SpinalHDL : A comprehensive HDL language

dolu1990, lambdaShade

May 21, 2016

Abstract

SpinalHDL is a new HDL language offering the ability to break the limitations of the most common hardware description languages, like *VHDL* and *Verilog*.

1 SpinalHDL Structure

SpinalHDL is a library constructed over the Scala internal DSL, allowing the user to design productively his digital hardware. The user designs can be converted to *VHDL* files for synthesis with any PLD/FPGA/ASIC specialized toolchain.

Figure 1: SpinalHDL structure layers

The Spinal library is separated in two parts : one - the **core** - providing basic elements for hardware description and another one on it - the **lib** - providing interfaces and utilities for construct powerful design and help the user in his tasks.

2 Highlights

Here's a list of the main advantages of SpinalHDL comparatively to *VHDL* and *Verilog* :

- No restriction with the genericity of your hardware description by using the power of the Scala language.
- No more endless wiring : create and connect complex buses like *AXI* in only one text line.
- Evolving capabilities : create your own buses definitions and abstraction layers.
- Reduce code size by a high factor - especially for wiring - allowing you to have a better visibility, more productivity and fewer headaches.
- Available user friendly IDE with auto-completion, error highlight, navigation shortcut and many others.
- Extract information from your digital design and generate files containing data about some latency and addresses, for example.
- Bidirectional translation between bunch of bits and any data type/structures. Useful to load a complex data structure from a CPU interface.
- Check for combinational loop/latch presence.
- Check for user unintentional cross clock domain violations.

3 Core primitives

- **Base Types** Bool, Bits, UInt, SInt, Enumeration.
- **Bundle** Allows to describe a data structure with the possibility to specify the direction (in, out) for each element. Useful to describe buses.
- **Reg** Creates a register signal.
- **Vec** Allows to create an array of data.
- **Mem** Gives the possibility to manipulate memory.
- **BlackBox** Allows to instantiate a third party HDL component.

4 Lib primitives

- **Interfaces** Flow, handshake, fragments, ...
- **Components** FIFOs, inter-clock domain bridges, ...
- **Peripherals** UART, ...
- **Utils** Gray conversions, counters, one-hot encoding, majority vote...
- **Digital design analysis** Tool for pipeline latency evaluation.

5 To go further...

- Repositories available on Github : `https://github.com/SpinalHDL`
- Some examples : `https://github.com/SpinalHDL/SpinalHDL/blob/master/README.md`