Abstract - FPGAs (“Field Programmable Gate Arrays”) constitute the base of many complex electronic systems with different applications ranging from the automotive sector to the multimedia market. Due to integrated circuits fabrication progress, current FPGAs includes very complex embedded digital blocks as serial transceivers or memory blocks. Therefore FPGA fundamentals and characteristics are not easy to explain considering that there are also a lot of devices and families from different manufacturers. By this reason, FPGA learning is usually based on describing a family and doesn’t give a general view of the wide range of commercial devices. Due to that it is essential for the electronic learning community to dispose of tools facilitating the learning process of the FPGA fundamentals and the design of systems based on them. The learning system combines a tutorial with hardware and software tools to achieve a friendly interface with a computer intended to facilitate FPGA distance learning for students with a basic knowledge of digital electronics and VHDL.

Index Terms - FPGA, distance learning, software, hardware.

INTRODUCTION

Figure 1 shows the block diagram of the system, including a hypermedia tutorial and hardware and software tools related by means of a set of practical examples. The hardware tools are:

- A main board with a FPGA and a USB2.0 driver [1].
- An expansion card including specific peripheral devices (analog to digital and digital to analogue converters, LCD display, keyboard, etc.).

The software tools are:

- The main board control software for the FPGA configuration and the communication channel implementation.
- The Altera’s Quartus II Web Edition tool for describing and synthesizing circuits using a schematic capture tool and a VHDL language compiler.

The tutorial is a hypermedia application running on a personal computer. It drives the student from the different FPGA basic concepts to actual FPGA digital system design. The set of practical examples included in the hypermedia tutorial are oriented to the APEX family circuits and tools from Altera but their functionality is generally enough to be used with devices of different manufacturers.

HYPERMEDIA TUTORIAL

Due to the fact that FPGA constitute a complex technology [2], the hypermedia system has been done using the method developed by the “Instituto de Electrónica Aplicada” of the University of Vigo [3] to obtain the descriptive model of a complex technology. Figure 2 shows this method comprising four main stages:

- Firstly, many different representative systems or devices are chosen.
- In the second stage the selected systems are analyzed in detail to define the concepts associated to the technology. This task is carried out in two different phases. In the first phase all the common characteristics are determined and classified to define the general characteristics or basic concepts of the complex technology. In the second phase the basic concepts are characterized (including functionality, implementation, architecture, etc.) taking into account the specific characteristics of each particular system in such a way that the subconcepts of the descriptive model are obtained as well as its dependence relations. The same subconcept can be present in different systems but the set of subconcepts associated to each system can be different.
- In the third stage all the basic concepts and subconcepts are structured to obtain the descriptive model.
- Finally, the descriptive model must be tested to verify its ability to describe not only the systems chosen to obtain the model but, all the commercial systems known.

Developing descriptive models is a tedious task requiring a lot of time and effort, nevertheless the result is a very useful tool for the analysis of complex technologies as well as the particular systems included in them. Besides, if new systems are developed, updating the model with the inclusion of new characteristics is very easy.
Once the descriptive model is obtained, it is necessary to use a graphical representation of it. So we studied the application of conceptual maps being used at present only to describe general human knowledge areas [4][5][6][7].

Figure 3 shows the developed FPGA’s conceptual map. The concepts are interrelated and due to that it is interesting to describe the map with hypermedia using the map concepts as key words to achieve a non sequential access to the information. Every concept is explained using web pages combined with a browser to implement the hypermedia system.

Every concept is described using text, pictures, diagrams and videos. The user can navigate through the lessons using a friendly interface [8]. To simplify navigation, every lesson has linkages with a glossary, including a multimedia definition of the different concepts. The interface includes a button to experiment which is active in those pages combining theory with the development board described next.

