DESIGNING A DIGITAL SYSTEM WITH VHDL

Valentina Stoyanova Kukenska

Dep. of Computer Systems and Technologies, TU of Gabrovo, Dimitar Str. 4,
5300 Gabrovo, Bulgaria, [1] tel. +359 66 223 456(411), e-mail: vally@tugab.bg,

Abstract: In this paper a digital system designing with VHDL is presented. Here are exposed sequentially all the phases of the very digital system's designing. The main methods are also on show here. The project descriptions' types are presented. The stress is put on the use of VHDL for synthesis of structural and behavioral models.

For creating the project of the chosen digital system an integrated system WebPack was used, as well as ModelSIm XE II for the model's simulation.

Keywords: Design, VHDL, digital systems, model, WebPack

1. INTRODUCTION

The digital systems are complex ones, consisting of lots of components. As far as the automated design of such systems is concerned, methods for designing time reducing and limiting the complexity of the task are sought out and applied. A method of the kind is connected with the decomposition and hierarchy principles. The decomposition of the systems is realized in a way, which differentiates functionally independent modules.

A digital system can be described as a module with inputs and/or outputs. The electrical values on the outputs are some function of the values on the inputs.

One way of describing the function of a module is to describe how it is composed of sub-modules. Each of the sub-modules is an *instance* of some entity, and the ports of the instances are connected using *signal*s. This kind of description is called a *structural* description.

In many cases, it is not appropriate to describe a module structurally. One such case is a module, which is at the bottom of the hierarchy of some other structural description. For example, if you are designing a system using IC packages bought from an IC shop, you do not need to describe the internal structure of an IC. In such cases, a description of the function performed by the module is required, without reference to its actual internal structure. Such a description is called a *functional* or *behavioral* description.

Usually, for structural and behavioral description, either Verilog or VHDL is used. In this paper a designing with VHDL is presented. Here are exposed sequentially all the phases of the very digital system's designing. The main methods are also on show here. The project descriptions' types are presented. The stress is put on the use of VHDL for synthesis of structural and behavioral models. Here are presented several VHDL models of computer systems' components.

2. METHODS AND STAGES IN DIGITAL SYSTEMS' DESIGN

In digital systems' design, as well as design of complex systems, a couple of methods are in use:

- top - down designing;
- up - down designing.

In top - down designing the building up of the system is usually started from below in upright direction through elaborating the element blocks' schemes, assembled later to form the whole product.

An advantage of this method is the use of representation on functional block level and the lower, the structural level, is addressed only during the error check simulations within the project.

The up-down designing starts with a specification on the highest level. After that, the project is being decomposed into functional blocks and the requirements for the income and outcome time proportions are specified. The functional models are described through behavioral models or by models on register levels and are subsequently simulated.

Some of the advantages of the methods are:

- an easier execution of the task's specifications;
- иt allows a projects' check on system level, without tackling the structural details;
- The project's check is done, with no regard to the technology of its realization. That allows that the choice of technology be made on a later stage of the designing project.

The most effective up-down designing method is the use of an abstract description of the scheme and the sequential details specifying of the different hierarchy levels' description.

The digital systems' design goes through the next stages:

- Specification;
- Functional (electrical) designing;
- Physical designing;
- Manufacturing;
- Testing.

Through specification the product parameters, necessary for its proper destination, are determined.

Through the functional (electrical) designing, the electrical scheme, responsible for the functions and parameters of the product, in terms of the specification, is elaborated.

The behavioral stage serves as a description for the scheme as a system, and its entries and exits are marked out. In most of the cases, VHDL models are used.

The Functional (electrical) designing deals with main functional blocks' elaboration. Usually a detailed VHDL description of the functional block is made and being checked by a VHDL simulation.

With the increasing complexity of the projects, for the elaboration on structural level, the technique of synthesis is applied. It allows that the scheme with logical elements be synthesized from a VHDL description. Through logical description details such as charging, elements' delay, are specified and crucial methods and problems with time scattering of signals are defined.

The Physical designing stages strongly depend on technology. The common task is concerned with the deploying of the logical elements and defining (tracing) their interrelations.

Provided that for the product realization PLD, CPLD or FPGA chips are used, then the result of the physical designing represents a configuration file for designing the chosen device's resources.

The testing of the project represents a number of procedures, used by designers, to provide:

- adequacy between project and specification;
- the execution of the project in terms of the chosen technology.

The designing process is usually iterative, including pre-designing of given parts, until the intended indicators are obtained.

For the tasks of testing in electrical designing (the functionality of the product and its electrical parameters), simulations are used.

The simulation on behavioral level defines how the product will run, before its actual compounding blocks are chosen. For working out of the behavioral models, the hardware description languages are used (VHDL, Verilog and others).

