Comparative Study of Hardware Languages and Programming Languages

Mrs. A. N. Dubey[1] Mrs. P. R. Autade[2]
[1,2]JSPM RSCOE Polytechnic IInd shift Tathawade, Pune, Maharashtra, India

Abstract— Comparison of programming languages is a common topic of discussion for engineers but comparison of hardware languages and programming languages is very rare or not found easily. In this paper we present a comparative study between hardware languages such as VHDL, Verilog and SystemC and programming languages such as C,C++ and Java languages; These languages are compared under the characteristics of syntax of code, evolution reliability, tools, readability, efficiency, familiarity and applications used in Industry for future carrier with terms differentiation of various languages.

Keywords: Hardware languages, HDL, Programming languages, C, Code, VHDL, Verilog

I. INTRODUCTION

The complexioness of integrated circuits increases every year, so reduce complexities invent Hardware Description Lanauages in1984 by Phil Moorby and Prabhu Goel around 1984. Implementing algorithms using hardware languages like VHDL or Verilog. The first level programming language was Plankalkül, created by Konrad Zuse between 1942 and 1945. The main and important comparison between HDL and Programming Language is that HDL describes the behavior of digital systems while Software Language provides a set of instructions for the CPU to perform a particular task. Programming languages are extremely interesting. Computer scientists tend to invent new programming languages. Thousand different languages have been created within previous few years. Some languages have large popularity and most commonly used in software world. and others creating new features and varieties in their languages. Each language has its pros and cons. At a recent time, the applications of software have influences in daily use.Otherhand hardware engineers this has resulted in a higher degree of design automation and increase in the number of tools available to an IC designer. Recently there has been an willingly toward the usage of Hardware Description Languages. These above all languages are compared under the characteristics of reliability, portability of tools, readability, efficiency, familiarity and expressiveness with terms differentiation with giving simple program code and syntax.

II. HISTORY

A. *History of Programming Languages*

Computer programming languages is the process of writing, testing, debugging and maintraing the source code of software programmer. History of programming languages describe in table no.

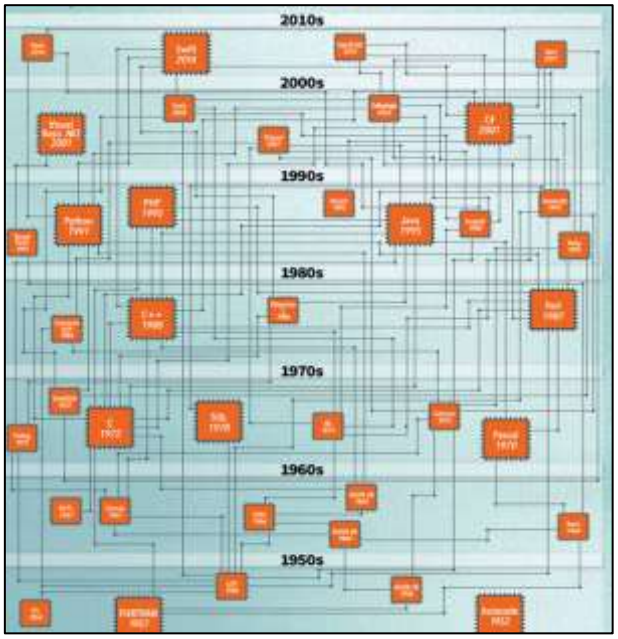

Fig. 1: History of Programming Languages

B. *History of Hardware Languages-*

Sr.no.	Year	Languages	Description
1	1972	FHL	Mostly designed for simulation, formal verification, and netlist generation
2	1983	VHDL	Developed under the VHSIC program of the U.S. Department of Defense. It was originally intended to serve as a language to document descriptions of complex digital circuits. The first modern HDL
3	1987	VHDL	IEEE adopted the VHDL language as standard 1076 (also referred to, as VHDL-87). It was revised in 1993 as the standard VHDL-93.
4	1984	Verilog	Invented by Phil Moorby and Prabhu Goel around .It

Table 1: History of Hardware Languages

Why so many programming languages? Every language is a tradeoff among completion world reaction to perceived failings of others; personal taste Notation is important – "Language shapes the way we think and determines what we can think about." Benjamin Whorf – the more natural and close to the problem domain, the easier it is to get the machine to do what you want .Higher-level languages hide differences between machines and between operating systems .We can define idealized "machines" or capabilities and have a

program simulate them -"virtual machines" – programming languages.

