPCache_Query               : out std_logic;
    ARP_IPCache_IPv4Address_rst     : in  std_logic;
    ARP_IPCache_IPv4Address_nxt     : in  std_logic;
    ARP_IPCache_IPv4Address_Data    : out T_SLV_8;
    -- from ARP
    ARP_IPCache_Valid               : in  std_logic;
    ARP_IPCache_MACAddress_rst      : out std_logic;
    ARP_IPCache_MACAddress_nxt      : out std_logic;
    ARP_IPCache_MACAddress_Data     : in  T_SLV_8;
    -- from upper layer
    TX_Valid                        : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Data                         : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    TX_SOF                          : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_EOF                          : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Ack                          : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_rst                     : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_SrcIPv4Address_nxt      : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_SrcIPv4Address_Data     : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_DestIPv4Address_nxt     : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_DestIPv4Address_Data    : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_Length                  : in  T_SLVV_16(PACKET_TYPES'length - 1 downto 0);
    -- to upper layer
    RX_Valid                        : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Data                         : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
```

```
    RX_SOF                          : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_EOF                          : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Ack                          : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_rst                     : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcMACAddress_nxt       : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcMACAddress_Data      : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestMACAddress_nxt      : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestMACAddress_Data     : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_EthType                 : out T_SLVV_16(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcIPv4Address_nxt      : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcIPv4Address_Data     : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestIPv4Address_nxt     : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestIPv4Address_Data    : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_Length                  : out T_SLVV_16(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_Protocol                : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0)
  );
end entity;
```

Source file: net/ipv4/ipv4_Wrapper.vhdl

ipv6

These are ipv6 entities....

ipv6_RX

Todo

No documentation available.

Entity Declaration:

```
entity ipv6_RX is
  generic (
    DEBUG                           : boolean             := FALSE
  );
  port (
    Clock                           : in  std_logic;                  --
    Reset                           : in  std_logic;                  --
    -- STATUS port
    Error                           : out std_logic;
    -- IN port
    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_SrcMACAddress_nxt       : out std_logic;
    In_Meta_SrcMACAddress_Data      : in  T_SLV_8;
    In_Meta_DestMACAddress_nxt      : out std_logic;
    In_Meta_DestMACAddress_Data     : in  T_SLV_8;
    In_Meta_EthType                 : in  T_SLV_16;
    -- OUT port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
```

```
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_SrcMACAddress_nxt      : in  std_logic;
    Out_Meta_SrcMACAddress_Data     : out T_SLV_8;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8;
    Out_Meta_EthType                : out T_SLV_16;
    Out_Meta_SrcIPv6Address_nxt     : in  std_logic;
    Out_Meta_SrcIPv6Address_Data    : out T_SLV_8;
    Out_Meta_DestIPv6Address_nxt    : in  std_logic;
    Out_Meta_DestIPv6Address_Data   : out T_SLV_8;
    Out_Meta_TrafficClass           : out T_SLV_8;
    Out_Meta_FlowLabel              : out T_SLV_24; --STD_LOGIC_VECTOR(19 downto 0);
    Out_Meta_Length                 : out T_SLV_16;
    Out_Meta_NextHeader             : out T_SLV_8
  );
end entity;
```

Source file: net/ipv6/ipv6_RX.vhdl

ipv6_TX

Todo

No documentation available.

Entity Declaration:

```
entity ipv6_TX is
  generic (
    DEBUG                           : boolean             := FALSE
  );
  port (
    Clock                           : in  std_logic;                --
    Reset                           : in  std_logic;                --
    -- IN port
    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_SrcIPv6Address_nxt      : out std_logic;
    In_Meta_SrcIPv6Address_Data     : in  T_SLV_8;
    In_Meta_DestIPv6Address_nxt     : out std_logic;
    In_Meta_DestIPv6Address_Data    : in  T_SLV_8;
    In_Meta_TrafficClass            : in  T_SLV_8;
    In_Meta_FlowLabel               : in  T_SLV_24; --STD_LOGIC_VECTOR(19 downto 0);
    In_Meta_Length                  : in  T_SLV_16;
    In_Meta_NextHeader              : in  T_SLV_8;
    -- to NDP layer
    NDP_NextHop_Query               : out std_logic;
    NDP_NextHop_IPv6Address_rst     : in  std_logic;
    NDP_NextHop_IPv6Address_nxt     : in  std_logic;
    NDP_NextHop_IPv6Address_Data    : out T_SLV_8;
    -- from NDP layer
    NDP_NextHop_Valid               : in  std_logic;
```

```
    NDP_NextHop_MACAddress_rst      : out std_logic;
    NDP_NextHop_MACAddress_nxt      : out std_logic;
    NDP_NextHop_MACAddress_Data     : in  T_SLV_8;
    -- OUT port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8
  );
end entity;
```

Source file: net/ipv6/ipv6_TX.vhdl

ipv6_FrameLoopback

Todo

No documentation available.

Entity Declaration:

```
entity ipv6_FrameLoopback is
  generic (
    MAX_FRAMES                    : positive            := 4
  );
  port (
    Clock                         : in  std_logic;
    Reset                         : in  std_logic;
    -- IN port
    In_Valid                      : in  std_logic;
    In_Data                       : in  T_SLV_8;
    In_SOF                        : in  std_logic;
    In_EOF                        : in  std_logic;
    In_Ack                        : out std_logic;
    In_Meta_rst                   : out std_logic;
    In_Meta_SrcIPv6Address_nxt    : out std_logic;
    In_Meta_SrcIPv6Address_Data   : in  T_SLV_8;
    In_Meta_DestIPv6Address_nxt   : out std_logic;
    In_Meta_DestIPv6Address_Data  : in  T_SLV_8;
    In_Meta_Length                : in  T_SLV_16;
    -- OUT port
    Out_Valid                     : out std_logic;
    Out_Data                      : out T_SLV_8;
    Out_SOF                       : out std_logic;
    Out_EOF                       : out std_logic;
    Out_Ack                       : in  std_logic;
    Out_Meta_rst                  : in  std_logic;
    Out_Meta_SrcIPv6Address_nxt   : in  std_logic;
    Out_Meta_SrcIPv6Address_Data  : out T_SLV_8;
    Out_Meta_DestIPv6Address_nxt  : in  std_logic;
    Out_Meta_DestIPv6Address_Data : out T_SLV_8;
    Out_Meta_Length               : out T_SLV_16
  );
end entity;
```

Source file: net/ipv6/ipv6_FrameLoopback.vhdl

ipv6_Wrapper

Todo

No documentation available.

Entity Declaration:

```
entity ipv6_Wrapper is
  generic (
    DEBUG                           : boolean                           := FALSE;
    PACKET_TYPES                    : T_NET_IPV6_NEXT_HEADER_VECTOR     := (0 => x"00")
  );
  port (
    Clock                           : in  std_logic;
    Reset                           : in  std_logic;
    -- to MAC layer
    MAC_TX_Valid                    : out std_logic;
    MAC_TX_Data                     : out T_SLV_8;
    MAC_TX_SOF                      : out std_logic;
    MAC_TX_EOF                      : out std_logic;
    MAC_TX_Ack                      : in  std_logic;
    MAC_TX_Meta_rst                 : in  std_logic;
    MAC_TX_Meta_DestMACAddress_nxt  : in  std_logic;
    MAC_TX_Meta_DestMACAddress_Data : out T_SLV_8;
    -- from MAC layer
    MAC_RX_Valid                    : in  std_logic;
    MAC_RX_Data                     : in  T_SLV_8;
    MAC_RX_SOF                      : in  std_logic;
    MAC_RX_EOF                      : in  std_logic;
    MAC_RX_Ack                      : out std_logic;
    MAC_RX_Meta_rst                 : out std_logic;
    MAC_RX_Meta_SrcMACAddress_nxt   : out std_logic;
    MAC_RX_Meta_SrcMACAddress_Data  : in  T_SLV_8;
    MAC_RX_Meta_DestMACAddress_nxt  : out std_logic;
    MAC_RX_Meta_DestMACAddress_Data : in  T_SLV_8;
    MAC_RX_Meta_EthType             : in  T_SLV_16;
    -- to NDP layer
    NDP_NextHop_Query               : out std_logic;
    NDP_NextHop_IPv6Address_rst     : in  std_logic;
    NDP_NextHop_IPv6Address_nxt     : in  std_logic;
    NDP_NextHop_IPv6Address_Data    : out T_SLV_8;
    -- from NDP layer
    NDP_NextHop_Valid               : in  std_logic;
    NDP_NextHop_MACAddress_rst      : out std_logic;
    NDP_NextHop_MACAddress_nxt      : out std_logic;
    NDP_NextHop_MACAddress_Data     : in  T_SLV_8;
    -- from upper layer
    TX_Valid                        : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Data                         : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    TX_SOF                          : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_EOF                          : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Ack                          : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_rst                     : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_SrcIPv6Address_nxt      : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_SrcIPv6Address_Data     : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
```

