Introduction to Design Verification (DV)

The TechWorks Academy present the course to introduce participants to DV

1 Course Objectives

By the end of the course, participants should be able to

1. Describe the best-practice DV strategies applied currently to semiconductor digital designs
2. Understand the main methodologies, tools and languages used in those best-practice DV strategies
3. Apply those DV methodologies, tools and languages to basic digital designs at IP and subsystem level
4. Analyse a "real" semiconductor digital design (IP and subsystem) and suggest an appropriate DV strategy
5. Understand best-practice DV so they can discuss DV topics confidently with colleagues
6. Have sufficient understanding of DV tools and methodologies to contribute effectively to real projects

2 Target audience

1. University students going on or started a placement at a semiconductor design company
2. Design engineers wanting to learn more about verification and/or how to write re-usable verification IP
3. University graduates starting or started work at a semiconductor design company
4. Engineers wanting to transition their career to or are just curious about DV
5. Managers wanting some understanding of the topic

3 Pre-requisites

There are no pre-requisites, although some experience of programming (preferably with an Object-Oriented language) would be useful. The course is suitable for a wide range of people working in semiconductor design and development, especially recent graduates.

4 Course overview and structure

The course runs for 5 weeks with the following main activities. Details are given using the links.

1. 7 online one-hour lectures covering the major DV topics (details)
2. 6 Practical sessions
3. 2 online one-hour lectures covering debug (details)
4. A large collection of design and verification examples and exercises graded as different levels of difficulty and complexity ready for running on a wide variety of simulators
   - For download so students can run them on their machines using their licenses
5. A number of pre-recorded videos covering the following: running tools; overview of design and verification languages; introduction to writing designs; overview of test bench languages, methodologies and structures;

Note that participants will get additional support outside the lectures and exercises as needed.

Preparatory and follow-on reading will be suggested but neither are obligatory. There is no (summative or formative) assessment (details).

5 Course content details

5.1 DV lectures: Details

The online DV lectures will cover the following. These sessions will include online questions and polls to increase the level of interaction.

1.	Introduction	What is Design Verification? Verification complexities Verification hierarchy Verification tools

		The cost of bugs
2.	IP, subsystem and SoC	Levels of abstraction Differences in designs at IP, subsystem and SoC Bus interfaces and connecting IP Differences in verification at IP, subsystem and SoC Difference between ASIC and FPGA design flows Introduction to the FIFO example: IP and subsystem
3.	The Verification Cycle	Verification planning Planning, milestones, tracking and completion criteria Verification metrics and closure Verification Methodologies
4.	Simulation-based Verification and Stimulus generation and checking	Foundations Manual vs. Automation Independent vs coordinated generators? Level of abstraction Online vs. offline generation Test length Randomness What and when to check Checking mechanisms including behavioural models General structure of a constrained random test bench How to identify suitable randomisation, sequences, checkers, coverage, assertions etc.
5.	Coverage	Introduction Code coverage Functional coverage Coverage analysis and closure
6.	Assertion Based Verification	Introduction Benefits and drawbacks of assertions Types of assertions Assertion libraries Writing assertions and properties
7.	Verification in practise	The verification cycle from start to finish Test bench qualification IP level verification in practise Higher level verification in practise

5.2 Practical Sessions: Details

These practical sessions are aimed at giving hands-on experience on developing test benches using a wide range of languages and methodologies. Note that there are also opportunities to develop designs[1]. All practical sessions will include interactive question and answer sessions. Where possible, sessions will be split between Verilog and VHDL

1.	Hardware design and verification languages	Overview of the exercises Overview of Hardware Description Languages (HDLs) Overview of the videos
2.	FIFO introduction to IP design and bus connection challenge	Designing IP Introduction to design challenge Overview of structure of design solution in Verilog and VHDL (including optional code walk though)
3.	FIFO design solution and introduction to IP level test bench challenge	Review of IP design in VHDL and Verilog Behavioural modelling

[1] Development of designs have been added to give participants the opportunity to better understand the designs they are trying to verify.