Generally a comparison between programming languages and hardware descriptions languages is based on the number of lines of code and execution time required to achieve a specific task, using the two languages. A number of additional parameters can be observed, such as features, existence or absence of constructs that facilitate coding, availability of optimization techniques, as well as others. These criteria vary slightly when attempting to compare two HDLs. For instance, HDLs need to have time-handling constructs, unlike most other computer languages. Comparable "building blocks" may synthesize into different circuitry, depending on the language's standard. Other points utilized as a basis for comparison include: efficiency of methods and language constructs, description, syntax, applications and used in Industry for future carrier and ease of implementation.

III. Conceptual Background

A. Fundamental differences in constructs:

Firstly describe the differences between diiferen hardware descrption language with its syntax of code with example VHDL is more verbose than Verilog and it is also has a non-C like syntax. With VHDL, you have a higher chance of writing more lines of code. Verilog has a better grasp on hardware modelling, but has a lower level of programming constructs. Verilog is not as verbose as VHDL so that's why it's more compact.

Example of simple VHDL code -

```
library ieee;
use ieee.std_logic_1164.all;
entity AND_gate is
port(       x: in std_logic;
            y: in std_logic;
            F: out std_logic
);
end AND_gate;
architecture behav of AND_gate is
begin
   process(x, y)
   begin
      -- truth table
      if ((x='1') and (y='1')) then
            F <= '1';
         else
            F <= '0';
         end if;
   end process;
end behav;
```

Example of simple Verilog code :

```
module andgate (a, b, y);
input a, b;
output y;
assign y = a & b;
endmodule
```

Example of simple SystemC code :

```
//file for and_gate.h
#include "systemc.h"
SC_MODULE (and_gate)
 {
  sc_in<SC_bit>a;
  sc_in<SC_bit>b;
  sc_out<SC_bit>c;
void  prc_and_gate() //process
{
   C=b&a;
}
SC CT_AND(and_gate)
{
  SC_METHOD (prc_and_gate();
  Sensitive <<a<<b;
}
};
```

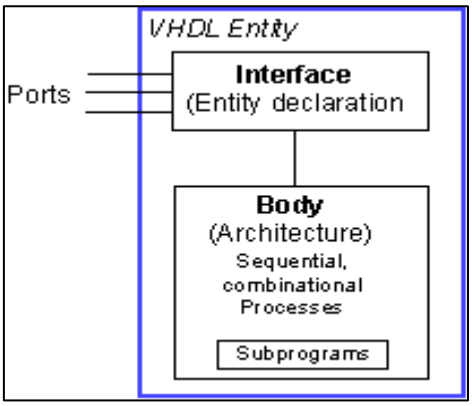

Fig. 2: History of Programming Languages-

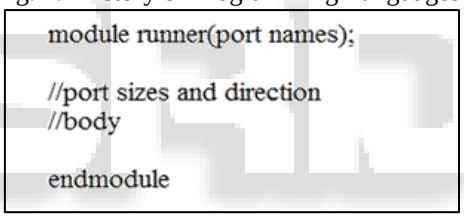

Fig. 3: History of Programming Languages-

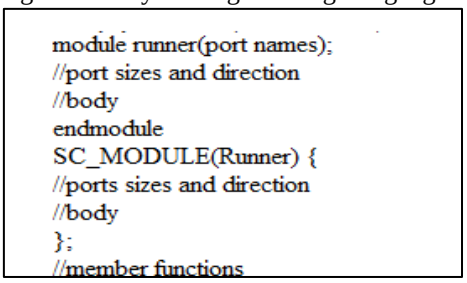

Fig. 4: History of Programming Languages-

IV. Description of Constructs

VHDL consists of a design entity that can contain other entities that are then considered components of the top-level entity. Each entity is modeled by an entity declaration and an architecture body. One can consider the entity declaration as the interface to the outside world that defines the input and output signals, while the architecture body contains the description of the entity and is composed of interconnected entities, processes and components, all operating concurrently, as schematically shown in Fig.3 above. In a typical design there will be many such entities connected together to perform the desired function.

Verilog structures which build the hierarchy are:

i. modules ii. Ports
A module is the basic unit of the model, and it may be composed of instances of other modules .Module name an identifier that uniquely names the module. . Port list a list of input, input and output ports which are referenced in other modules. Declares section specifies data objects as registers, memories and wires as well as procedural constructs such as functions and tasks.
SystemC structures which build the hierarchy are-
SystemC accomplishes its own can declare a function separately from its body, as in C. This property of the language can also be viewed as an advantage, since a module can therefore call several different processes. The functions are written as members of the module class being designed, allowing the designer to more easily integrate additional functionality in the same design.

