COMPUTER-AIDED DESIGN AND HARDWARE DESCRIPTION LANGUAGES

Zinchenko Y. Y.
Computer Science Department, Donetsk State Technical University (DonSTU)
zinchenko@cs.dgtu.donetsk.ua

Abstract

Zinchenko Y. Y. Computer-aided design and hardware description languages.
The directions of computer-aided design are considered. The hardware description languages (HDL) as a basis of CAD are considered, among which the special notice is given to the language VHDL. The systems on the basis of VHDL are described. The problems of VHDL and systems on its basis are defined.

Introduction

It is obvious, that the further development of computing systems is impossible without usage of computer-aided design systems (CAD). The top-down design systems are in this connection a singular urgency, as they create a possibility of implementation of an old dream of the developers - complete automation of all design stages.

This article is devoted to the analysis of a current state of computer-aided design.

At first two directions of computer-aided design - bottom-up and top-down design are considered, the disadvantages of the first direction and advantage of the second one are emphasized. Hardware description languages (HDL) as a basis of CAD further are parsed, among which the special attention is paid to the language VHDL. The systems on a basis of VHDL are considered. The problems of VHDL and systems on its basis are defined, that allows to select directions and to determine the tasks on their further improvement.

Bottom-up and top-down design of computing systems

Today a computer-aided design is a basic problem of computer industry development. Two basic ways of computer system design are known [1]:

- bottom-up design and
- top-down design.

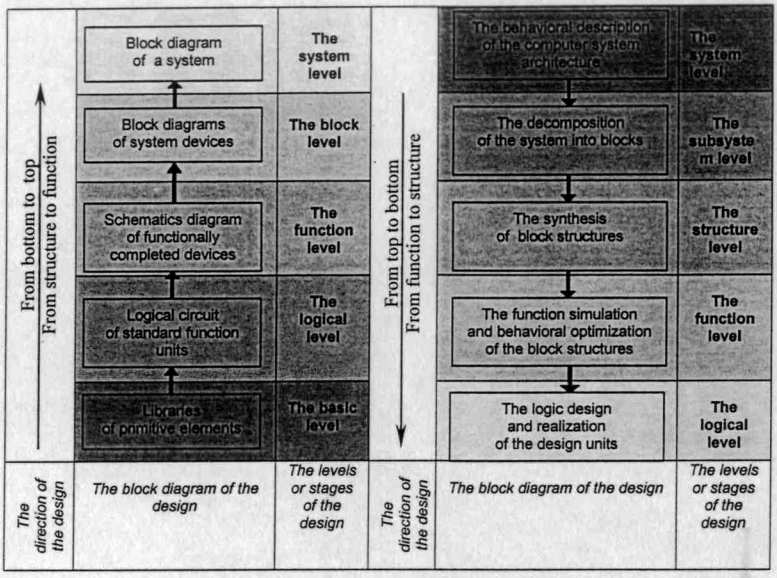
The bottom-up design is a traditional way of design. The design in this case is represented by sequentially executed stages or levels. The stages or levels are executed in the direction from construction from simple to more complicated elements of design. Each next stage or level of the design is executed on the base of previous levels.

The bottom-up design can be presented by the Figure 1,a.

At the first stage the primitive indivisible elements of the design are developed. The *logical elements* are united in libraries of elements. It is basis of the design. This level is called *basic*.

At the second *(logical) level* on the basis of logical elements the *logical circuits* of the design elements of a higher rank, i.e. of the standard function units are created: triggers, registers, counters, adders etc.

Further from standard function units *the schematic diagrams* of functionally completed devices of the design are developed: RAM, ROM, arithmetic logical unit (ALU) etc. It is a function level.



a *b*

Figure 1. The process of bottom-up (a) and top-down (b) design

At the last levels the block diagrams of the design devices and designed system as a whole are developed from the elements designed at the previous stages. These levels are called *block and system levels* correspondingly.

After each level *the check for design errors* is performed. This procedure is called *verification*. If there are errors at some level, then there is often a necessity of modification of all circuit fragments developed at the underlying levels.

It is obvious from the figure that **the main attention by a bottom-up design is paid rather to a structure of a designed system, then to its functions.** It is also possible to say that **the process of design is directed from a structure to functions of a designed system.** Therefore it is possible to call a bottom-up design **"from structure to functions"**.

The bottom-up design of modern computer systems appears *inefficient* due to the following reasons:

- **high labor-consuming factor** of design, in particular for LSI and VLSI;
- **a low degree of automation** - only separate stages of the design are automated, but not the design as a whole;
- **large number of manual operations** by the circuit entry;
- **complexity of coordination** among the large number of designers;
- **low flexibility** (or adaptability) - the errors detected at a certain level of design can result in the necessity of modification of all previous levels.

