**Basic Functional Verification Fundamentals**

ANSWERS

**Q1. What prevents verification engineers from creating test benches for every possible scenario in a DUV?**  
*The verification engineers have to face with huge space of state, function for ech design. With the requirement to cover all the functions, states and check each sub-module's performance, it can take a lot of time, several years.  
With the requirement, speed of product, the verification engineer cannot create test benches for every possible scenario in DUT.*  
**#Q2. Describe the role of functional verification within the chip design process based on your investigation.**  
*The verification process aims to assure that design can work as requirement and satisfy the specification. The first target is checking the design is correct or not, the second is checking any lack of the designer's idea which can lead to the lack functions in the design. There are three major problems which are affected by verification process.   
1. Quality of product:  
Test all the functions of the design. Make sure that the design can work in all condition, although there are rarely conditions. Not only the whole design can work, verification engineer has to check each module's operation, make sure that each module can work correctly as requirements. It will help the design can work exactly in detail.  
2. Cost of production cycle:   
Fix bugs as soon as possible, the faster bug is found and fix will reduce the cost of production cycle.  
3. Time to market:  
Control the verification process smoothly, because the verification process now is the bottle neck of production cycle, it takes about 70 -80% effort of time and human resource in verification process. This process can affect heavily to the time to market of production, one of the most important factor of the production's success.*

**#Q3. Why do verification engineers must find design bugs as soon as possible?**

*The sooner the bug is found, the larger fixing cost is save. With the bug is found in HDL design phase, it is easy for engineer to spend his effort by recoding some lines. But if the bug is not found and bring to physical implement phase, it can take a lot of money to fix. The worst case is if any customer can find the bug, it can affect to company's brand.*

*Clean bug as soon as possible will help the good product reaches to market on time. That is the most important to production's success.*

**#Q4. From which stage (or stages) of the verification cycle are these activities?**

*Create verification plan*

*(i) List the tools needed for verification*

*(b) Understand the interesting scenarios and corner cases in the DUV*

*(g) Discuss design intent and specification with designer and architect*

*Develop verification environment*

*(a) Write code to drive and check the DUV*

*(j) Create a reference model to check the design’s behavior*

*Debug Design and Environment*

*(e) Find bugs by using a simulation engine*

*(f) Find bugs by using an oscilloscope*

*(h) Discuss a miscompare between the HDL and the reference model with the designer*

*Do lessons learned*

*(c) Contemplate what improvements need to be made to the verification environment based on hardware results*

**#Q5. Which of the following should verification engineers do, and which should they** avoid? Please explain your answers.

*Do's:*

*(a) Talk to designers about the function and understand the design*

*Verifier needs to understand the design specification in order to write the verification plan/test scenarios. It also be useful if designers share their design intention.*

*(b) Rely on the DUV designer’s description for input/output specification*

*Where else should we rely on?*

*(d) Try to think of situations the designer might have missed*

*Kind of thinking out of the box :-). That's the corner cases the designer might have missed.*

*(g) Try to fill all queues during simulation*

*One of the corner case.*

*(h) Focus on multiple events at the same time*

*One of the main cause of problems. Architecture/designer does not anticipate all scenarios in which multiple events rise.*

*Don'ts:*

*(c) When creating checkers and reference models, look at the HDL implement (RTL) for hard-to-predict cases.*

*Should rely on the design specification only. There may be mismatches between design intention and design implementation (RTL). If verifier builds checkers and reference models referring RTL implementation, he/she will be biased which may lead to bug escape.*

*(f) After receiving a bug fix, move on to the next job in the test plan*

*Need rerun the same test to ensure the bug fix is correct and not release the new bug.*

*(e) Focus on exotic scenarios and situations*

*Not only exotic scenarios (or corner cases) but also usual cases.*

*(i) Move on to the next product after the initial tape-out because the work is complete*

*Need to proceed "do lessons learned" step, or bug escape analysis. And who ensures that there is not second, or third tape-out :-)*

*(k) Inform a bug to designer as soon as possible, then move to next jobs right after without explanation of the bug and fixing proposals.*

*Violate code of ethics for verifier.*

*(l) When a test-vector is failed in simulation, put it aside and do next job. Check it again when possible.*

*As we know the sooner bugs are found and fixed, the less fixing cost is consumed, this fail may lead us to a bug soon and save more cost for fixing it.*

*There may be bug in environment which affects the "next job".*

*There may be RTL bug which needed to be fix.*

*Well, it is acceptable only if the failed test-vector has lower priority than the others.*

**Q6. In a project, should random verification or directed verification be processed firstly? Why?**

*If time permits you should run directed verification first check. Because it helps us to check the points (usually new/updated points, …) that we want to make sure the design is good or not first, clear all the corner of design. The random test is should use after that to cover all functions of design.*