Through simulation, on a logical primitives level, the schemes are built up with basic logical elements "AND-NO", "OR-NO", invertors and triggers and are being simulated in order to find out irrelevances with their expected acting.

In functional testing, the delays are not concerned or they are supposed similar for all logical elements.

Error identification after the physical design

After topology's final elaboration are made the next procedures:

- check out of the tech norms throughout manufacturing;
- check out for the project's authenticity.

The tech norms for manufacturing are specific for each technological process. They include geometrical limitations for the mutual situation and the sizes of the objects in the multi-layer sketches.

The authenticity verification of the project aims to guarantee the product's proper working. It includes:

- finding out the interconnection of the scheme;
- finding out the parasite components of the topology.

3. TYPES OF DESIGN DESCRIPTIONS

Through the designing process, three types of design description are in use:

- behavioral;
- structural;
- physical.

The behavioral description tackles the system as if it were a kind of "black box" with its entrances and exits, with no regard to its structure. The aim is to ignore the redundant details and to concentrate on the specification of the necessary for the functions, which are to be done by the product. On this stage, languages for the apparatus part are used HDL (Hardware Description Languages) - VHDL, Verilog and others.

The structural description defines the way that the system is to be built up. Here, the system's structure, made of blocks and their interrelations, is tackled. The subsystems, which are to provide its functional execution, as well as their detailed description for analysis of the operational speed, charging and so on, are defined. The structural description can be presented by languages for the description of the hardware, as well as by electrical schemes.

The physical description shows exactly how, physically, this structure should be realized. For a given structural description, several physical realizations are possible. This description is connected with a concrete technological process.

The design process is connected with the transformations of the systems' descriptions and their sequential details specification. Decomposition from behavioral to structural description can be realized on a number of levels in a hierarchy. From the highest to the lowest, these levels can be outlined as it follows:

- system level;
- functional level;
- logical level;
- scheme level.

On the highest system level, the system's behavior is represented by algorithms that describe its functions. In order that these functions be executed, the architecture of the system is worked out, including microprocessors, memories, main boards and other structural components.

On the lower level, the system's behavior is described by Bolivia equations. For their execution, logical elements and triggers are used.

4. USE OF VHDL FOR SYNTHESIS OF STRUCTURAL AND BEHAVIORAL MODELS

VHDL is a Hardware Description Language for describing digital system [2].

VHDL is designed to full a number of needs in the design process. Firstly, it allows description of the structure of a design that is how it is decompressed into sub-designs, and how those sub-designs are interconnected. Secondly, it allows the specification of the function of designs using familiar programming language forms. Thirdly, as a result, it allows a design to be simulated before manufactured, so that designers can quickly compare alternatives and test for correctness without delay and expense of hardware prototyping.

VHDL contains a number of facilities for modifying the state of objects and controlling the flow of execution of modules.

In VHDL, an entity is such a module which may be used as a component in a design, or which may be the top-level module of the design. The entity declarative part may be used to declare items, which are to be used in the implementation of the entity.

Once an entity has had its interface specified in an entity declaration, one or more implementations of the entity can be described in architecture bodies. Each architecture body can describe a different view of the entity.

The declarations in the architecture body define items that will be used to construct the design description.

Signals are used to connect sub modules in a design. The sub modules in an architecture body can be described as blocks. A block is a unit of module structure, with its own interface, connected to other blocks or ports by signals. A signal assignment schedules one or more transactions to a signal (or port).

The primary unit of behavioral description in VHDL is the process. A process is a sequential body of code, which can be activated in response to changes in state. When more than one process is activated at the same time, they execute concurrently.

A process statement which can be used in an architecture body or block. The declarations define items which can be used locally within the process.

A process may contain a number of signal assignment statements for a given signal, which together form a driver for the signal.

VHDL descriptions write them in a design file. After then invoke a compiler to analyze them and insert them into a design library. A number of VHDL constructs may be separately analyzed for inclusion in a design library. These constructs are called library units. A design file may contain a number of library units.

There are two special libraries which are implicitly available to all design units, and so do not need to be named in a library clause. The first is called work, and refer to the working design library into which the current design units will be placed by the analyzer.

The second special library is called STD, and contains the packages standard. Standard contains all of the predefined types and functions.

The behavioral model represents a functional interpretation of the designed digital system. The hardware of the digital device is regarded as a kind of a discreet system. Its behavior is described as a number of operations. These operations are applied within the system's database. Within the creation of behavioral VHDL models, operations are described by processes and their interconnections-by signals. Processes represent programs, composed of sequential VHDL operators. On fig 1 is presented a VHDL model of a linear decipherer.

