An Introduction to VHDL

VHDL is an acronym for VHSIC (Very High-Speed Integrated Circuit) Hardware Description Language which is a programming language that describes a logic circuit by function, data flow behavior, and/or structure. This hardware description is used to configure a programmable logic device (PLD), such as a field programmable gate array (FPGA), with a custom logic design.

The general format of a VHDL program is built around the concept of BLOCKS which are the basic building units of a VHDL design. Within these design blocks a logic circuit of function can be easily described.

A VHDL design begins with an ENTITY block that describes the interface for the design. The interface defines the input and output logic signals of the circuit being designed. The ARCHITECTURE block describes the internal operation of the design. Within these blocks are numerous other functional blocks used to build the design elements of the logic circuit being created.

After the design is created, it can be simulated and synthesized to check its logical operation. SIMULATION is a bare bones type of test to see if the basic logic works according to design and concept. SYNTHESIS allows timing factors and other influences of actual field programmable gate array (FPGA) devices to effect the simulation thereby doing a more thorough type of check before the design is committed to the FPGA or similar device.

VHDL Application

VHDL is used mainly for the development of Application Specific Integrated Circuits (ASICs). Tools for the automatic transformation of VHDL code into a gate-level net list were developed already at an early point of time. This transformation is called synthesis and is an integral part of current design flows.

For the use with Field Programmable Gate Arrays (FPGAs) several problems exist. In the first step, Boolean equations are derived from the VHDL description, no matter, whether an ASIC or a FPGA is the target technology. But now, this Boolean code has to be partitioned into the configurable logic blocks (CLB) of the FPGA. This is more difficult than the mapping onto an ASIC library. Another big problem is the routing of the CLBs as the available resources for interconnections are the bottleneck of current FPGAs.

While synthesis tools cope pretty well with complex designs, they obtain usually only suboptimal results. Therefore, VHDL is hardly used for the design of low complexity Programmable Logic Devices (PLDs).

VHDL can be applied to model system behavior independently from the target technology. This is either useful to provide standard solutions, e.g., for micro controllers, error correction (de-)coders, etc., or behavioral models of microprocessors and RAM devices are used to simulate a new device in its target environment.

An ongoing field of research is the hardware/software design. The most interesting question is which part of the system should be implemented in software and which part in hardware. The decisive constraints are the costs and the resulting performance.

History of VHDL

VHDL was developed in the early 1980s as a spin-off of a high-speed integrated circuit research project funded by the U.S. Department of Defense. During the VHSIC program, researchers were confronted with the daunting task of describing circuits of enormous scale (for their time) and of managing very large circuit design problems that involved multiple teams of engineers. With only gate-level design tools available, it soon became clear that better, more structured design methods and tools would need to be developed.

To meet this challenge, a team of engineers from three companies -- IBM, Texas Instruments and Intermetrics -- were contracted by the Department of Defense to complete the specification and implementation of a new, language-based design description method. The first publicly available version of VHDL, version 7.2, was made available in 1985.

In 1986, the IEEE was presented with a proposal to standardize the language, which it did in 1987 after substantial enhancements and modifications were made by a team of commercial, government and academic representatives. The resulting standard, IEEE 1076-1987, is the basis for virtually every simulation and synthesis product sold today. An enhanced and updated version of the language, IEEE 1076-1993, was released in 1994, and VHDL tool vendors have been responding by adding these new language features to their products.

Although IEEE standard 1076 defines the complete VHDL language, there are aspects of the language that make it difficult to write completely portable design descriptions (descriptions that can be simulated identically using different vendors' tools). The problems stem from the fact that VHDL supports many abstract data types, but does not directly address the problem of characterizing different signal strengths or commonly used simulation conditions such as unknowns and high-impedance. Soon after IEEE 1076-1987 was adopted, simulator companies began enhancing VHDL with new signal types (typically through the use of syntactically legal, but non-standard enumerated types) to allow their customers to accurately simulate complex electronic circuits. This causes problems because design descriptions entered using one simulator were often incompatible with other simulation environments. VHDL was quickly becoming a non-standard.

To get around this problem of non-standard data types, another standard was developed by committee and adopted by the IEEE. This standard, numbered 1164, defines a standard package (a VHDL feature that allows commonly used declarations to he collected into an external library) containing definitions for a standard nine-valued data type. This standard data type is called standard logic, and the IEEE 1164 package is often referred to as the standard logic package, or sometimes MVL9 (for multivalued logic, nine values).

The IEEE 1076-1987 and IEEE 1164 standards together form the complete VHDL standard in widest use today. (IEEE 1076-1993 is slowly working its way into the VHDL mainstream, but does not add significant new features for synthesis users).

The VITAL initiative (VHDL Initiative Toward ASIC Libraries) is an effort to enhance VHDL's abilities for modelling timing in ASIC and FPGA design environments. VITAL borrows liberally from existing methods for timing annotation used in Verilog HDL. Specifically, the VITAL standard (standard 1076.4, which as of this writing is in balloting phase) describes a method for annotating delay information using the same underlying tabular format as specified in Verilog. The adoptance of this

standard will make it much easier for ASIC and FPGA vendors to create timing-annotated netlists and other data describing the detailed behavior of their devices.

Introduction to Xilinx ISE

Xilinx ISE (Integrated Synthesis Environment) is a software tool produced by Xilinx for synthesis and analysis of HDL designs. It a design environment for FPGA (Field Programmable Gate Array) products from Xilinx, and is tightly-coupled to the architecture of such chips, and cannot be used with FPGA products from other vendors. It is primarily used for circuit synthesis and design, while ISIM or the ModelSim logic simulator is used for system-level testing. Since 2012, Xilinx ISE has been discontinued in favor of Vivado Design Suite, that serves the same roles as ISE with additional features for system on a chip development.