International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 5, May - 2014

VLang and SystemVerilog Verification Languages : A Comparision

Geeta Yadav
Institute of Technology and Management
Gurgaon, Haryana, India

Medha Chhillar, Neeraj Kr. Shukla
Institute of Technology and Management
Gurgaon, Haryana, India

***Abstract*— Verification of the design has become a critical issue due to the increasing complexity of the chip according to the Moore's law. On the other hand in the present scenario the Moore's law no longer persists as no further reduction in the chip size is possible, hence Amdahl's law is the main area of focus. About 70% of the design effort is consumed in the verification, so productivity, reusability and multi-threading in multi-core processors have become the key issues. SystemVerilog and Valng are the languages providing the means of reusability and productivity through the creation of the Verification IP. The edge that VLang has over SystemVerilog is the multi-threading feature which helps in faster simulation and hence more efficient. The VLang uses the D language as its base. The most significant feature is that Vlang is Open Source.**

Keywords—SystemVerilog; Vlang; Open Source; RAL; TLM; multi-core; multithreading; Verilog; VHDL; Vlang; UVM; OVM; VMM; RVM

I. INTRODUCTION

A. Verification

Verification is a process demostrating that the purpose of the design in preserved in its implementation. It provides high quality design after the verification of the SoC. In accordance to the Moore's law the complexity that is the number of transistors on a chip increases every two years on the other hand the Amdhal's law emphasizes on parallel computing as the no. of processor's are increasing the concurrency should also increase to speed up the processing and hence the efficiency, so due to the increasing complexity of the SoC verification has become a critical domain. SystemVerilog and Vlang are the languages used for verification. Parallelism, higher abstraction levels or automation reduces the overall verification time.

Verification languages helps in the creation of the Verification IP which is made using a specific methodology, the most efficient being the UVM with the generic base class libraries. The verification languages build an artificial environment for the design verification and hence the design purpose can be checked whether fulfilled or not. SystemVerilog evolved as the very advanced language for verification with the OOPS concepts being present. Later on came the VLang with the advanced feature of concurrency.

Verification is basically of two types: formal verification and functional verification. Formal verification falls under the equivalence checking and property checking. On the other hand the functional verification shows that the design meets its specifications but does not proves it. This verification follows the three basic approaches like black-box. White-box and grey-box verification. Advantage of black-box is that it does not depend on any specific implementation but the absence of the visibility is a disadvantage. In the white-box the different state combinations can be easily set up whereas the disadvantage being that the changes in the design will require changes in the testbench. The grey-box verification is a combination of the black and the white-box.

B. Multi-threading

In multithreading architectures , a single processor can execute more than one thread at a time. In modern processor's a substantial amount of internal parallelism is present so they can execute out of order instructions, hence to keep the hardware units busy, multithreaded processor's can mix instructions from multiple streams. Nowdays processor's architectures combine the multi-core with multithreading. The use of multithreading hides the high latency of accessing the memory [4]. A multiprocessor consists of multiple hardware processor's, each of them executiong a sequential program. A thread is a software construct. The use of multithreading is the faster execution being possible, there may not be locking mechanism as all the threads run in parallel.

Multi-threading makes the simulation faster and hence more productivity. It is a very efficient technique for increasing the overall systems throughout. By this technique better synchronization and efficient cache sharing can be achieved. One thread will not have to wait for the completion of the other threads for its execution all threads can be simultaneously executed. For verification of the design the multi-threading can be very advantageous, it will basically speed up the simulation of the environment for the design check.

According to the amdahl's law the number of multi-threading process in a processor should also increase as the number of cores in a processor. Hence this process have become a critical issue. Moore's law no longer persists because the chip size cannot be reduced any further so the other solutions have to be found like enabling concurrency.

Hence the significance of the ahmdal's law is increasing and hence the verification most critical domain is coming out to be the enabling of multithreading. The multi-core architectures need to have the multithreading to increase the simulations according to the law.

II. CHARACTERISTICS OF SYSTEMVERILOG AND VLANG

A. *SystemVerilog*

SystemVerilog evolved as a verification langauge with very significant features as compared to the one present in Verilog and VHDL. It had the advanced feature of constraint random stimulus generation, assertions interprocess communication, transaction level modeling, object oriented programming and it also has an advanced scheduler as compared to the Verilog scheduler. This language uses various methodologies for the verification environment creation like UVM, VMM, RVM and OVM.The best suited in the UVM because of its reusability functinality [cris spear]. It brings a higher abstraction level to the design and verification. It has a seperate program block for the testbenches to avoid race conditions. The integration of the new modules to the existing code becomes easy hence, the risks and cost of the new verification language is being reduced [1]. It has the coverage support as well as the RAL support.