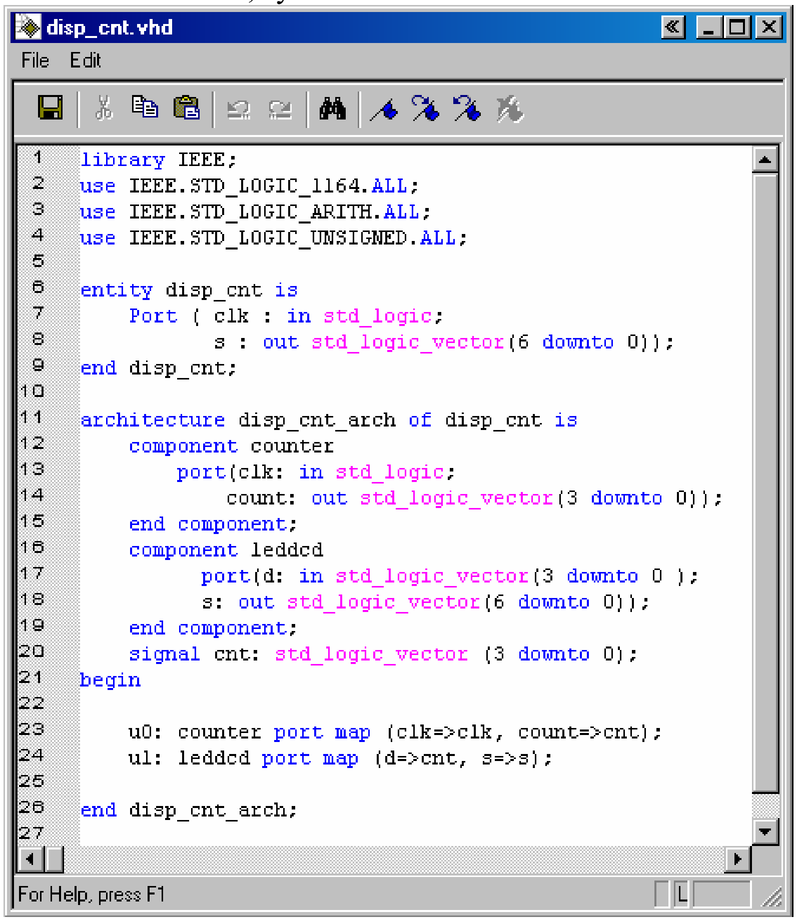
```
library IEEE;
  use IEEE.std_logic_1164.all;
entity DESHIF is
  port (x1,x2,x3 in: std_logic;
       J: out std_logic_vector(0 to 7));
end DESHIF;
arhitecture STRUCTURAL of DESHIF is
  component AND3
    port (I1,I2,I3: in std_logic;
         O1: out std_logic);
  end component;
  component NOT1
    port (I1: in std_logic;
         O1: out std_logic);
  end component;
  signal a,b,c: std_logic;
begin
  U1: NOT1 port map (I1=>x1,O1=>a);
  U2: NOT1 port map (I1=>x2,O1=>b);
  U3: NOT1 port map (I1=>x3,O1=>c);
  U4: AND3 port map (I1=>a,I2=>b,I3=>c,O1=>J(0));
  U5: AND3 port map (I1=>a,I2=>b,I3=>x3,O1=>J(1));
  U6: AND3 port map (I1=>a,I2=>x2,I3=>c,O1=>J(2));
  U7: AND3 port map (I1=>a,I2=>x2,I3=>x3,O1=>J(3));

  U8: AND3 port map (I1=>x1,I2=>b,I3=>c,O1=>J(4));
  U9: AND3 port map (I1=>x1,I2=>b,I3=>x3,O1=>J(5));
  U10: AND3 port map (I1=>x1,I2=>x2,I3=>c,O1=>J(6));
  U11: AND3 port map (I1=>x1,I2=>x2,I3=>x3,O1=>J(7));
end STRUCTURAL;
architecture DATA_FLOW of DESHIF is
signal T1,T2,T3: bit;
   begin
      T1<= not x1;
      T2<= not x2;
      T3<= not x3;
      F1<=T1 and T2 and T3;
      F2<=T1 and T2 and x3;
      F3<=T1 and x2 and T3;
      F4<=T1 and x2 and x3;
      F5<=x1 and T2 and T3;
      F6<=x1 and T2 and x3;
      F1<=x1 and x2 and T3;
      F1<=x1 and x2 and x3;
end DATA_FLOW;
```

fig.1 VHDL code of a linear decipherer

Structural VHDL models are means for reflecting the project's hierarchy. They are built up by decomposition of digital systems of functionally interconnected parts. These parts are presented as components, and their interconnections are realized through signals. These signals enter and exit the components via ports.