```
    TX_Meta_DestIPv6Address_nxt      : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_DestIPv6Address_Data     : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_TrafficClass             : in  T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_FlowLabel                : in  T_SLVV_24(PACKET_TYPES'length - 1 downto 0);
    TX_Meta_Length                   : in  T_SLVV_16(PACKET_TYPES'length - 1 downto 0);
    -- to upper layer
    RX_Valid                         : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Data                          : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_SOF                           : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_EOF                           : out std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Ack                           : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_rst                      : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcMACAddress_nxt        : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcMACAddress_Data       : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestMACAddress_nxt       : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestMACAddress_Data      : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_EthType                  : out T_SLVV_16(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcIPv6Address_nxt       : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_SrcIPv6Address_Data      : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestIPv6Address_nxt      : in  std_logic_vector(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_DestIPv6Address_Data     : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_TrafficClass             : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_FlowLabel                : out T_SLVV_24(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_Length                   : out T_SLVV_16(PACKET_TYPES'length - 1 downto 0);
    RX_Meta_NextHeader               : out T_SLVV_8(PACKET_TYPES'length - 1 downto 0)
  );
end entity;
```

Source file: net/ipv6/ipv6_Wrapper.vhdl

mac

These are mac entities....

mac_RX_DestMAC_Switch

Todo

No documentation available.

Entity Declaration:

```
entity mac_RX_DestMAC_Switch is
  generic (
    DEBUG                           : boolean                           := FALSE;
    MAC_ADDRESSES                   : T_NET_MAC_ADDRESS_VECTOR    := (0 => C_NET_MAC_ADDRESS_EMPTY);
    MAC_ADDRESSE_MASKS              : T_NET_MAC_ADDRESS_VECTOR    := (0 => C_NET_MAC_MASK_DEFAULT)
  );
  port (
    Clock                           : in  std_logic;
    Reset                           : in  std_logic;

    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
```

```
16
17      Out_Valid                       : out std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
18      Out_Data                        : out T_SLVV_8(MAC_ADDRESSES'length - 1 downto 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

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_EM
5       MAC_ADDRESSE_MASKS            : T_NET_MAC_ADDRESS_VECTOR        := (0 => C_NET_MAC_MASK_DEFAU
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;
29      Out_Meta_SrcMACAddress_Data   : out T_SLV_8
30    );
31  end entity;
```

Source file: net/mac/mac_RX_SrcMAC_Filter.vhdl

mac_RX_Type_Switch

Todo

No documentation available.

Entity Declaration:

```
entity mac_RX_Type_Switch is
  generic (
    DEBUG                         : boolean                           := FALSE;
    ETHERNET_TYPES                : T_NET_MAC_ETHERNETTYPE_VECTOR     := (0 => C_NET_MAC_ETHERNETTYI
  );
  port (
    Clock                         : in  std_logic;
    Reset                         : in  std_logic;

    In_Valid                      : in  std_logic;
    In_Data                       : in  T_SLV_8;
    In_SOF                        : in  std_logic;
    In_EOF                        : in  std_logic;
    In_Ack                        : out std_logic;
    In_Meta_rst                   : out std_logic;
    In_Meta_SrcMACAddress_nxt     : out std_logic;
    In_Meta_SrcMACAddress_Data    : in  T_SLV_8;
    In_Meta_DestMACAddress_nxt    : out std_logic;
    In_Meta_DestMACAddress_Data   : in  T_SLV_8;

    Out_Valid                     : out std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_Data                      : out T_SLVV_8(ETHERNET_TYPES'length - 1 downto 0);
    Out_SOF                       : out std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_EOF                       : out std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_Ack                       : in  std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_Meta_rst                  : in  std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_Meta_SrcMACAddress_nxt    : in  std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_Meta_SrcMACAddress_Data   : out T_SLVV_8(ETHERNET_TYPES'length - 1 downto 0);
    Out_Meta_DestMACAddress_nxt   : in  std_logic_vector(ETHERNET_TYPES'length - 1 downto 0);
    Out_Meta_DestMACAddress_Data  : out T_SLVV_8(ETHERNET_TYPES'length - 1 downto 0);
    Out_Meta_EthType              : out T_NET_MAC_ETHERNETTYPE_VECTOR(ETHERNET_TYPES'length - 1 d
  );
end entity;
```

Source file: net/mac/mac_RX_Type_Switch.vhdl

mac_TX_SrcMAC_Prepender

Todo

No documentation available.

Entity Declaration:

```
entity mac_TX_SrcMAC_Prepender is
  generic (
    DEBUG                         : boolean                           := FALSE;
    MAC_ADDRESSES                 : T_NET_MAC_ADDRESS_VECTOR          := (0 => C_NET_MAC_ADDRESS_EMI
  );
```

```
  port (
    Clock                           : in  std_logic;
    Reset                           : in  std_logic;
    -- IN Port
    In_Valid                        : in  std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
    In_Data                         : in  T_SLVV_8(MAC_ADDRESSES'length - 1 downto 0);
    In_SOF                          : in  std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
    In_EOF                          : in  std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
    In_Ack                          : out std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
    In_Meta_rst                     : out std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
    In_Meta_DestMACAddress_nxt      : out std_logic_vector(MAC_ADDRESSES'length - 1 downto 0);
    In_Meta_DestMACAddress_Data     : in  T_SLVV_8(MAC_ADDRESSES'length - 1 downto 0);
    -- OUT Port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8
  );
end entity;
```

Source file: net/mac/mac_TX_SrcMAC_Prepender.vhdl

mac_TX_DestMAC_Prepender

Todo

No documentation available.

Entity Declaration:

```
entity mac_TX_DestMAC_Prepender is
  generic (
    DEBUG                           : boolean                       := FALSE
  );
  port (
    Clock                           : in  std_logic;
    Reset                           : in  std_logic;

    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_DestMACAddress_nxt      : out std_logic;
    In_Meta_DestMACAddress_Data     : in  T_SLV_8;

    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic
  );
end entity;
```

Source file: net/mac/mac_TX_DestMAC_Prepender.vhdl

mac_TX_Type_Prepender

Todo

No documentation available.

Entity Declaration:

Source file: net/mac/mac_TX_Type_Prepender.vhdl

mac_FrameLoopback

Todo

No documentation available.

Entity Declaration:

```
entity mac_FrameLoopback is
  generic (
    MAX_FRAMES                    : positive            := 4
  );
  port (
    Clock                         : in  std_logic;
    Reset                         : in  std_logic;
    -- IN Port
    In_Valid                      : in  std_logic;
    In_Data                       : in  T_SLV_8;
    In_SOF                        : in  std_logic;
    In_EOF                        : in  std_logic;
    In_Ack                        : out std_logic;
    In_Meta_rst                   : out std_logic;
    In_Meta_SrcMACAddress_nxt     : out std_logic;
    In_Meta_SrcMACAddress_Data    : in  T_SLV_8;
    In_Meta_DestMACAddress_nxt    : out std_logic;
    In_Meta_DestMACAddress_Data   : in  T_SLV_8;
    -- OUT Port
    Out_Valid                     : out std_logic;
    Out_Data                      : out T_SLV_8;
    Out_SOF                       : out std_logic;
    Out_EOF                       : out std_logic;
    Out_Ack                       : in  std_logic;
    Out_Meta_rst                  : in  std_logic;
    Out_Meta_SrcMACAddress_nxt    : in  std_logic;
    Out_Meta_SrcMACAddress_Data   : out T_SLV_8;
    Out_Meta_DestMACAddress_nxt   : in  std_logic;
    Out_Meta_DestMACAddress_Data  : out T_SLV_8
  );
end entity;
```

Source file: net/mac/mac_FrameLoopback.vhdl

mac_Wrapper

Todo

No documentation available.

Entity Declaration:

```
entity mac_Wrapper is
  generic (
    DEBUG                         : boolean                                 := FALSE;
    MAC_CONFIG                    : T_NET_MAC_CONFIGURATION_VECTOR
  );
  port (
    Clock                         : in  std_logic;
    Reset                         : in  std_logic;

    Eth_TX_Valid                  : out std_logic;
    Eth_TX_Data                   : out T_SLV_8;
    Eth_TX_SOF                    : out std_logic;
    Eth_TX_EOF                    : out std_logic;
    Eth_TX_Ack                    : in  std_logic;

    Eth_RX_Valid                  : in  std_logic;
    Eth_RX_Data                   : in  T_SLV_8;
    Eth_RX_SOF                    : in  std_logic;
    Eth_RX_EOF                    : in  std_logic;
    Eth_RX_Ack                    : out std_logic;

    TX_Valid                      : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_Data                       : in  T_SLVV_8(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_SOF                        : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_EOF                        : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_Ack                        : out std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_Meta_rst                   : out std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_Meta_DestMACAddress_nxt    : out std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    TX_Meta_DestMACAddress_Data   : in  T_SLVV_8(getPortCount(MAC_CONFIG) - 1 downto 0);