1) Modeling Styles in VHDL & Verilog- It is a programming language used to model a digital system by dataflow, behavioral and structural style of modeling. Structural modeling (Gate-level) -Use gates. ,,
2) Dataflow modeling - Use assignment statements ,,
3) Behavioral modeling -Use procedural assignment statements.

Modeling Styles in SystemC-

1) System-level modeling(ESL) design
2) Transaction-Level Modeling (TLM).

Name Of HDL	Hardware Requirements	Software Requirements
VHDL	FPGA Kit	Xilinx ISE, Quartus II from Intel, Modelsim
Verilog	FPGA Kit	xilinxISE,Quartus II from Intel, Modelsim
SystemC	FPGA Kit	Powersim, Verilator

Table 2: Hardware requirement tools

Sr no.	VHDL	Verilog	SystemC
1	Beginner designers may want to start with VHDL	Beginner designers may want to start with Verilog	Difficult for Beginner designers may want to start with SystemC.
2	More rich and strongly typed language, deterministic and more verbose than Verilog	Weakly-typed smaller vocabulary(Less verboseness)	Strongly typed but execution time more.
3	Better suited for synthesized gates.	better suited for structural designs,	better suited for behavioral
4	Easy to synthesize than VHDL	difficult to synthesize than VHDL	more difficult to synthesize than Verilog
5	Not case-sensitive	Verilog is case-sensitive	case-sensitive
`6	No C-like syntax.	More C-like syntax.	More C-like syntax
7	High-level descriptions are not closer to actual hardware	Low-level descriptions are closer to actual hardware.	Very less than Verilog and Vhdl
8	FPGA designers prefer to use VHDL	ASIC designers prefer to go with using Verilog	FPGA
9	Widely used in Europe, rest of the world.	Widely used in North America, Japan.	Widely used in Large scale industry semiconductor devices.
10	Easy to debug	Easy to code	Easy to code but difficult to simulation
11	Origin From Ada	Origin From C	Origin From C/C++

Table 3: Differences with VHDL, Verilog, System C

The layout of these programs are very similar; you can reasonably. The VHDL is longer than the Verilog.
Fundamental differences in constructs for programming languages:

1) Hello World Program in C:

```
#include stdio.h>
void main ()
{
        printf("Hello World!");
        getch();
}
```

2) Hello World Program in C++:

```
#include <iostream>
int main()
{
 std::cout << "Hello World!";
}
```

3) Hello World Program in Java:

```
Class Helloworld
{
        Public static void main(String args[])
        {
                System.ot.println("Hello World!");
        }
}
```

When you run the programs, the output will be:
Hello World!
JAVA is Object-Oriented while C is procedural. Most differences between the features of the languages arise due to the use of different programming paradigms. C breaks down functions while JAVA breaks down to Objects. C is more procedure-oriented while JAVA is data-oriented. C++ is a superset of C, Java is neither a superset nor a subset of C or C++.C, C++, and Java are the most popular programming languages used today at a broad level. They have a pretty similar syntax for basic concepts. Most of the basic constructs like if statements, loops, function syntax, switch case statements and concepts like recursion are still valid. Many

other concepts like the syntax for comments, and the idea of static class variables, also held in both Java and C++.

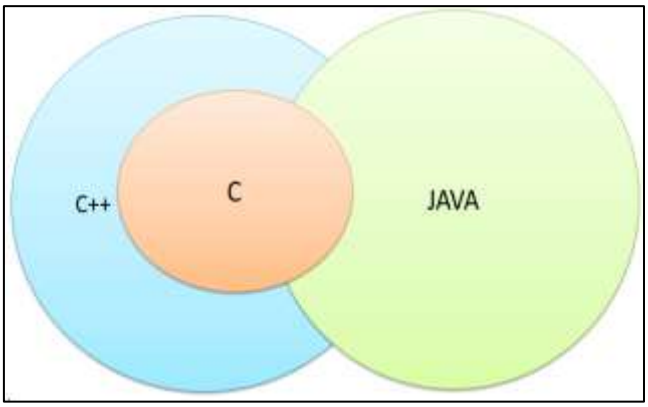

Fig. 5: Overview of C, C++&Java

The main purpose of this paper learning a basic differences between programming languages and hardware languages to become a better theoretically, conceptually clear for lower level study or higher level of study to become a good programmer i.e. to become more effective at theoretically and conceptually with better understanding.