For example, within the designing of the digital module, presented on [2], its structural model has three main components-counter, decoder and light-diode seven-segment display. On fig.2 is shown a part of the structural model, synthesized in WebPack.



```
 1  library IEEE;
 2  use IEEE.STD_LOGIC_1164.ALL;
 3  use IEEE.STD_LOGIC_ARITH.ALL;
 4  use IEEE.STD_LOGIC_UNSIGNED.ALL;
 5
 6  entity disp_cnt is
 7      Port ( clk : in std_logic;
 8             s : out std_logic_vector(6 downto 0));
 9  end disp_cnt;
10
11  architecture disp_cnt_arch of disp_cnt is
12      component counter
13          port(clk: in std_logic;
14              count: out std_logic_vector(3 downto 0));
15      end component;
16      component leddcd
17            port(d: in std_logic_vector(3 downto 0 );
18            s: out std_logic_vector(6 downto 0));
19      end component;
20      signal cnt: std_logic_vector (3 downto 0);
21  begin
22
23      u0: counter port map (clk=>clk, count=>cnt);
24      u1: leddcd port map (d=>cnt, s=>s);
25
26  end disp_cnt_arch;
27
```

Fig.2 . VHDL structural description

For the structural section of the disp_cnt module is declared an I/O components interface, counter and a LED decoder (respectively line 12-15 and 16-19). On the 20th line is declared the internal four-bits counter (cnt), that transports values from the counter to the LED decoder.

On row 23 a resave of the counter module is conducted. The synchronizing entrance of the disp_cnt module is connected to the counter's synchronizing entrance, as well as the counter's exits are connected to the four-bits signal. On the other hand, on row 24, the cnt signal is linked to the LED decoder's entrances, while its exits are linked to the disp_cnt module's exits.

Once the structure and behavior of a module have been specified, it is possible to simulate the module by executing its bevioural description. This is done by simulating the passage of time in discrete steps. At some simulation time, a module input may be stimulated by changing the value on an input port. The module reacts by running the code of its behavioral description and scheduling new values to be placed on the signals connected to its output ports at some later simulated time. This is called scheduling a *transaction* on that signal. If the new value is different from the previous value on the signal, an *event* occurs, and other modules with input ports connected to the signal may be activated.

The simulation starts with an *initialization phas*e, and then proceeds by repeating a two-stage *simulation cycl*e. In the initialization phase, all signals are given initial values, the simulation time is set to zero, and each module's behavior program is executed. This usually results in transactions being scheduled on output signals for some later time.

In the first stage of a simulation cycle, the simulated time is advanced to the earliest time at which a transaction has been scheduled. All transactions scheduled for that time are executed, and this may cause events to occur on some signals.

In the second stage, all modules which react to events occurring in the first stage have their behavior program executed. These programs will usually schedule further transactions on their output signals. When all of the behavior programs have finished executing, the simulation cycle repeats. If there are no more scheduled transactions, the whole simulation is completed.

The purpose of the simulation is to gather information about the changes in system state over time. This can be done by running the simulation under the control of a *simulation monito*r. The monitor allows signals and other state information to be viewed or stored in a trace file for later analysis. It may also allow interactive stepping of the simulation process, much like an interactive program debugger.

The computer-synthesized models of the structure and behavior of the digital systems are used for the elaboration of project units. They are afterwards synthesized in the integrated WebPack system.

The ModelSim module realizes the project unit's functional VHDL simulation.

5. CONCLUSION

With the increasing complexity of the projects, structural presentation on a logical elements' level, becomes a hard, even impossible. Therefore, a higher abstraction level description would allow optimal results to be reached, such as consummation, characteristics, size and price.

The hardware description language VHDL is quite suitable for purposes of that kind. It can be used for a high-level behavioral description, as well as for detailed structural description.

This language provides:

- a standard way for documenting the project;
- means for creation of abstract simulation models, which can be used by each VHDL-simulator;
- possibility for an automatic synthesis of the electrical scheme from the project's abstract description.

The VHDL language allows the elaboration of a complete functional structural model of the specialized integral scheme, which can be simulated in order to assess its adequacy in terms of the specification's requirements. Thus, a higher quality of the project is guaranteed, because errors and problems are found out shortly after the start of the designing process.

6. REFERENCES

[1]. Lipsett R., C. Ussery, VHDL: Hardware Description and Design, 1989.

[2]. Kukenska V., I. Simeonov, Designing of a Digital module for Management of a Seven-segmented indication with Programming logics through the use of the Description language VHDL, The 12 International Scientific and Applied Science Conference, ELECTRONICS ET'2002, Sozopol, Bulgaria, September 25-27, 2003.

[3]. Navabi Z. , VHDL: Analysis and Modeling of Digital Systems, McGraw-Hill, 1993.