    RX_Valid                      : out std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Data                       : out T_SLVV_8(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_SOF                        : out std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_EOF                        : out std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Ack                        : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Meta_rst                   : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Meta_SrcMACAddress_nxt     : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Meta_SrcMACAddress_Data    : out T_SLVV_8(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Meta_DestMACAddress_nxt    : in  std_logic_vector(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Meta_DestMACAddress_Data   : out T_SLVV_8(getPortCount(MAC_CONFIG) - 1 downto 0);
    RX_Meta_EthType               : out T_NET_MAC_ETHERNETTYPE_VECTOR(getPortCount(MAC_CONFIG) - 1 d
  );
end entity;
```

Source file: net/mac/mac_Wrapper.vhdl

ndp

These are ndp entities....

ndp_DestinationCache

Todo

No documentation available.

Entity Declaration:

Source file: net/ndp/ndp_DestinationCache.vhdl

ndp_FSMQuery

Todo

No documentation available.

Entity Declaration:

Source file: net/ndp/ndp_FSMQuery.vhdl

ndp_NeighborCache

Todo

No documentation available.

Entity Declaration:

Source file: net/ndp/ndp_NeighborCache.vhdl

NDP_Wrapper

Todo

No documentation available.

Entity Declaration:

Source file: net/ndp/ndp_Wrapper.vhdl

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

stack_UDPv4

Todo

No documentation available.

Entity Declaration:

Source file: net/stack/stack_UDPv4.vhdl

stack_UDPv6

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

udp

These are udp entities....

udp_RX

Todo

No documentation available.

Entity Declaration:

```
entity udp_RX is
  generic (
    DEBUG                           : boolean           := FALSE;
    IP_VERSION                      : positive          := 6
  );
  port (
    Clock                           : in  std_logic;                --
    Reset                           : in  std_logic;                --
    -- STATUS port
    Error                           : out std_logic;
    -- IN port
    In_Valid                        : in  std_logic;
    In_Data                         : in  T_SLV_8;
    In_SOF                          : in  std_logic;
    In_EOF                          : in  std_logic;
    In_Ack                          : out std_logic;
    In_Meta_rst                     : out std_logic;
    In_Meta_SrcMACAddress_nxt       : out std_logic;
    In_Meta_SrcMACAddress_Data      : in  T_SLV_8;
    In_Meta_DestMACAddress_nxt      : out std_logic;
    In_Meta_DestMACAddress_Data     : in  T_SLV_8;
    In_Meta_EthType                 : in  T_SLV_16;
    In_Meta_SrcIPAddress_nxt        : out std_logic;
    In_Meta_SrcIPAddress_Data       : in  T_SLV_8;
    In_Meta_DestIPAddress_nxt       : out std_logic;
    In_Meta_DestIPAddress_Data      : in  T_SLV_8;
--    In_Meta_TrafficClass            : in  T_SLV_8;
--    In_Meta_FlowLabel               : in  T_SLV_24;
    In_Meta_Length                  : in  T_SLV_16;
    In_Meta_Protocol                : in  T_SLV_8;
    -- OUT port
    Out_Valid                       : out std_logic;
    Out_Data                        : out T_SLV_8;
    Out_SOF                         : out std_logic;
    Out_EOF                         : out std_logic;
    Out_Ack                         : in  std_logic;
    Out_Meta_rst                    : in  std_logic;
    Out_Meta_SrcMACAddress_nxt      : in  std_logic;
    Out_Meta_SrcMACAddress_Data     : out T_SLV_8;
    Out_Meta_DestMACAddress_nxt     : in  std_logic;
    Out_Meta_DestMACAddress_Data    : out T_SLV_8;
    Out_Meta_EthType                : out T_SLV_16;
    Out_Meta_SrcIPAddress_nxt       : in  std_logic;
    Out_Meta_SrcIPAddress_Data      : out T_SLV_8;
    Out_Meta_DestIPAddress_nxt      : in  std_logic;
    Out_Meta_DestIPAddress_Data     : out T_SLV_8;
--    Out_Meta_TrafficClass           : out T_SLV_8;
--    Out_Meta_FlowLabel              : out T_SLV_24;
    Out_Meta_Length                 : out T_SLV_16;
    Out_Meta_Protocol               : out T_SLV_8;
    Out_Meta_SrcPort                : out T_SLV_16;
```

```
    Out_Meta_DestPort               : out T_SLV_16
  );
end entity;
```

Source file: net/udp/udp_RX.vhdl

udp_TX

Todo

No documentation available.

Entity Declaration:

```
entity udp_TX is
  generic (
    DEBUG                       : boolean           := FALSE;
    IP_VERSION                  : positive          := 6
  );
  port (
    Clock                       : in  std_logic;                  --
    Reset                       : in  std_logic;                  --
    -- IN port
    In_Valid                    : in  std_logic;
    In_Data                     : in  T_SLV_8;
    In_SOF                      : in  std_logic;
    In_EOF                      : in  std_logic;
    In_Ack                      : out std_logic;
    In_Meta_rst                 : out std_logic;
    In_Meta_SrcIPAddress_nxt    : out std_logic;
    In_Meta_SrcIPAddress_Data   : in  T_SLV_8;
    In_Meta_DestIPAddress_nxt   : out std_logic;
    In_Meta_DestIPAddress_Data  : in  T_SLV_8;
    In_Meta_SrcPort             : in  T_SLV_16;
    In_Meta_DestPort            : in  T_SLV_16;
    In_Meta_Length              : in  T_SLV_16;
    In_Meta_Checksum            : in  T_SLV_16;
    -- OUT port
    Out_Valid                   : out std_logic;
    Out_Data                    : out T_SLV_8;
    Out_SOF                     : out std_logic;
    Out_EOF                     : out std_logic;
    Out_Ack                     : in  std_logic;
    Out_Meta_rst                : in  std_logic;
    Out_Meta_SrcIPAddress_nxt   : in  std_logic;
    Out_Meta_SrcIPAddress_Data  : out T_SLV_8;
    Out_Meta_DestIPAddress_nxt  : in  std_logic;
    Out_Meta_DestIPAddress_Data : out T_SLV_8;
    Out_Meta_Length             : out T_SLV_16
  );
end entity;
```

Source file: net/udp/udp_TX.vhdl

udp_FrameLoopback

Todo

No documentation available.

Entity Declaration:

```
entity udp_FrameLoopback is
  generic (
    IP_VERSION                  : positive            := 6;
    MAX_FRAMES                  : positive            := 4
  );
  port (
    Clock                       : in  std_logic;
    Reset                       : in  std_logic;
    -- IN port
    In_Valid                    : in  std_logic;
    In_Data                     : in  T_SLV_8;
    In_SOF                      : in  std_logic;
    In_EOF                      : in  std_logic;
    In_Ack                      : out std_logic;
    In_Meta_rst                 : out std_logic;
    In_Meta_DestIPAddress_nxt   : out std_logic;
    In_Meta_DestIPAddress_Data  : in  T_SLV_8;
    In_Meta_SrcIPAddress_nxt    : out std_logic;
    In_Meta_SrcIPAddress_Data   : in  T_SLV_8;
    In_Meta_DestPort            : in  T_NET_UDP_PORT;
    In_Meta_SrcPort             : in  T_NET_UDP_PORT;
    -- OUT port
    Out_Valid                   : out std_logic;
    Out_Data                    : out T_SLV_8;
    Out_SOF                     : out std_logic;
    Out_EOF                     : out std_logic;
    Out_Ack                     : in  std_logic;
    Out_Meta_rst                : in  std_logic;
    Out_Meta_DestIPAddress_nxt  : in  std_logic;
    Out_Meta_DestIPAddress_Data : out T_SLV_8;
    Out_Meta_SrcIPAddress_nxt   : in  std_logic;
    Out_Meta_SrcIPAddress_Data  : out T_SLV_8;
    Out_Meta_DestPort           : out T_NET_UDP_PORT;
    Out_Meta_SrcPort            : out T_NET_UDP_PORT
  );
end entity;
```

Source file: net/udp/udp_FrameLoopback.vhdl

udp_Wrapper

Todo

No documentation available.

Entity Declaration:

