

A monitor-based approach to verification

By Bipin Patel

[TechOnline India](#)

(06/02/10, 02:43:00 AM EDT)

Abstract: Selecting verification methodology that allows maximum reusability within a project is very important to complete verification within the defined timeline. If we fail to choose an appropriate methodology then there are possible chances of work being duplicated at different levels of the verification process in the same project, which is nothing but a waste of time and resources.

Thinking through the process of selecting a verification methodology needs to take into consideration the various levels at which we need to do the verification and how verification environment (VE) components from bottom to top level (block to chip) verification can be directly or indirectly reused with no or little modifications. This can help a lot and save significant time during the verification cycle.

A monitor-based verification approach is an efficient way to make many VE components reusable within the same project and to keep the environment more simple and similar across various levels. One can easily understand the flow, and any newcomer can ramp up his knowledge in a short period of time, and can start working on any module of a complex chip if the approach is common for the entire team in any verification project.

Monitor-based verification approach for complex chips

Functional Verification is one of the biggest challenges for any chip design company even today. The functional verification process has evolved a lot in the last decade in terms of methodology to follow and the availability of a bunch of verification languages. The methodology says how the verification environment looks like, which are the major VE components and how their hierarchy is.

Languages provide a huge set of features that a user can easily utilize and implement the required VE in a reasonably short time. On top of this, the biggest challenges/responsibilities of the verification team is to decide which verification approach is the most suitable for the chip they want to verify. Here, the verification approach is not just adopting a specific methodology but also the effective reusability at various levels of verification during the complete verification cycle.

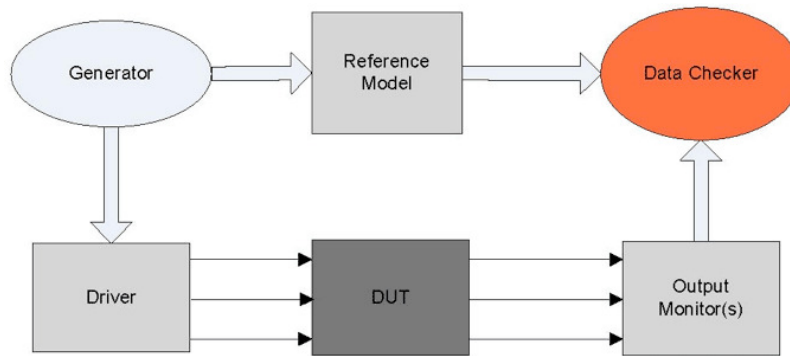
The most common flow of verification is to begin with the sub- block, block, cluster and then, the final full-chip verification. While moving ahead in this sequence, the user has less control on internal sub-blocks. This means we can execute any test scenario (which may be correct or erroneous) easily at the sub-block level, but it is comparatively very difficult to generate the same scenario at the chip level.

The following are the major components for verification of a chip at any level (we may have different names based on methodology):

- **Generator:** Generation of Random, Directed database
- **Driver:** Drivers generated database to DUT pins
- **Monitor:** Monitors DUT buses and provides collected data to Reference Model or Data Checker
- **Reference Model:** Implements cycle accurate model of DUT and used as reference entity for verification
- **Data Checker:** Simply compares Actual vs. Expected transactions

These components can be connected using one of the following approaches.

Generator-based approach:

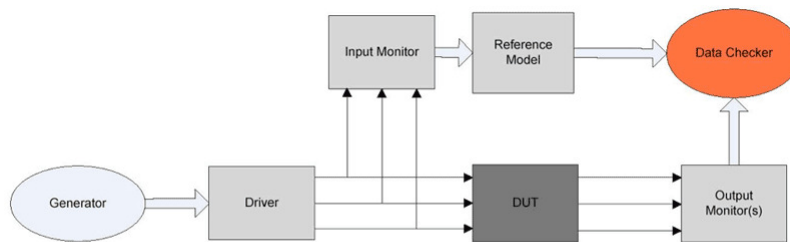


Note: in above figure, generator feeds reference model and driver.

Figure 1: Generator-based approach

[Click on image to enlarge.](#)

Monitor-based approach:



Note: in above figure, generator feeds driver only while input monitor placed on DUT interface pins is feeding reference model.

Figure 2: Monitor-based approach.

[Click on image to enlarge.](#)

Let us first consider that we are verifying this DUT using a generator-based approach. We start with each block verification first. Here, we need the generator at each interface of each block that feeds both drivers at that interface and corresponding transactions to the reference model of that block.

Once we start chip-level verification, we can only make use of generators and drivers for Block-0 and Block-4 because for Block-2, Block-1 becomes the driver and for Block-3, Block-2 is the driver. This means we cannot use reference models for Blocks-2 and Block-3 as it is, as we do not have generators to feed them.

We can make use of these reference models only if we can convert the basic input from the generator of Block-1 into the format which that specific block's reference model can understand. Such conversion may not be that easy to implement from scratch depending on the complexity of each block/chip. The previous block's reference model can be used for this but the limitation is that we may not have those reference models available from the block-level verification team or may have limited features supported in that time.

From the examples above, it can be inferred that we may not have available the reference model of Block-2, so we might not even use reference models of Block-3 or Block-4 which are fully ready. We do not have any VE component ready that can convert IF1 data to IF3 or IF4. If we plug in a partially verified reference model for such conversion, then we may end up with redundant failures and redundant debugging related to those block.