The methodology of *top-down design* is an alternative to a bottom-up design.
In this case the process of design is also represented as a hierarchy of sequentially executed stages or levels. However, the direction of the process is reversed and the essence of the

executed stages is changed. **The design is performed not "from structure to structure", but from the given functions of a system to a structure**, which realizes these functions.

The process of a top-down design is shown in the figure 1,b.

At first a designer *describes architecture* of a computer system with the help of behavioral or functional models reflecting functional and timing data of a system.
It is a *system level* of the design.

Further on a *subsystem level* the *decomposition* of a system, i.e. a partitioning of a system into integrated blocks connected with each other with the help of buses and wires is executed.

The *block functions* are assigned to blocks such as logic and arithmetical formulas, command lists, transition graphs of finite state automaton and so on can be.

Then on the base of behavioral description of blocks the *synthesis* of their structures is performed.

At the next stage the *simulation and behavioral optimization* are done, as a result of which the structure of a system and specifications of each standard function unit are determined.

At the last stage the *logic design* of standard function units and their realizations on the selected basis are executed.

Each stage of a top-down design is also accompanied by procedures of *verification*, however, in contrast to a bottom-up design the return from the next level in case of detection is done only on a higher level.

Thus, by **top-down design the main attention is paid to behavioral, but not to a structural representation** [1].

The top-down design in comparison to the bottom-up design has the following *advantages*:

- **Simplicity of the design description** – it is enough to set the realizable functions of the design;
- **High flexibility** – for a modification of the design it is enough to change output functions;
- For detection of errors at some level it is enough to correct errors only of previous level;
- **Possibility of full automation** of the design as a whole, not only on individual stages.

The last advantage of a top-down design is the most valuable. It creates the prerequisite for the realization of an old *dream of designers* – full automation of the design, including automatic synthesis of structure of complicated devices by their functional description. For the realization of such possibility a *medium of a top-down design should include:*

- **Behavioral description** of a designed system;
- **Methods and tools of decomposition** of a behavioral description of a system;
- **Set of behavioral and or structural descriptions** of standard functional components, which correspond to the standard elements of hardware;
- **Rules of choosing description form of a component** for hardware realization.

The carried out analysis of the top-down and bottom-up design methodologies allows making a choice in favor of the last. First of all, it is explained by the possibility of full design automation, that result in the sharp technical progress and as a consequence, rise of economic indexes of the design.

The specialized (problem-oriented) *hardware description language* and *CAD* basing on it are necessary for automation of a top-down design.

HDL - Hardware Description Languages

An *Old dream* of the designers of computing systems was the creation of hardware description languages (HDLs), which would enable to automate all stages of a top-down design.

HDLs are known for a long time. One of the first successful HDLs was *LOTIS* - formal language describing logical combinational and a sequential circuits. IBM developed LOTIS in 1964.

In 70's in Europe and USA were designed *more than two hundred HDLs.* Most known are МОДИС, AUTOCODE, МПЛ, ОСС-2, FOROS, EPICURE, CDL, DDL, ISPS, CONLAN, HILO, CASCADE, REGLAN, ADA etc.[2].

HDLs were for a long time the *property* of those small teams, which created them. They used them for the design of specialized electronic systems.

On the first attempt to make the order in this question was a decision of HDL conference about creation of the *CONLAN* language. This work began in 1976 and was successfully completed by creation of the foundation for "*third HDL generation*". Main problem on this way was the support of the *new parallel design technology* [3,4].

For support of interaction between the various designers it was necessary to develop *standards* of the design models, protocols and data formats. The language CONLAN successfull solved this task. After success of the language CONLAN the *Department of Defense* of USA decided to attend to special attention a HDL. In 1993 the Department of Defense finance the design of *VHDL*[5].

The language VHDL is a description language of hardware including very high-speed integrated circuits (*VHSIC*).

At first VHDL was used for support of the interface between the various designers of VHSIC. However designers of the language have collected and realized the *recommendations* of many prominent computer engineers and scientists. This resulted in the creation of a very efficient and comfortable HDL.

In August, 1985 the *version 7.2* of the language VHDL was issued. After the appearance of the version 7.2 IEEE became the sponsor of the development of VHDL. In December, 1987 IEEE officially ratified the new version as the standard IEEE *1076* VHDL [3]. IEEE updates standards every five – six years. " *IEEE 1076-1993* " it the last standard VHDL, which was published in 1993 [4,5].