```
entity udp_Wrapper is
  generic (
```

```
    DEBUG                                : boolean                        := FALSE;
    IP_VERSION                           : positive                       := 6;
    PORTPAIRS                            : T_NET_UDP_PORTPAIR_VECTOR      := (0 => (x"0000", x"0000"))
  );
  port (
    Clock                                : in  std_logic;
    Reset                                : in  std_logic;
    -- from IP layer
    IP_TX_Valid                          : out std_logic;
    IP_TX_Data                           : out T_SLV_8;
    IP_TX_SOF                            : out std_logic;
    IP_TX_EOF                            : out std_logic;
    IP_TX_Ack                            : in  std_logic;
    IP_TX_Meta_rst                       : in  std_logic;
    IP_TX_Meta_SrcIPAddress_nxt          : in  std_logic;
    IP_TX_Meta_SrcIPAddress_Data         : out T_SLV_8;
    IP_TX_Meta_DestIPAddress_nxt         : in  std_logic;
    IP_TX_Meta_DestIPAddress_Data        : out T_SLV_8;
    IP_TX_Meta_Length                    : out T_SLV_16;
    -- to IP layer
    IP_RX_Valid                          : in  std_logic;
    IP_RX_Data                           : in  T_SLV_8;
    IP_RX_SOF                            : in  std_logic;
    IP_RX_EOF                            : in  std_logic;
    IP_RX_Ack                            : out std_logic;
    IP_RX_Meta_rst                       : out std_logic;
    IP_RX_Meta_SrcMACAddress_nxt         : out std_logic;
    IP_RX_Meta_SrcMACAddress_Data        : in  T_SLV_8;
    IP_RX_Meta_DestMACAddress_nxt        : out std_logic;
    IP_RX_Meta_DestMACAddress_Data       : in  T_SLV_8;
    IP_RX_Meta_EthType                   : in  T_SLV_16;
    IP_RX_Meta_SrcIPAddress_nxt          : out std_logic;
    IP_RX_Meta_SrcIPAddress_Data         : in  T_SLV_8;
    IP_RX_Meta_DestIPAddress_nxt         : out std_logic;
    IP_RX_Meta_DestIPAddress_Data        : in  T_SLV_8;
--    IP_RX_Meta_TrafficClass              : in  T_SLV_8;
--    IP_RX_Meta_FlowLabel                 : in  T_SLV_24;
    IP_RX_Meta_Length                    : in  T_SLV_16;
    IP_RX_Meta_Protocol                  : in  T_SLV_8;
    -- from upper layer
    TX_Valid                             : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_Data                              : in  T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    TX_SOF                               : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_EOF                               : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_Ack                               : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_Meta_rst                          : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_Meta_SrcIPAddress_nxt             : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_Meta_SrcIPAddress_Data            : in  T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    TX_Meta_DestIPAddress_nxt            : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    TX_Meta_DestIPAddress_Data           : in  T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    TX_Meta_SrcPort                      : in  T_SLVV_16(PORTPAIRS'length - 1 downto 0);
    TX_Meta_DestPort                     : in  T_SLVV_16(PORTPAIRS'length - 1 downto 0);
    TX_Meta_Length                       : in  T_SLVV_16(PORTPAIRS'length - 1 downto 0);
    -- to upper layer
    RX_Valid                             : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Data                              : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    RX_SOF                               : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_EOF                               : out std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Ack                               : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Meta_rst                          : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Meta_SrcMACAddress_nxt            : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Meta_SrcMACAddress_Data           : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    RX_Meta_DestMACAddress_nxt           : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
```

```
    RX_Meta_DestMACAddress_Data       : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    RX_Meta_EthType                   : out T_SLVV_16(PORTPAIRS'length - 1 downto 0);
    RX_Meta_SrcIPAddress_nxt          : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Meta_SrcIPAddress_Data         : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    RX_Meta_DestIPAddress_nxt         : in  std_logic_vector(PORTPAIRS'length - 1 downto 0);
    RX_Meta_DestIPAddress_Data        : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
--    RX_Meta_TrafficClass              : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
--    RX_Meta_FlowLabel                 : out T_SLVV_24(PORTPAIRS'length - 1 downto 0);
    RX_Meta_Length                    : out T_SLVV_16(PORTPAIRS'length - 1 downto 0);
    RX_Meta_Protocol                  : out T_SLVV_8(PORTPAIRS'length - 1 downto 0);
    RX_Meta_SrcPort                   : out T_SLVV_16(PORTPAIRS'length - 1 downto 0);
    RX_Meta_DestPort                  : out T_SLVV_16(PORTPAIRS'length - 1 downto 0)
  );
end entity;
```

Source file: net/udp/udp_Wrapper.vhdl

net_FrameChecksum

Todo

No documentation available.

Entity Declaration:

```
entity net_FrameChecksum is
  generic (
    MAX_FRAMES                    : positive        := 8;
    MAX_FRAME_LENGTH              : positive        := 2048;
    META_BITS                     : T_POSVEC        := (0 => 8);
    META_FIFO_DEPTH               : T_POSVEC        := (0 => 16)
  );
  port (
    Clock                         : in  std_logic;
    Reset                         : in  std_logic;
    -- IN port
    In_Valid                      : in  std_logic;
    In_Data                       : in  T_SLV_8;
    In_SOF                        : in  std_logic;
    In_EOF                        : in  std_logic;
    In_Ack                        : out std_logic;
    In_Meta_rst                   : out std_logic;
    In_Meta_nxt                   : out std_logic_vector(META_BITS'length - 1 downto 0);
    In_Meta_Data                  : in  std_logic_vector(isum(META_BITS) - 1 downto 0);
    -- OUT port
    Out_Valid                     : out std_logic;
    Out_Data                      : out T_SLV_8;
    Out_SOF                       : out std_logic;
    Out_EOF                       : out std_logic;
    Out_Ack                       : in  std_logic;
    Out_Meta_rst                  : in  std_logic;
    Out_Meta_nxt                  : in  std_logic_vector(META_BITS'length - 1 downto 0);
    Out_Meta_Data                 : out std_logic_vector(isum(META_BITS) - 1 downto 0);
    Out_Meta_Length               : out T_SLV_16;
    Out_Meta_Checksum             : out T_SLV_16
  );
end entity;
```

Source file: net/net_FrameChecksum.vhdl

FrameLoopback

Todo

No documentation available.

Entity Declaration:

```
entity FrameLoopback is
  generic (
    DATA_BW                     : positive        := 8;
    META_BW                     : natural         := 0
  );
  port (
    Clock                       : in  std_logic;
    Reset                       : in  std_logic;

    In_Valid                    : in  std_logic;
    In_Data                     : in  std_logic_vector(DATA_BW - 1 downto 0);
    In_Meta                     : in  std_logic_vector(META_BW - 1 downto 0);
    In_SOF                      : in  std_logic;
    In_EOF                      : in  std_logic;
    In_Ack                      : out std_logic;


    Out_Valid                   : out std_logic;
    Out_Data                    : out std_logic_vector(DATA_BW - 1 downto 0);
    Out_Meta                    : out std_logic_vector(META_BW - 1 downto 0);
    Out_SOF                     : out std_logic;
    Out_EOF                     : out std_logic;
    Out_Ack                     : in  std_logic
  );
end entity;
```

Source file: net/net_FrameLoopback.vhdl

2.4.13 sort

These are sorting entities....

Sub-Namespaces

- PoC.sort.sortnet

Entities

- PoC.sort.ExpireList
- PoC.sort.InsertSort
- PoC.sort.LeastFrequentlyUsed
- PoC.sort.lru_cache
- PoC.sort.lru_list

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

sortnet_BitonicSort

This sorting network uses the *bitonic sort* algorithm.

Entity Declaration:

```
entity sortnet_BitonicSort is
  generic (
    INPUTS                : positive  := 32;       -- input count
    KEY_BITS              : positive  := 32;       -- the first KEY_BITS of In_Data are used as a
    DATA_BITS             : positive  := 64;       -- inclusive KEY_BITS
    META_BITS             : natural   := 2;        -- additional bits, not sorted but delayed as l
    PIPELINE_STAGE_AFTER  : natural   := 2;        -- add a pipline stage after n sorting stages
    ADD_INPUT_REGISTERS   : boolean   := FALSE;    --
    ADD_OUTPUT_REGISTERS  : boolean   := TRUE      --
  );
  port (
    Clock       : in  std_logic;
    Reset       : in  std_logic;

    Inverse     : in  std_logic   := '0';

    In_Valid    : in  std_logic;
    In_IsKey    : in  std_logic;
    In_Data     : in  T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
    In_Meta     : in  std_logic_vector(META_BITS - 1 downto 0);

    Out_Valid   : out std_logic;
    Out_IsKey   : out std_logic;
    Out_Data    : out T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
    Out_Meta    : out std_logic_vector(META_BITS - 1 downto 0)
  );
end entity;
```

Source file: sort/sortnet/sortnet_BitonicSort.vhdl

sortnet_MergeSort_Streamed

Todo

No documentation available.

Entity Declaration:

```
entity sortnet_MergeSort_Streamed is
  generic (
    FIFO_DEPTH  : positive    := 32;
    KEY_BITS    : positive    := 32;
    DATA_BITS   : positive    := 32
  );
  port (
    Clock     : in  std_logic;
    Reset     : in  std_logic;

    Inverse   : in  std_logic     := '0';

    In_Valid  : in  std_logic;
    In_Data   : in  std_logic_vector(DATA_BITS - 1 downto 0);
    In_SOF    : in  std_logic;
    In_IsKey  : in  std_logic;
    In_EOF    : in  std_logic;
    In_Ack    : out std_logic;

    Out_Sync  : out std_logic;
    Out_Valid : out std_logic;
    Out_Data  : out std_logic_vector(DATA_BITS - 1 downto 0);
    Out_SOF   : out std_logic;
    Out_IsKey : out std_logic;
    Out_EOF   : out std_logic;
    Out_Ack   : in  std_logic
  );
end entity;
```

Source file: sort/sortnet/sortnet_MergeSort_Streamed.vhdl

sortnet_OddEvenMergeSort

Todo

No documentation available.

Entity Declaration:

```
entity sortnet_OddEvenMergeSort is
  generic (
    INPUTS                : positive  := 128;    -- input count
    KEY_BITS              : positive  := 32;     -- the first KEY_BITS of In_Data are used as a so
    DATA_BITS             : positive  := 32;     -- inclusive KEY_BITS
    META_BITS             : natural   := 2;      -- additional bits, not sorted but delayed as lon
    PIPELINE_STAGE_AFTER  : natural   := 2;      -- add a pipline stage after n sorting stages
    ADD_INPUT_REGISTERS   : boolean   := FALSE; --
    ADD_OUTPUT_REGISTERS  : boolean   := TRUE   --
  );
  port (
    Clock       : in  std_logic;
    Reset       : in  std_logic;