Considering the timeline and simulation time it takes at chip-level verification, this is not an effective solution.

Now consider the monitor-based approach for verification. Here, we do not have any direct dependence on generators used for each block to feed the reference models. We have independent monitors used for this. If we start chip verification in parallel to block-level activities, then we can plug in a reference model of each block as and when it is reasonably verified at the block level.

We can reuse all the input monitors along with reference model from block environment. By reusing the data checkers related to specific block, it is very easy to spot the root cause of any issue/failure that occurs in chip simulations.

Let us imagine that have a real issue in Block-3 in our example. Here we expect Data Checkers (DCs) for Block-1

and Block-2 to work perfectly fine.

DCs for block-3 will shout ERROR here. If we have implemented any interface level assertions in monitors at block-level, we can reuse them as they are in the chip environment. This will help us to catch interface level issues between blocks very quickly. Almost everything including monitor-based functional coverage bins can be reused at chip level. The same reusability can be maintained at system-level simulations too.

The same monitors and reference model usage can allow the block-level teammate/s help in debugging at any level of verification because he/she/they only need to debug his/her/their own debug messages. This helps hold down any issue very effectively within a very short period of time.

Reference model implementation in both the above cases is different. In the case of the generator-based approach, we have straightforward logical blocks that can use the data from the generator and can set expectations. Here, the reference model has all the required information available at time zero from the generator that is enough to set expectations. While in the case of the monitor-based approach, the reference model needs to manage queues for each input transaction on different input ports.

Logical blocks can process these transactions only when some data is available in all such queues which are mandatory to decide what the output should be. Now, as the monitor collects data in a real-time domain, the different input ports of reference model have transactions at different times, as and when they are monitored. The monitor or reference model needs to detect the valid transactions and transactions that needs to be ignored. A good monitor takes care to send only valid transactions to the reference model.

The following figure shows how the final chip environment looks like for a monitor-based approach.

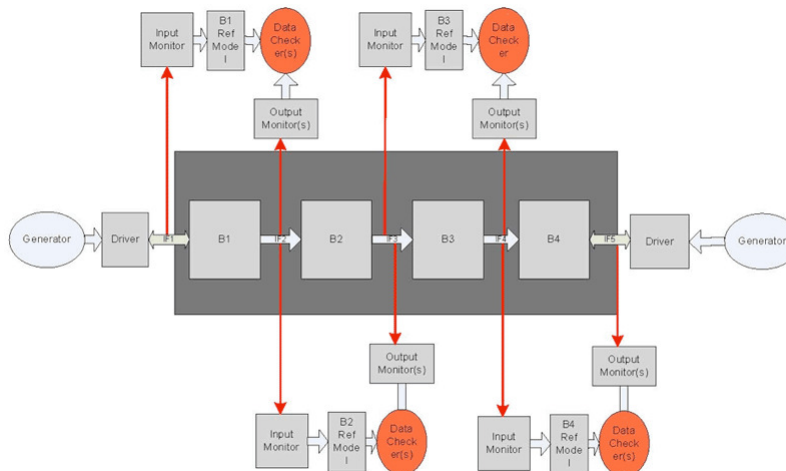


Figure 4: How the final chip environment looks like for Monitor- based approach.
[Click on image to enlarge.](#)

In the above figure, please note that Input Monitor(s), Reference Models, Output Monitor(s) and Data Checker(s) are reused as they are from the block- level verification environment.

Given below is summary of the pros and cons of both approaches.

1) Conventional Generator- based verification approach:

Advantages:

- Reference Model can directly use the high level data transactions from generator and it does not need some specific implementation approach.
- User need not to write monitors on DUT pins (low level data) to prepare Reference model transactions. As and when generator is ready, reference model development can be started and validated.

Disadvantages:

- Generators cannot be reused at chip level or system level simulations and so is the reference models. (Only those generators placed at boundary interfaces can be reused).
- In order to reuse block reference models, we need lots of conversion to feed reference models or need to wait until all reference models are fully verified and ready to reuse.

2) Monitor-Based Verification Approach:

Advantages:

- Block level monitors and reference models are reused as it is at higher (chip/system) level verification.
- As and when reference models for various blocks are available, they can be easily and independently ported at chip level.
- At chip level, if something fails, we can easily detect the issue due to independent reference models and data integrity checking.
- Monitored data can be used to write assertions around any block and can also be used for functional coverage. Same can be reused at all level of verification.
- Verification Environment looks even at each level of verification that allows anybody to easily ramp up and start working on project in reasonably short time.

Disadvantages:

- We need to probe (block) interface pins to monitor data and to prepare high level transactions for reference models. Such high level transaction may directly be available from generator at block level verification.
- Reference model has to play with different queues (based on number of input ports) for further processing this leads to little different way of implementation.

Considering the advantages of the monitor-based approach, I believe that is a better methodology for adoption during the verification of complex chips.

About the Author: Bipin Patel is a member of the Technical Staff (ASIC) at eInfochips Limited, Ahmedabad, India, and can be reached at bipin.patel@einfochips.com.

Please [login or register here](#) to post a comment o
to get an email when other comments are made o
this article