Now, after the standards on VHDL were developed, any design, documenting and algorithms of hardware **should have the description in the language VHDL.** The VHDL-description is the integral part of documentation for the designers and users of hardware.

Except for VHDL the greatest popularity among HDLs was acquired by the following languages: HHDL, ISP, UDL/I and developed on a basis of C the language Verylog. The language *Verylog* has is the most popular among hardware description languages after VHDL. However VHDL is the **most efficient** and **most used** hardware description language. This language became a standard for a systematic designing.

Now there is a huge interest to VHDL. In the USA and Europe *the American and European groups,* which apply VHDL in industry, are created. Except for the Department of Defense of the USA the sponsors of works on VHDL development are also *six large companies. In Russia* six large scientific centers and educational organizations use VHDL. *In Ukraine* over 16 such organizations use VHDL [6].

Feature of VHDL and CADs based on VHDL

Principles, on which the language VHDL is founded, and the properties, which it has, can be circumscribed as follows [2,5,7,8,9].

1. Possibility of the **design description at a behavioral level** with the help of two forms:
   - *streaming form* – it imitates behavior of the design as a sequence of the dataflow;
   - *processing form* - it imitates behavior of the design as a collection of independent but synchronized processes;
2. Combination of **behavioral and structural methods of the design description**. The design at the structural level is imitated by a collection of components of structure and links between them. Any of the above mentioned behavioral forms can be selected for the description of the design components.
3. **The arbitrary quantity of levels** of nesting of the design model, that allows to describe computer system of any degree of complexity.
4. The possibility of **detection** by the simulation **of glitches and undefined** states on outputs of the circuits and
5. **simulation of computer system with the bus interface** using of the **principle of multivalued logic.**
6. **Real time imitation,**
7. **parallel process synchronization** and
8. **parallel process imitation** in a uniprocessor simulating computer using **event-driven simulation;**
9. **Usage of various models of signal propagation delays** (slugged, transport etc.). This allows to parse the stability of a circuit from high-frequency noise on its inputs and to play back a "pure" propagation delay of the circuit signals;
10. The possibility of **parallel development of the project** at various hierarchy levels, which essentially reduces the time of designing.

The modern computer-aided design systems (CAD or CADS) are created on the basis of HDL. Such systems include a subsystem of simulation, that allows to solve successfully the *following tasks:*

- Design and debugging of work **algorithms** of complex computing systems;
- Debugging on model **of software and microcode software** of the system;
- **The debugging of a model** of the designed block together with a model of that equipment, in the structure of which the given block will be used;
- **The verification** - check of service capability of the designed system by simulation. *This allows to eliminate a difficult stage of the design breadboarding;*
- Realization of the design model written in HDL as a **library unit** of CADS and use of such library units for designing more complex systems;
- **Automatic test generation;**
- **Documenting of the design.**

Many among such systems are based on VHDL.

Now there are many systems of computer-aided design, based on VHDL. Some of them are *industrial*, others are *university*.

Among *industrial* CADS the following systems are the most known: ORCAD, VERIBEST, GALILEO, MAX+PLUS, COMPASS, CADENCE, TANNER, SMASH, MENTOR GRAPHICS, WORKVIEW OFFICE, ACTIVE-CAD and ACTIVE-HDL.

Among *university* CADS the following systems are the most known: ALLIANCE, AMICAL, VANTAGE, SYNOPSYS, Educational versions of ACTIVE-HDL.

CADS are also divided into two groups:

- *universal and*

- *specialized.*

The *universal* systems are intended for designing different objects. Most of CADs are universal. However, any hardware digital language, including VHDL, cannot completely satisfy the developer. Therefore there are systems, which supplement and expand possibilities of VHDL in specific areas of design. Such CADs are named *specialized.* For example, there are specialized CADs basing on the division of a design objects into *controlling and operational parts* (CASCADE). The Silicon 1076 system is oriented on designing specialized VLSI circuits according to the principle *of parallel design.* This system has the *synthesizer of register transmissions*, specialized resources for the *synthesis of memory and arithmetic devices* [2].

Now one of the most popular and most efficient CAD on the basis of VHDL is the *ACTIVE-HDL* system, which was designed by ALDEC company [10] on the basis of its predecessor ACTIVE-CAD. The last version of this CADS is the version 3.5, which was produced in June 1999 [11,12]. The structure of *ACTIVE-HDL* 3.5 is shown in figure 2.