V. Study of Difference between Hardware Languages &Programming Languages

Both are computer languages but used for hardware design and design. A hardware description language looks very similar to a programming language like C; it is a textual description consisting of expressions, statements and control structures. One main important difference between most programming languages and HDLs is that HDLs explicitly include the notion of time.

Hardware language specialized computer language used to describe the structure and behavior of electronic circuits, mostly for difgital logic circuits. Software / programming language is a computer language used to write a set of instructions to allow to CPU to perform specific task .Not as complex than HDL languages also help in developing various applications such as games, software, app, webpage.

HDL supports expressing things like parallel operation, waiting for clock edges, tri-state logic, hardware propagation delays, and hierarchical structure in ways that make sense for hardware. Programming languages typically specify sequential operations of an abstract machine. HDL let you specify specific, concrete machines.

For Example:

```
//Statements
C = A * B;
A = B * C;
```

It would mean "first calculate A times B and then put the result into C; then calculate B * C and put the result into A". If the starting values of A, B and C are 3, 2 and 1, then the ending values would be C = 6 and A = 12.

A similar expression in a HDL might mean "always be calculating A * B and B * C. On a clock tick, put the result of A * B into C, and the result of B * C into A". If the starting values of A, B and C are 1, 2 and 3, then the ending values would be C = 6 and A = 2.

VI. Conclusions

This paper isn't planned as an entire characterization of all hardware and programming languages. There are several other attributes of languages besides strength of writing and the level of programming, and there are many many languages that can't be feature cleanly as a hardware description & programming language. Learner designers/students might want to start with Verilog it has a far smaller vocabulary, and doesn't need previous knowledge of another language &advantages over VHDL language. It also includes a smaller amount of task-specific constructs to be remembered. Whereas Verilog is also be considered a weak object oriented language, SystemC is more suited for such programming style, because of its roots.

Generally Verilog is best suited to structural designs, because it permits for better control of modules within the same abstraction layers, even though it lacks component hierarchy management. SystemC's nature is behavioral, which can make it more difficult to synthesize than Verilog and VHDL. Hardware and programming languages have many various syntax format, coding, execution time ,application .These each types have different applications categorized into hardware designer and software designers for different purpose ,so their role are totally different in practical oriented. However each languages are widespread in their industry area.

Depending upon application we can used in real life .Also scope of these languages industry are very high level in all over the world. In future VLSI professionals are always in high demand in the fast-changing chip designing industry and R&D, and for programmer Data Analytics, Big Data, IOT, iOs design, android and UX are popular in the market. It would be wise to shift your career into one of the above as these are going to be the most lucrative and interesting domains in industry or R&D. In future this paper will be include stastical comparison between languages also describes other features.

Acknowledgment

It is the great pleasure that we acknowledged the enormous assistance and excellent of this technology, extended to use of in various applications. And also guidance of co –Author Mrs.A.N.Dubey (HOD of E&TC).

References

[1] IEEE. IEEE Standard VHDL Language Reference Manual: IEEE Std 1076-1993. IEEE Press, 1993.
[2] Shiva, Sajjan G. "Computer hardware description languages—A tutorial." Proceedings of the IEEE 67.12 (1979): 1605-1615.
[3] Wirth, Niklaus. "A brief history of software engineering." IEEE Annals of the History of Computing 30.3 (2008): 32-39.
[4] IEEE Design Automation Sub-Committee. "IEEE Standard Hardware Description Language Based on the Verilog Hardware Description Language." (1996)
[5] Martin, Grant. "SystemC and the future of design languages: Opportunities for users and research." 16th

Symposium on Integrated Circuits and Systems Design, 2003. SBCCI 2003. Proceedings.. IEEE, 2003.
[6] Wexelblat, Richard L., ed. History of programming languages. Academic Press, 2014.
[7] MacLennan, Bruce J. Values and Objects in Programming Languages. No. NPS52-81-006. NAVAL POSTGRADUATE SCHOOL MONTEREY CA, 1981.
[8] Prechelt, Lutz. "Comparing Java vs. C/C++ Efficiency Differences to Interpersonal Differences." Commun. ACM 42.10 (1999): 109-112.
[9] Moreira, José Eduardo, Samuel P. Midkiff, and Manish Gupta. "A comparison of Java, C/C++, and FORTRAN for numerical computing." IEEE Antennas and Propagation Magazine 40.5 (1998): 102-105.
[10] Nanz, Sebastian, and Carlo A. Furia. "A comparative study of programming languages in rosetta code." 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering. Vol. 1. IEEE, 2015.
[11] Su, Stephen YH. "A survey of computer hardware description languages in the USA." Computer 7.12 (1974): 45-51.