The key features of the SystemVerilog are constrained randomization, assertions, easy C model integration, new data types, OOP support, and a very narrow gap between design n verification engineer. It was basically the productivity crises that lead to the development of SystemVerilog [3]. SystemVerilog is not an opensource langauge. The language includes design, testbench, and assertions constructs in it hence testbench can easily access all parts of the environment without the requirement of Application Programming Interface [3]. It has a single constrained randomiser so only one thread can access at a time. SystemVerilog is an extension of the Verilog standard. The main drawback of this language is the absence of multithreading in multi-core processors.

B. *VLang*

Its an opensource verification language available under Boost Software license. It is built with D language as its base so, its a systems programming language. It addresses the critical issue of the multi-core programming. Multiple threads run in parallel in VLang. It is also possible to instantiate multiple simulators and run them in parallel, each having its own uvm_root instance [2]. VLang supports user, heirarchial and language level concurrency. It is compatible with C/C++, SystemVerilog, SystemC, VHDL and MATLAB. Its a transaction of the SystemVerilog UVM version 1.1d.

The constrained random solver in Vlang instantiates many copies of BDD runnng on seperate threads making it lightening fast [2]. It can be easily forked from github and used. It has extreamly faster compilation and simulation. SystemVerilog uses the monolithic testbenches which can have many disadvantages like the probability of the gotchas which are similar becomes more so the identification of the bug becomes difficult, the reuse of verification environment becomes difficult as well as a small incremental change needs the elaboration of the whole testbenche again. The Vlang uses the polylithic testbench which is more advantageous. It has a short learning curve with built in unit testing, no use of pointers makes it even more simpler.

The Vlang uses the Boost license which allows only the sharing of the binary version to the customer. The opensource nature helps in the academic reseacrh and the productivity. It takes the advantage of the D language compiler's feature. The language is ABI compatible with C++.

III. COMPARISION

Serial no.	**Verification languages**		
	Feature	***Vlang***	***SystemVerilog***
1.	Concurrency	Concurrent threads are present	No concurrency
2.	Library support	Supports a standard library	No standard library is present
3.	Compile time	Compile time is faster	Comparatively slow compile time
4.	Memory management	Memory allocation and deallocation	No memory allocation and deallocation
5.	Root entity	Can have multiple root entity	Has only single root entity
6.	Header files	No header files, import of specific library or package	Header files are present
7.	Variables	Variables are static shared for multi-core architectures	Only static not shared
8.	Macros	No macros only mixin or functions	Macros are present
9.	Variable declaration	A variable can be declared anywhere	Variable declared only at the start of the block
10.	Packages	Packages are explicit	Not explicit
11.	Parallelism	Multiple simulators running in parallel	One simulator at a time
12.	Exception handeling	Present	Absent
13.	Reflections	Supported	Not supported
14.	Native compilation	Present	Absent
15.	Testbench	Polylithic	Monolithic
16.	Randomization	Present	Present
17.	RAL	Absent	Present

IV. FUTURE SCOPE

Vlang is a very efficient language in terms of faster simulation and hence more productive. It enables multithreading in multi-core processors thats the key feature. The Vlang does not have a Register Abstraction Layer support to make the verification register based and protocol independent, hence more productive this is under the development process. Transaction Level Modelling 2 support

is not there which is one of the future scope and the coverage is also not present in the current release on github.

V. CONCLUSION

Verification has become a need for the design to enhance the quality of the design and to make sure that the design behaves as intended in the real environment. The reusability, productivity and concurrency are the challenges in this domain . The moore's law no longer holds and its the ahmdal's law which is being emphasized. SystemVerilog as a verification language has overcome the issues of the Verilog and has made the reusability, productivity possible. The use of the UVM methodology has made it possible to build the whole of the reusable verification environment and hence saving of time. The verification environment using UVM is made by the extension of the base classes. On the other hand Vlang is a verification language that brings the concurrency feature which being a drawback of the SystemVerilog. It enables multithreading by the use of the pthreads. Vlang has the multi-core feature as well hence satisfies the ahmdal's law. So, compared to SystemVerilog Vlang has much faster simulation and enables mutithreading. It also has mostly all the features present in SystemVerilog.

REFERENCES

[1] URL: http://www.systemverilog.in/overview.php
[2] URL: http://verlang.org
[3] URL: http://link.springer.com/book/10.1007/b138536
[4] URL: http://horstmann.com/corejava/cj7v2ch1.pdf