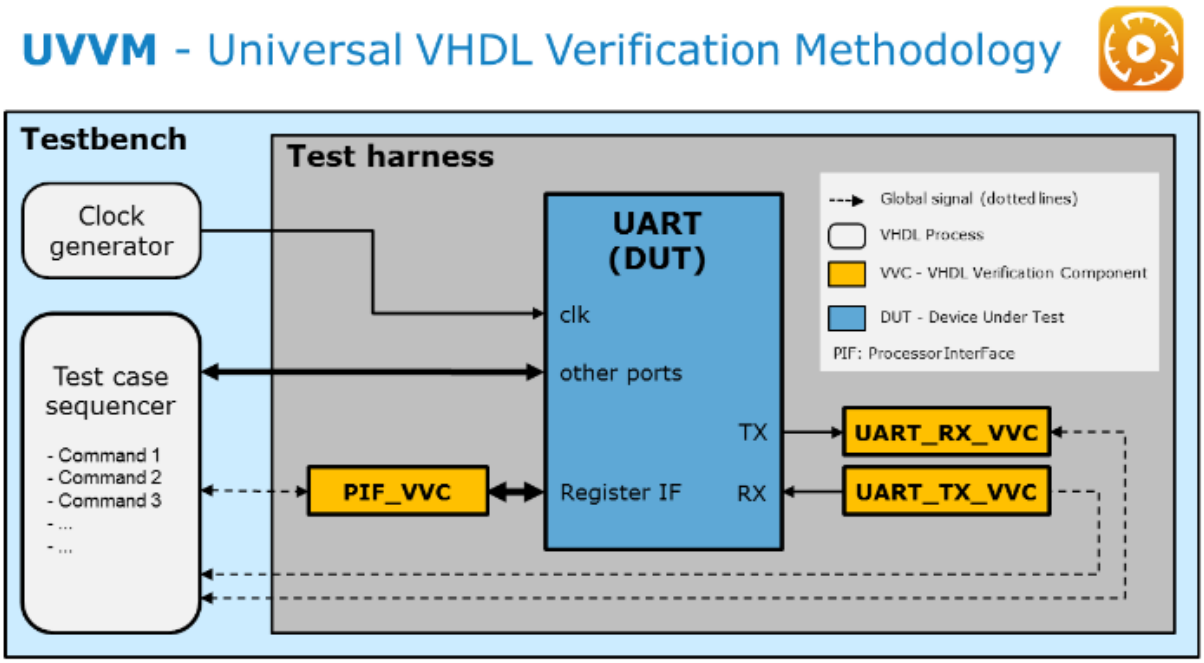

Advanced VHDL Verification
- Made simple - For anyone

Part 1: The testbench architecture

Everybody understands the schematic above. It's just a test harness with some verification components (orange) connected to the interfaces of a design component (blue). In addition, some other interfaces - including the clock - are connected in/out of the test harness to signals at the testbench top level - controlled by the two processes there.

Could you make a similar test harness for an SPI design, or an FPGA with 6 different interfaces if you had the matching 6 verification components?
In fact, you could make that FPGA test harness in less than one hour - because all you have to do is to connect the corresponding interfaces between the FPGA and the verification components, - and you are done.

For your SPI testbench you could reuse your PIF_VVC from your UART testbench, and for your FPGA top level testbench you could reuse all your VVCs from your module testbenches.

This is the overview, modifiability, extendibility, maintainability and re-use you get with UVVM VVC Framework, - a free and open source Universal VHDL Verification Methodology (UVVM) - available from github.com and bitvis.no (released in Feb. 2016).

And by the way - the dotted connections from the test sequencer to the Verification components are global signals. So they are connected by default once their packages, in which they are defined, are included in the testbench. Less work, better overview and more efficient reuse.

You might ask yourself 'how do I control these verification components to apply stimuli and check data out of my DUT?'
- The brief answer is that this is really easy via simple commands from the test sequencer like 'write(PIF_VVC, my_reg_address, my_data)' or 'transmit(UART_TX_VVC, my_data)'. This means anyone can understand and write test sequencers - just like a simple software sequencer. I'll come back to this in part 2. https://www.linkedin.com/pulse/advanced-vhdl-verification-made-simple-2-espen-tallaksen?published=u

What about the verification components?
- They have a very structured micro architecture that is almost the same for all Verification Components, making it easy to make new VVCs from an automatically generated template. I'll come back to this is part 3.

The testbench and test harness architecture is critical to understand for efficient verification. This is why I start by showing how simple a UVVM testbench will be - even for complex design modules.

Next:
Controlling stimuli and checking outputs - in sequence or simultaneously.
UVVM provides the simplest possible approach - yet extremely powerful.
A game changer for VHDL verification.
https://www.linkedin.com/pulse/advanced-vhdl-verification-made-simple-2-espen-tallaksen?published=u