    Inverse     : in  std_logic   := '0';

    In_Valid    : in  std_logic;
```

```
    In_IsKey     : in  std_logic;
    In_Data      : in  T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
    In_Meta      : in  std_logic_vector(META_BITS - 1 downto 0);

    Out_Valid    : out std_logic;
    Out_IsKey    : out std_logic;
    Out_Data     : out T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
    Out_Meta     : out std_logic_vector(META_BITS - 1 downto 0)
  );
end entity;
```

Source file: sort/sortnet/sortnet_OddEvenMergeSort.vhdl

sortnet_OddEvenSort

Todo

No documentation available.

Entity Declaration:

```
entity sortnet_OddEvenSort is
  generic (
    INPUTS                : positive  := 8;      -- input count
    KEY_BITS              : positive  := 32;     -- the first KEY_BITS of In_Data are used as a so
    DATA_BITS             : positive  := 32;     -- inclusive KEY_BITS
    META_BITS             : natural   := 2;      -- additional bits, not sorted but delayed as lon
    PIPELINE_STAGE_AFTER  : natural   := 2;      -- add a pipline stage after n sorting stages
    ADD_INPUT_REGISTERS   : boolean   := FALSE; --
    ADD_OUTPUT_REGISTERS  : boolean   := TRUE   --
  );
  port (
    Clock        : in  std_logic;
    Reset        : in  std_logic;

    Inverse      : in  std_logic   := '0';

    In_Valid     : in  std_logic;
    In_IsKey     : in  std_logic;
    In_Data      : in  T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
    In_Meta      : in  std_logic_vector(META_BITS - 1 downto 0);

    Out_Valid    : out std_logic;
    Out_IsKey    : out std_logic;
    Out_Data     : out T_SLM(INPUTS - 1 downto 0, DATA_BITS - 1 downto 0);
    Out_Meta     : out std_logic_vector(META_BITS - 1 downto 0)
  );
end entity;
```

Source file: sort/sortnet/sortnet_OddEvenSort.vhdl

sortnet_Stream_Adapter

Todo

No documentation available.

Entity Declaration:

```
entity sortnet_Stream_Adapter is
  generic (
    STREAM_DATA_BITS      : positive        := 32;
    STREAM_META_BITS      : positive        := 2;
    SORTNET_IMPL          : T_SORTNET_IMPL  := SORT_SORTNET_IMPL_ODDEVEN_MERGESORT;
    SORTNET_SIZE          : positive        := 32;
    SORTNET_KEY_BITS      : positive        := 32;
    SORTNET_DATA_BITS     : natural         := 32;
    INVERSE               : boolean         := FALSE
  );
  port (
    Clock       : in  std_logic;
    Reset       : in  std_logic;

    In_Valid    : in  std_logic;
    In_IsKey    : in  std_logic;
    In_Data     : in  std_logic_vector(STREAM_DATA_BITS - 1 downto 0);
    In_Meta     : in  std_logic_vector(STREAM_META_BITS - 1 downto 0);
    In_Ack      : out std_logic;

    Out_Valid   : out std_logic;
    Out_IsKey   : out std_logic;
    Out_Data    : out std_logic_vector(STREAM_DATA_BITS - 1 downto 0);
    Out_Meta    : out std_logic_vector(STREAM_META_BITS - 1 downto 0);
    Out_Ack     : in  std_logic
  );
end entity;
```

Source file: sort/sortnet/sortnet_Stream_Adapter.vhdl

sortnet_Stream_Adapter2

Todo

No documentation available.

Entity Declaration:

```
entity sortnet_Stream_Adapter2 is
  generic (
    STREAM_DATA_BITS      : positive        := 32;
    STREAM_META_BITS      : positive        := 2;
    DATA_COLUMNS          : positive        := 2;
    SORTNET_IMPL          : T_SORTNET_IMPL  := SORT_SORTNET_IMPL_ODDEVEN_MERGESORT;
    SORTNET_SIZE          : positive        := 32;
    SORTNET_KEY_BITS      : positive        := 32;
    SORTNET_DATA_BITS     : natural         := 32;
    SORTNET_REG_AFTER     : natural         := 2;
    MERGENET_STAGES       : positive        := 2
  );
  port (
    Clock       : in  std_logic;
```

```
    Reset        : in  std_logic;

    Inverse      : in  std_logic       := '0';

    In_Valid     : in  std_logic;
    In_Data      : in  std_logic_vector(STREAM_DATA_BITS - 1 downto 0);
    In_Meta      : in  std_logic_vector(STREAM_META_BITS - 1 downto 0);
    In_SOF       : in  std_logic;
    In_IsKey     : in  std_logic;
    In_EOF       : in  std_logic;
    In_Ack       : out std_logic;

    Out_Valid    : out std_logic;
    Out_Data     : out std_logic_vector(STREAM_DATA_BITS - 1 downto 0);
    Out_Meta     : out std_logic_vector(STREAM_META_BITS - 1 downto 0);
    Out_SOF      : out std_logic;
    Out_IsKey    : out std_logic;
    Out_EOF      : out std_logic;
    Out_Ack      : in  std_logic
  );
end entity;
```

Source file: sort/sortnet/sortnet_Stream_Adapter2.vhdl

sortnet_Transform

Todo

No documentation available.

Entity Declaration:

```
entity sortnet_Transform is
  generic (
    ROWS         : positive    := 16;
    COLUMNS      : positive    := 4;
    DATA_BITS    : positive    := 8
  );
  port (
    Clock      : in  std_logic;
    Reset      : in  std_logic;

    In_Valid   : in  std_logic;
    In_Data    : in  T_SLM(ROWS - 1 downto 0, DATA_BITS - 1 downto 0);
    In_SOF     : in  std_logic;
    In_EOF     : in  std_logic;

    Out_Valid  : out std_logic;
    Out_Data   : out T_SLM(COLUMNS - 1 downto 0, DATA_BITS - 1 downto 0);
    Out_SOF    : out std_logic;
    Out_EOF    : out std_logic
  );
end entity;
```

Source file: sort/sortnet/sortnet_Transform.vhdl

list_expire

Todo

No documentation available.

Entity Declaration:

Source file: sort/sort_ExpireList.vhdl

list_lru_systolic

Todo

No documentation available.

Entity Declaration:

Source file: sort/sort_InsertSort.vhdl

sort_LeastFrequentlyUsed

Todo

No documentation available.

Entity Declaration:

Source file: sort/sort_LeastFrequentlyUsed.vhdl

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 indizes from 0 to ELEMENTS-1. The least-recently used index `KeyOut` is always valid.

The first outputed 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:

```
entity sort_lru_cache is
  generic (
    ELEMENTS        : positive          := 32
  );
  port (
    Clock    : in std_logic;
    Reset    : in std_logic;

    Insert   : in  std_logic;
    Free     : in  std_logic;
    KeyIn    : in  std_logic_vector(log2ceilnz(ELEMENTS) - 1 downto 0);

    KeyOut   : out std_logic_vector(log2ceilnz(ELEMENTS) - 1 downto 0)
  );
end entity;
```

Source file: sort/sort_lru_cache.vhdl

sort_lru_list

List storing `(key, value)` pairs. The least-recently inserted pair is outputed 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:

```
entity sort_lru_list is
  generic (
    ELEMENTS                : positive                      := 16;
    KEY_BITS                : positive                      := 4;
    DATA_BITS               : positive                      := 8;
    INITIAL_ELEMENTS        : T_SLM                         := (0 to 15 => (0 to 7 => '0'));
    INITIAL_VALIDS          : std_logic_vector              := (0 to 15 => '0')
  );
  port (
    Clock                   : in  std_logic;
    Reset                   : in  std_logic;

    Insert                  : in  std_logic;
    Remove                  : in  std_logic;
    DataIn                  : in  std_logic_vector(DATA_BITS - 1 downto 0);

    Valid                   : out std_logic;
    DataOut                 : out std_logic_vector(DATA_BITS - 1 downto 0)
  );
end entity;
```

Source file: sort/sort_lru_list.vhdl

2.4.14 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
- PoC.xil.SystemMonitor_Virtex6
- PoC.xil.SystemMonitor_Series7

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.

reconfig

These are reconfig entities....

Entities

- PoC.xil.reconfig.icap_fsm
- PoC.xil.reconfig.icap_wrapper

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:

```
entity reconfig_icap_fsm is
  port  (
    clk       : in  std_logic;
    reset     : in  std_logic;            -- high-active reset
    -- interface to connect to the icap
    icap_in     : out std_logic_vector(31 downto 0);  -- data that will go into the icap
    icap_out    : in  std_logic_vector(31 downto 0);  -- data from the icap
    icap_csb    : out std_logic;
    icap_rw     : out std_logic;