		Introduction to IP test bench challenge: How to identify suitable randomisation, sequences, checkers, coverage, assertions etc Overview of code structure of test bench solution
4.	FIFO IP level test bench solution and introduction to subsystem test bench challenge	Review of IP test bench solution in VHDL and Verilog[2] FIFO debug exercises and examples Introduction to subsystem test bench challenge Overview of structure of test bench solution
5.	FIFO subsystem test bench solution and CPU-based verification challenge	Review of subsystem test bench solution in VHDL and Verilog Introduction to CPU-based test bench challenge Overview of structure of test bench solution CPU-based verification solution
6.	FIFO test bench qualification and debug	Can your test bench find the FIFO bugs? Does your test bench enable faster debug?

5.3 Debug lectures: Details

The online DV lectures will cover the following

1.	Introduction to debug	What is debug and where do bugs lie? Debugging designs and verification environments Working with (not against) the design team
2.	Making debug efficient and effective	Techniques to reduce debug time Ensuring a bug is fixed Making use of “bug clustering” Ensure my verification environment is designed for ease of debug?

6 Pre-recorded videos

TBD

7 Exercises: Access to exercises and DV tools

Participants will be given access to numerous small DV examples and exercises to support what is being taught in both the DV and the debug lectures. The exercises will be available in both SystemVerilog (some using UVM) and VHDL (some using OSVVM[3]) and can be run on a wide range of simulators[4]. Exercises are also graded into “simple, medium, hard” to allow students to find exercises suited to their level of expertise:

- Small design exercises are provided for those participants who want to learn about design[5].
- Small test bench exercises are provided to allow participants to implement the concepts learned in the lectures
- There is documentation and videos explaining how to run the examples on a wide range of tools.
- The exercises will be kept small and simple due to time limitations
    - There are directed test exercises and some randomisation is added.
    - There are separate exercises for assertions, coverage and debug.
    - There are small complete test bench UVM/OSVVM exercises
    - Participants will mostly be expected to complete a small part of a design or change to a design or test bench.
    - There will be full solutions available for every exercise

[2] Adding gate level simulations (GLS) for the FIFO design are under consideration

[3] Expecting to add UVVM and cocotb too

[4] Including Cadence Xcelium, Siemens Questa, Synopsys VCS, Aldec ActiveHDL. Also expecting to add exercises for free tools such as GHDL + GTK wave for VHDL and Verilator for SV UVM

[5] As mentioned, design exercises have been added to give participants the opportunity to better understand the designs they are trying to verify. This course is not aimed at teaching hardware design.

The exercises will be available on 2 platforms: the cloud; and download. Cloud access will need to be specifically requested otherwise download will be assumed

7.1 Online in the cloud

Online tools for running and updating the exercises.

- All participants will be given access to an Amazon cloud server dedicated for use with the course
- Participants will get access to tools to allow them to run simulations, perform debug, edit test benches and measure coverage (code and functional)
    - Students will be allowed to download their solutions but uploading by students will be prohibited

7.2 Download

Students will be able to download the exercises for running on their own machines with their own tools and licenses.

8 Assessment

The following assessment activities are under consideration

- Participants who attend the full course will receive a TechWorks Academy certificate of attendance
- There will be a multiple-choice end of course test[6]

9 Proposed schedule for Q1 2023

The table below gives the schedule for the course. All sessions will run 16.00 to 17.00 UK time.

Date	Timing
Tue 7th Feb	16.00 to 17.00 UK time
Wed 8th Feb	16.00 to 17.00 UK time
Thu 9th Feb	16.00 to 17.00 UK time
Tue 14th Feb	16.00 to 17.00 UK time
Wed 15th Feb	16.00 to 17.00 UK time
Thu 16th Feb	16.00 to 17.00 UK time
Tue 21st Feb	16.00 to 17.00 UK time
Wed 22nd Feb	16.00 to 17.00 UK time
Thu 23rd Feb	16.00 to 17.00 UK time
Tue 28th Feb	16.00 to 17.00 UK time
Wed 1st March	16.00 to 17.00 UK time
Thu 2nd March	16.00 to 17.00 UK time
Tue 7th March	16.00 to 17.00 UK time
Wed 1st March	16.00 to 17.00 UK time
Thu 2nd March	16.00 to 17.00 UK time

[6] This is seen as higher value than an attendance certificate so attendees can still acquire a high value certificate even if they do not attend all sessions