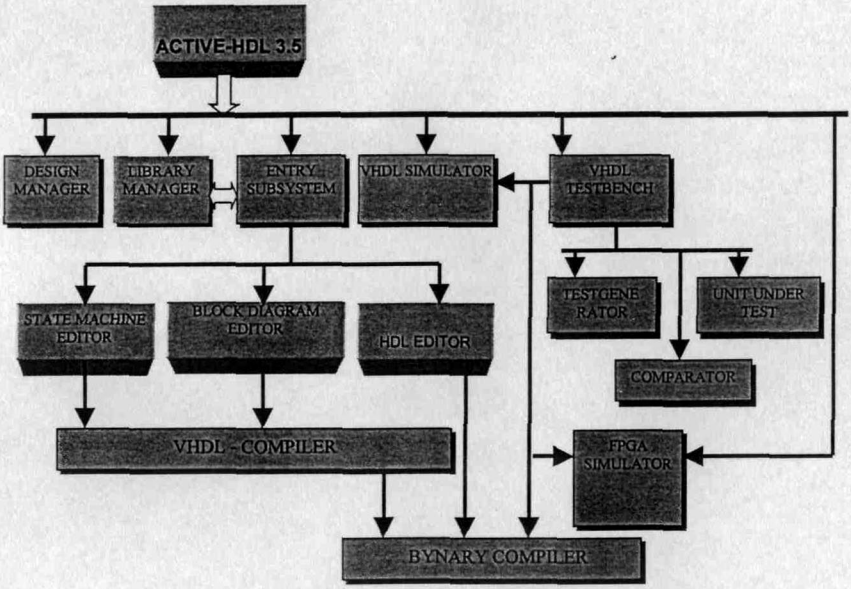


Figure 2. The structure of ACTIVE-HDL 3.5

Nowadays ALDEC company introduces its developments into educational process of Russian and Ukrainian universities. Donetsk State Technical University also cooperates with the ALDEC company.

Problems of VHDL

1. **Problem of adequacy** (correspondence of the circuit and its VHDL-description), stipulated by multiple-alternative and combinatorial complexity of solution of this task [2].
2. **Problem of automatic synthesis** [1].

To enable the possibility of automatic synthesis the following assumption should be fulfilled:

- The operation of the generated circuits is adequate to the VHDL-descriptions;
- The choice of optimum variant of VHDL-description of functions for each hardware implementation by given criterion of optimality of the circuit synthesis is provided;
- For efficient hardware solutions it is necessary to take into account as early as possible the technological information on the design

3. **Problem of development of diagnostic support.**
Here it is possible to allocate two aspects of this problem:
- *Adequacy* - the diagnostic support, designed on the basis of the behavioral description of the diagnostic object does not adequately reflect actual diagnostic processes.
- *The uniform approach.* Development of diagnostic support is reduced, as a rule, to the design of special software.

Conclusion

The analysis of main directions of computer-aided design, hardware description languages and CADs on their basis Fulfilled in this article allows to make the following conclusions:

- The promising direction of the computer-aided design is the top-down design, because it basically allows to solve the problem of complete design automation;
- Among the hardware description languages the VHDL and VERYLOG have received the generally recognition of developers;
- Now any design, documenting and algorithms of hardware **should have the description in the VHDL language;**
- The further advance in the field of computer-aided design can be accomplished by solving the problems of complete design automation, adequacy and building of diagnostic support.

Literature

1. Chang K.C. Digital design and modeling with VHDL and synthesis. IEEE: Computer Society Press, 1997.
2. Дорошенко А. VHDL - язык синтеза дискретных систем / www.aldec.com.ua.
3. VHDL Tutorial for IEEE Standart 1076 VHDL. CAD Language Systems, Inc.
4. Индустриальные стандарты и VHDL. / www.aldec.com.ua .
5. Берже Ж. М. VHDL'92. Новые свойства языка описания аппаратуры VHDL/Пер. с англ.- М: Радио и связь, 1995.-256с.
6. Организации и учреждения ведущие работы по проектированию средств вычислительной техники по технологии *Aldec, Inc.* / www.aldec.com.ua .
7. Армстронг Дж. Р. Моделирование цифровых систем на языке VHDL. М.: Мир.- 1992, 175с.
8. Бибило П.Н. Основы языка VHDL. Минск: Ин-т техн. кибернетики НАН Беларуси, 1999. – 202с.
9. Справочник по языку VHDL. / www.aldec.com.ua.
10. Компания *Aldec, Inc.* / www.aldec.com.
11. ACTIVE-VHDL Series. Book 3 "Quick Start Guide", ALDEC, Inc., May 1999, 195s.
12. ACTIVE-VHDL Series. Book 4 "User's Guide", ALDEC, Inc., May 1999, 213s.