    -- data interface, no internal fifos
    in_data     : in  std_logic_vector(31 downto 0);  -- new configuration data
    in_data_valid : in  std_logic;            -- input data is valid
    in_data_rden  : out std_logic;            -- possible to send data
    out_data    : out std_logic_vector(31 downto 0);  -- data read from the fifo
    out_data_valid  : out std_logic;            -- data from icap is valid
    out_data_full : in  std_logic;            -- receiving buffer is full, halt icap

    -- control structures
    status      : out std_logic_vector(31 downto 0) -- status vector
  );
end reconfig_icap_fsm;
```

Source file: xil/reconfig/reconfig_icap_fsm.vhdl

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:

```
entity reconfig_icap_wrapper is
  generic (
    MIN_DEPTH_OUT     : positive := 256;
    MIN_DEPTH_IN      : positive := 256
  );
  port (
    clk       : in  std_logic;
    reset     : in  std_logic;
    clk_icap    : in  std_logic;    -- clock signal for ICAP, max 100 MHz (double check with manua

    icap_busy   : out std_logic;    -- the ICAP is processing the data
    icap_readback : out std_logic;    -- high during a readback
    icap_partial_res: out std_logic;    -- high during reconfiguration

    -- data in
    write_put   : in  std_logic;
    write_full    : out std_logic;
    write_data    : in  std_logic_vector(31 downto 0);
    write_done    : in  std_logic;    -- high pulse/edge after all data was written

    -- data out
    read_got    : in  std_logic;
    read_valid    : out std_logic;
    read_data     : out std_logic_vector(31 downto 0)
  );
end reconfig_icap_wrapper;
```

Source file: xil/reconfig/reconfig_icap_wrapper.vhdl

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:

```
entity xil_BSCAN is
  generic (
    JTAG_CHAIN          : natural;
    DISABLE_JTAG        : boolean     := FALSE
  );
  port (
    Reset               : out std_logic;
    RunTest             : out std_logic;
    Sel                 : out std_logic;
    Capture             : out std_logic;
```

```
    drck                : out std_logic;
    Shift               : out std_logic;
    Test_Clock          : out std_logic;
    Test_DataIn         : out std_logic;
    Test_DataOut        : in  std_logic;
    Test_ModeSelect     : out std_logic;
    Update              : out std_logic
  );
end entity;
```

Source file: xil/xil_BSCAN.vhdl

xil_ChipScopeICON

This module wraps 15 ChipScope ICON IP core netlists generated from ChipScope ICON xco files. The generic parameter `PORTS` selects the apropriate 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:

```
entity xil_ChipScopeICON is
  generic (
    PORTS       : positive
  );
  port (
    ControlBus  : inout T_XIL_CHIPSCOPE_CONTROL_VECTOR(PORTS - 1 downto 0)
  );
end entity;
```

Source file: xil/xil_ChipScopeICON.vhdl

See also:

Using PoC -> Synthesis For how to run synthesis with PoC and CoreGenerator.

xil_DRP_BusMux

Todo

No documentation available.

Entity Declaration:

Source file: xil/xil_DRP_BusMux.vhdl

xil_DRP_BusSync

Todo

No documentation available.

Entity Declaration:

Source file: xil/xil_DRP_BusSync.vhdl

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:

```
entity xil_ICAP is
  generic (
    ICAP_WIDTH  : string := "X32";          -- Specifies the input and output data width to be us
                               -- Spartan 6: fixed to 16 bit
                               -- Virtex 4:  X8 or X32
                               -- Rest: X8, X16, X32
    DEVICE_ID : bit_vector := X"1234567";       -- pre-programmed Device ID value for simulation
                               -- supported by Spartan 6, Virtex 6 and above
    SIM_CFG_FILE_NAME : string  := "NONE"     -- Raw Bitstream (RBT) file to be parsed by the sim
                               -- supported by Spartan 6, Virtex 6 and above
  );
  port (
    clk     : in std_logic;           -- up to 100 MHz (Virtex-6 and above, Virtex-5??)
    disable   : in std_logic;           -- low active enable -> high active disable
    rd_wr   : in std_logic;           -- 0 - write, 1 - read
    busy    : out std_logic;            -- on Series-7 devices always '0'
    data_in   : in std_logic_vector(31 downto 0); -- on Spartan-6 only 15 downto 0
    data_out  : out std_logic_vector(31 downto 0) -- on Spartan-6 only 15 downto 0
  );
end entity;
```

Source file: xil/xil_ICAP.vhdl

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:

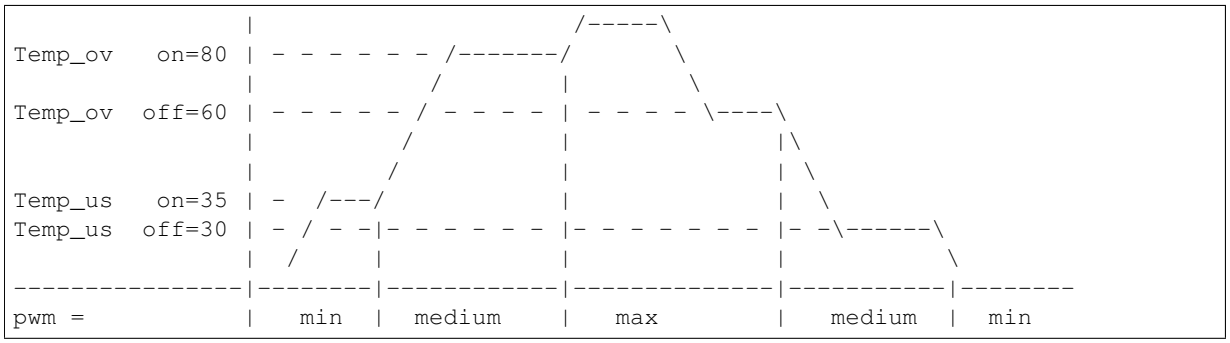
```
entity xil_Reconfigurator is
  generic (
    DEBUG           : boolean                   := FALSE;                                                   --
    CLOCK_FREQ      : FREQ                      := 100 MHz;                                                 --
    CONFIG_ROM      : in  T_XIL_DRP_CONFIG_ROM  := (0 downto 0 => C_XIL_DRP_CONFIG_SET_EMPTY)               --
  );
  port (
    Clock           : in  std_logic;
    Reset           : in  std_logic;

    Reconfig        : in  std_logic;                                                                        --
    ReconfigDone    : out std_logic;                                                                        --
    ConfigSelect    : in  std_logic_vector(log2ceilnz(CONFIG_ROM'length) - 1 downto 0);                     --

    DRP_en          : out std_logic;                                                                        --
    DRP_Address     : out T_XIL_DRP_ADDRESS;                                                                --
    DRP_we          : out std_logic;                                                                        --
    DRP_DataIn      : in  T_XIL_DRP_DATA;                                                                   --
    DRP_DataOut     : out T_XIL_DRP_DATA;                                                                   --
    DRP_Ack         : in  std_logic                                                                         --
  );
end entity;
```

Source file: xil/xil_Reconfigurator.vhdl

xil_SystemMonitor

This module generates a PWM signal for a 3-pin (transistor controlled) or 4-pin fan header. The FPGAs temperature is read from device specific system monitors (normal, user temperature, over temperature).

For example the Xilinx System Monitors are configured as follows:



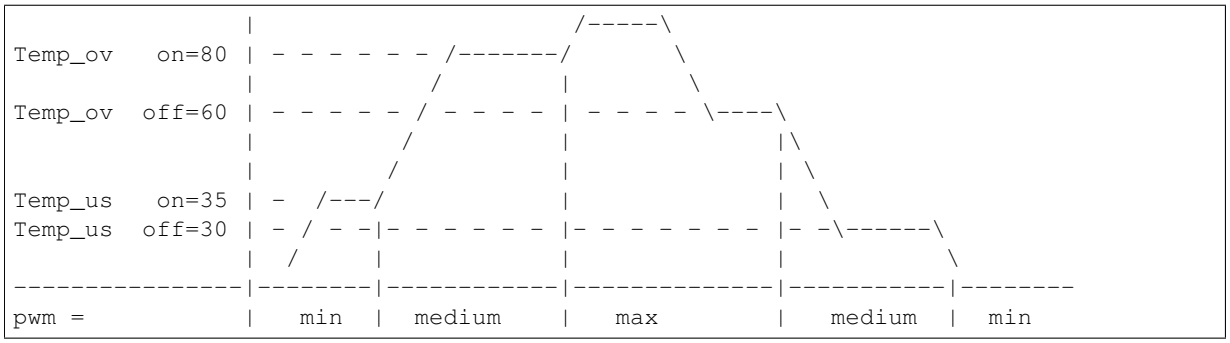
Entity Declaration:

Source file: xil/xil_SystemMonitor.vhdl

xil_SystemMonitor_Virtex6

This module wraps a Virtex-6 System Monitor primitive to report if preconfigured temperature values are overrun.

Temperature Curve



Entity Declaration:

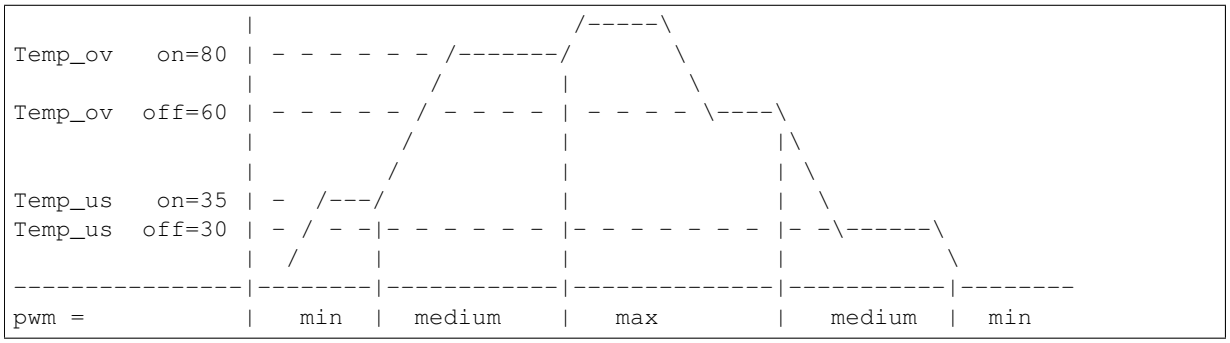
```
entity xil_SystemMonitor_Virtex6 is
  port (
    Reset               : in  std_logic;        -- Reset signal for the System Monitor control lo

    Alarm_UserTemp      : out std_logic;        -- Temperature-sensor alarm output
    Alarm_OverTemp      : out std_logic;        -- Over-Temperature alarm output
    Alarm               : out std_logic;        -- OR'ed output of all the Alarms
    VP                  : in  std_logic;        -- Dedicated Analog Input Pair
    VN                  : in  std_logic
  );
end entity;
```

Source file: xil/xil_SystemMonitor_Virtex6.vhdl

xil_SystemMonitor_Series7

This module wraps a Series-7 XADC to report if preconfigured temperature values are overrun. The XADC was formerly known as “System Monitor”.

Temperature Curve



Entity Declaration:

```
entity xil_SystemMonitor_Series7 is
  port (
    Reset               : in  std_logic;        -- Reset signal for the System Monitor control lo

    Alarm_UserTemp      : out std_logic;        -- Temperature-sensor alarm output
    Alarm_OverTemp      : out std_logic;        -- Over-Temperature alarm output
    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_Series7.vhdl

2.5 Third Party Libraries

The PoC-Library is shiped with different third party libraries, which are located in the `<PoCRoot>/lib/` folder. This document lists all these libraries, their websites and licenses.

2.5.1 Cocotb

Cocotb is a coroutine based cosimulation library for writing VHDL and Verilog testbenches in Python.

Folder:	`<PoCRoot>\lib\cocotb\`
Copyright:	Copyright © 2013, Potential Ventures Ltd., SolarFlare Communications Inc.
License:	Revised BSD License (local copy)
Documentation:	http://cocotb.readthedocs.org/
Source:	https://github.com/potentialventures/cocotb

2.5.2 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:	`<PoCRoot>\lib\osvvm\`
Copyright:	Copyright © 2012-2016 by SynthWorks Design Inc.
License:	Artistic License 2.0 (local copy)
Website:	http://osvvm.org/
Source:	https://github.com/JimLewis/OSVVM

2.5.3 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:	Mozilla Public License, Version 2.0 (local copy)
Website:	https://vunit.github.io/
Source:	https://github.com/VUnit/vunit

2.5.4 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:

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

On OS X

Please see the Linux instructions.

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 ..
}
```

2.6 Constraint Files

2.6.1 IP Core Contraint Files

- fifo
- **misc**
  - sync
- **net**
  - eth

fifo

- fifo_ic_got

fifo_ic_got

misc

- sync

sync

- sync_Bits
- sync_Reset
- sync_Vector
- sync_Command

fifo_ic_got

fifo_ic_got

fifo_ic_got

fifo_ic_got

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

2.6.2 Board Contraint Files

- **Altera Boards**
  - Cyclone III
  - Stratix IV
  - Stratix V
- Lattice Boards
- **Xilinx Boards**
  - Spartan-3 Boards
  - Spartan-6 Boards
  - Artix-7
  - Kintex-7

  - Virtex-5
  - Virtex-6
  - Virtex-7
  - Zynq-7000

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

Lattice

- ECP5 * ECP5 Versa

ECP5

- ECP5 Versa

ECP5 Versa

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

2.7 References

2.7.1 Command Reference

Contents of this Page

Headline 4

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 5

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 6

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

2.7.2 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.

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.

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
```

2.7.3 File Formats

*.files Format

Contents of this Page

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.

*.rules Format

Contents of this Page

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

2.7.4 Interfaces

PoC.CSE

Todo

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

PoC.Mem

Todo

Define the PoC.Memory interface used in ...

PoC.Stream

Todo

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

2.7.5 Naming Conventions

Todo

Write an intruduction paragraph for this page.

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 constaint 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 choosen 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 hierachy 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 miscelaneous 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.

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	PoC.fifo.cc_got
VHDL entity name	fifo_cc_got
File name	fifo_cc_got.vhdl
IP Core Description File	\src\fifo\fifo_cc_got.files
Source File Location	\src\fifo\fifo_cc_got.vhdl
Testbench Location	\tb\fifo\fifo_cc_got_tb.vhdl
Testbench Description File	\tb\fifo\fifo_cc_got_tb.files
Waveform Description Files	\sim\fifo\fifo_cc_got_tb.*

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.

Signal Names

Todo

No documentation available.

2.7.6 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

2.7.7 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>**	

2.7.8 Glossary

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

cc - Common clock All ports of a module use the same clock.

dc - Dependent clock The clock inputs of a module have a known relation in phase or are multiples of a shared base clock.

***flag*-signal** No documentation available.

FWFT - First-word-fall-through No documentation available.

ic - Independent clock The clock inputs have no known relation and are considdered independent. Modules with ic interfaces implement clock domain crossing (CDC) circuits.

OCRAM - On-Chip RAM, OCROM - On-Chip ROM An On-Chip RAM is a embedded memory block, mostly called BlockRAM, Dirstributed Memory, ...

PoC.CSE - Command-Status-Error A control and monitoring protocol in a layer-based architecture.

PoC.Stream A streaming optimized, FIFO-like on-chip protocol.

PoCRoot The PoC root directory.

ProjectRoot The project's root directory, which hosts PoC.

***strobe*-signal** No documentation available.

2.7.9 Known Issues

Aldec

Active-HDL Student-Edition

- Aliases to functions and protected type methods

Altera

Quartus-II

- Generic types of type strings filled with NUL

GHDL

- Aliases to protected type methods

Xilinx

ISE

- Shared Variables in Simulation (VHDL-93)

Vivado

- Physical types in synthesis
- VHDL-2008 mode in simulation
- Shared variables in simulation (VHDL-93 and VHDL-2008))

2.7.10 Local License Copies

This documentation contains local copies of all used licenses by either PoC itself or embedded libraries or IP cores. Each formatted [1] license text can be downloaded from the original source. Therefor see the top-most INFO box, which contains a link to the orginal license file source.

Note: This is a local copy of the Apache License Version 2.0.

Apache License 2.0

Version 2.0, January 2004

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

[1] Formatted means, that reStructured Test (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).

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.
```

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.

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 li-

cense 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.

Artistic License 2.0

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.

BSD License for Cocotb

Cocotb is licensed under the Revised BSD License. Full license text below.

Copyright (c) 2013 Potential Ventures Ltd Copyright (c) 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 Mozilla Public License, Version 2.0.

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.
```

2.8 Get Involved

A first step might be to use and explore PoC and it's infrastructure in an own project. Moreover, we encurage 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 interresting project, please send us feedback or get listed on our Who uses PoC?

If you are more familiar with PoC and it's components, you might start asking youself 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 interessted 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 interessted in getting your library or components shipped as part of PoC or as a third-party components, please contact us.

2.8.1 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 closed. All enhancements and feature requests are tracked on GitHub at GitHub Issues.

2.8.2 Feature Request

Please report missing features of any kind. We are allways 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.

2.8.3 Talk to us on Gitter

You can chat with us on Gitter in our Giiter Room VLSI-EDA/PoC. You can use Gitter for free with your GitHub account.

2.8.4 Contributers License Agreement

We require all contributers 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 can login with your GitHub account.

2.8.5 Contribute to PoC

Contibuting 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 you changes in your own repository and notify us about outstanding changes via pull-requests.

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

1. Fork our Repository

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

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`.

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. Checkout a Branch

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

```
cd PoCRoot
git checkout master
```

4. Setup PoC for Developers

Run PoC's configuration routines and setup the developer tools. You can skip (`P`) all tool chain questions until you reach the Git questions.

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

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.

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 `utils.vhdl`
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>
```

7. Commit and Push Changes

Commit your porposed changes to 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>
```

8. Create a Pull-Request

Go to your forked repository and klick 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 Contributers License Agreement (CLA).

9. Keep your `master` up-to-date

Todo

undocumented

2.8.6 Give us Feedback

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

2.8.7 List of Contributers

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
Preußer, 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

2.9 Change Log

[1] In alphabetical order.
[2] Maintainer.

Content of this page

2.9.1 2016

New in 1.x (upcomming)

Already documented changes are available on the `release` branch at GitHub.

- Python Infrastructure
  - Common changes
    - The classes `Simulator` and `Compiler` now share common methods in base class called `Shared`.
  - `*.files` Parser
    - Implemented path expressions: sub-directory expression, concatenate expression
    - Implemented InterpolateLiteral: access database keys in `*.files` files
    - New Path statement, which defines a path constant calculated from a path expression
    - Replaced string arguments in statements with path expressions if the desired string was a path
    - Replaced simple StringToken matches with Identifier expressions
  - All Simulators
    -
  - All Compilers
    -

    - GHDL
        * Reduced `-P<path>` parameters: Removed doublings
- Documentation
    -
- VHDL common packages
    -
- VHDL Simulation helpers
    - Mark a testbench as failed if (registered) processes are active while finilize is called
- New Entities
    -
- New Testbenches
    -
- New Constraints
    -
- Shipped Tool and Helper Scripts
    - Updated and new Notepad++ syntax files

New in 1.0 (13.05.2016)

- Python Infrastructure (Completely Reworked)
    - New Requirements
        * Python 3.5
        * py-flags
    - New command line interface
        * Synopsis: `poc.sh|ps1 [common options] <command> <entity> [options]`
        * Removed task specific wrapper scripts: `testbench.sh|ps1`, `netlist.sh|ps1`, ...
        * Updated `wrapper.ps1` and `wrapper.sh` files
    - New ini-file database
        *
        * Added a new config.boards.ini file to list known boards (real and virtual ones)
    - New parser for `*.files` files
        * conditional compiling (if-then-elseif-else)
        * include statement - include other `*.files` files
        * library statement - reference external VHDL libraries
        * prepared for Cocotb testbenches
    - New parser for `*.rules` files
        *
    - All Tool Flows
        * Unbuffered outputs from vendor tools (realtime output to stdout from subprocess)
        * Output filtering from vendor tools

            - verbose message suppression
            - error and warning message highlighting
            - abort flow on vendor tool errors
        - All Simulators
            - Run testbenches for different board or device configurations (see `--board` and `--device` command line options)
        - New Simulators
            - Aldec Active-HDL support (no GUI support)
                - Tested with Active-HDL from Lattice Diamond
                - Tested with Active-HDL Student Edition
            - Cocotb (with QuestaSim backend on Linux)
        - New Synthesizers
            - Altera Quartus II and Quartus Prime
                - Command: `quartus`
            - Lattice Synthesis Engine (LSE) from Diamond
                - Command: `lse`
            - Xilinx Vivado
                - Command: `vivado`
        - GHDL
            - GHDLSimulator can distinguish different backends (mcode, gcc, llvm)
            - Pre-compiled library support for GHDL
        - QuestaSim / ModelSim Altera Edition
            - Pre-compiled library support for GHDL
        - Vivado Simulator
            - Tested Vivado Simulator 2016.1 (xSim) with PoC -> still produces errors or false results
- New Entities
    -
- New Testbenches
    -
- New Constraints
    -
- New dependencies
    - Embedded Cocotb in `<PoCRoot>/lib/cocotb`
- Shipped Tool and Helper Scripts
    - Updated and new Notepad++ syntax files
    - Pre-compiled vendor library support
        - Added a new `<PoCRoot>/temp/precompiled` folder for precompiled vendor libraries
        - QuestaSim supports Altera QuartusII, Xilinx ISE and Xilinx Vivado libraries
        - GHDL supports Altera QuartusII, Xilinx ISE and Xilinx Vivado libraries

New in 0.21 (17.02.2016)

New in 0.20 (16.01.2016)

New in 0.19 (16.01.2016)

2.9.2 2015

New in 0.18 (16.12.2015)

New in 0.17 (08.12.2015)

New in 0.16 (01.12.2015)

New in 0.15 (13.11.2015)

New in 0.14 (28.09.2015)

New in 0.13 (04.09.2015)

New in 0.12 (25.08.2015)

New in 0.11 (07.08.2015)

New in 0.10 (23.07.2015)

New in 0.9 (21.07.2015)

New in 0.8 (03.07.2015)

New in 0.7 (27.06.2015)

New in 0.6 (09.06.2015)

New in 0.5 (27.05.2015)

- Updated Python infrastructure
- New testbenches:
  - sync_Reset_tb
  - sync_Flag_tb
  - sync_Strobe_tb
  - sync_Vector_tb
  - sync_Command_tb
- Updated modules:
  - sync_Vector
  - sync_Command
- Updated packages:
  - physical
  - utils
  - vectors

  - xil

New in 0.4 (29.04.2015)

- New Python infrastructure
  - Added simulators for:
    - GHDL + GTKWave
    - Mentor Graphic QuestaSim
    - Xilinx ISE Simulator
    - Xilinx Vivado Simulator
- New packages:
  - simulation
- New modules:
  - PoC.comm - communication modules
    - comm_crc
  - PoC.comm.remote - remote communication modules
    - remote_terminal_control
- New testbenches:
  - arith_addw_tb
  - arith_counter_bcd_tb
  - arith_prefix_and_tb
  - arith_prefix_or_tb
  - arith_prng_tb
- Updated packages:
  - board
  - config
  - physical
  - strings
  - utils
- Updated modules:
  - io_Debounce
  - misc_FrequencyMeasurement
  - sync_Bits
  - sync_Reset

New in 0.3 (31.03.20015)

- Added Python infrastructure
  - Added platform wrapper scripts (*.sh, *.ps1)
  - Added IP-core compiler scripts Netlist.py
- Added Tools

  - Notepad++ syntax file for Xilinx UCF/XCF files
  - Git configuration script to register global aliases
- New packages:
  - components - hardware described as functions
  - physical - physical types like frequency, memory and baudrate
  - io
- New modules:
  - PoC.misc
    - misc_FrequencyMeasurement
  - PoC.io - Low-speed I/O interfaces
    - io_7SegmentMux_BCD
    - io_7SegmentMux_HEX
    - io_FanControl
    - io_PulseWidthModulation
    - io_TimingCounter
    - io_Debounce
    - io_GlitchFilter
- New IP-cores:
  - PoC.xil - Xilinx specific modules
    - xil_ChipScopeICON_1
    - xil_ChipScopeICON_2
    - xil_ChipScopeICON_3
    - xil_ChipScopeICON_4
    - xil_ChipScopeICON_6
    - xil_ChipScopeICON_7
    - xil_ChipScopeICON_8
    - xil_ChipScopeICON_9
    - xil_ChipScopeICON_10
    - xil_ChipScopeICON_11
    - xil_ChipScopeICON_12
    - xil_ChipScopeICON_13
    - xil_ChipScopeICON_14
    - xil_ChipScopeICON_15
- New constraint files:
  - ML605
  - KC705
  - VC707
  - MetaStability
  - xil_Sync

- Updated packages:
  - board
  - config
- Updated modules:
  - xil_BSCAN

New in 0.2 (09.03.2015)

- New packages:
  - xil
  - stream
- New modules:
  - PoC.bus - Modules for busses
    - bus_Arbiter
  - PoC.bus.stream - Modules for the PoC.Stream protocol
    - stream_Buffer
    - stream_DeMux
    - stream_FrameGenerator
    - stream_Mirror
    - stream_Mux
    - stream_Source
  - PoC.misc.sync - Cross-Clock Synchronizers
    - sync_Reset
    - sync_Flag
    - sync_Strobe
    - sync_Vector
    - sync_Command
  - PoC.xil - Xilinx specific modules
    - xil_SyncBits
    - xil_SyncReset
    - xil_BSCAN
    - xil_Reconfigurator
    - xil_SystemMonitor_Virtex6
    - xil_SystemMonitor_Series7
- Updated packages:
  - utils
  - arith

New in 0.1 (19.02.2015)

- New packages:
  - board - common development board configurations
  - config - extract configuration parameters from device names
  - utils - common utility functions
  - strings - a helper package for string handling
  - vectors - a helper package for std_logic_vector and std_logic_matrix
  - arith
  - fifo
- New modules
  - PoC.arith - arithmetic modules
    - arith_counter_gray
    - arith_counter_ring
    - arith_div
    - arith_prefix_and
    - arith_prefix_or
    - arith_prng
    - arith_scaler
    - arith_sqrt
  - PoC.fifo - FIFOs
    - fifo_cc_got
    - fifo_cc_got_tempgot
    - fifo_cc_got_tempput
    - fifo_ic_got
    - fifo_glue
    - fifo_shift
  - PoC.mem.ocram - On-Chip RAMs
    - ocram_sp
    - ocram_sdp
    - ocram_esdp
    - ocram_tdp
    - ocram_wb

2.9.3 2014

New in 0.0 (16.12.2014)

- Initial commit

Note: This is a local copy of the Apache License Version 2.0.

2.10 Apache License 2.0

Version 2.0, January 2004

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

2.10.1 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.

2.10.2 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.

2.10.3 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.

2.10.4 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.

2.10.5 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.

2.10.6 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.

2.10.7 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.

2.10.8 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.

2.10.9 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.
```

C

D

F

I

O

P

S