**FIGURE 2**
**COMPLEX TECHNOLOGIES CHARACTERIZING METHODOLOGY**

**USB2: FPGA DEVELOPMENT SYSTEM**

**I. Hardware Description**

Actual FPGA development systems do not have a high speed communication channel for data transfer between the FPGA and the computer. To overcome this limitation, the authors of this paper developed the USB2-FPGA board with the following features:

- **A communication channel based on the USB2 bus:**
  This channel can be used for the configuration of the FPGA as well as to support a high speed general purpose communication channel between the USB2-FPGA board and the PC. In comparison to other hardware platforms it reduces the configuration time providing a high speed channel support to communicate the configurable device with the PC. The control of the bidirectional data transfer is implemented in the FPGA but such a system is very simple and consumes just a few logic resources.

- **The input/output pins of the FPGA are accessible:** In order to facilitate the connection of peripherals required in different applications all the input/output pins of the FPGA can be accessed through standard connectors.
Figures 4 and 5 show a photo and a schematic of the developed board. The main parts are:

- The FPGA APEX EP20K100EQC240-2X from Altera.
- The USB2.0 controller from Cypress.
- The EEPROM EPC2LC20 from Altera for the storage of the configuration file.
- A +5V DC voltage source coming from an external AC/DC adaptor or from the USB connection. 1.8V and 3.3V DC voltages can be obtained from the +5V DC voltage source.
- Configuration mode selector. The board supports three configuration modes: from the PC through the USB channel, from the PC through the JTAG interface (passive serial mode) and from an EEPROM memory.
- A USB connector.
- A JTAG connector.
- Three 64 pins connectors to access all the input/output pins of the FPGA. Different expansion cards containing specific peripheral devices can be connected to the FPGA through the connectors.

II. Software Description

A program named “USB2-FPGA Control Panel” has been developed to support the USB2-FPGA board configuration and communication. As can be observed in Figure 6 the main functions of the program are the FPGA configuration or programming. The supervising of the data transfer between the FPGA and the PC is shown in Figure 7.

FPGA configuration timing is an important parameter for applications where the FPGA must be reconfigured many times. The programmer included in the “USB2-FPGA Control Panel” shows a significant reduction of the programming timing. In order to verify its performance a comparison with the Altera Quartus II programmer was made. The test conditions and the results demonstrate the superiority (in terms of programming time) of the developed system. These results are included in Table 1.
In the communication mode (Figure 7b) the USB2-FPGA Control Panel verifies the data transfer between the computer and the FPGA and supervises the transfer rate and the state of the FIFO memories (inside the USB controller) that temporally store the received and sent data.

To use the USB2-FPGA development system the channel communication between the computer and the FPGA must be established and the FPGA must be programmed. Figure 8 describes this process including the following stages:

- The PC initializes the USB controller through the USB2.0 channel.
- The PC configures the SIE unit of the USB controller and its internal microcontroller takes the control. The SIE unit includes the FIFO memories supporting the temporal storage of the transferred data.
- The internal microcontroller of the USB controller programs the FPGA transferring the configuration file from the PC to the device through the SIE unit and the JTAG interface.
- The USB controller gives the control of the SIE unit to the FPGA and the communication between the user application and the PC is established.

![FIGURE 8](image-url)

**TABLE I**

<table>
<thead>
<tr>
<th>Test conditions</th>
<th>Quartus II: BYTEBLASTERMV :</th>
<th>USB2.0-FPGA: CONTROL PANEL:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows XP</td>
<td>FILE FORMAT: *.sof</td>
<td>FILE FORMAT: *.rbf</td>
</tr>
<tr>
<td>Pentium® IV 2.40GHz.</td>
<td>SIZE: 121 KB</td>
<td>SIZE: 123 KB</td>
</tr>
<tr>
<td>RAM: 256 MB DDR</td>
<td>TIMING-&gt; 4.1 SEG.</td>
<td>TIMING-&gt; 1.31 SEG.</td>
</tr>
<tr>
<td>FPGA Device: APEX20K100EQC240-2X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The USB2-FPGA development system can be considered as a self-learning laboratory intended for FPGA applications by means of the resolution of practical exercises of increasing complexity. It is supposed that the user knows the basic digital blocks (logic gates, flip-flops, multiplexers, decoders, counters, memories, etc.) as well as the basis of the VHDL hardware description language.

In this way the development system has the following objectives:

- Make easy the learning of FPGA based systems design methods.
- Apply the design methods using the Quartus II Web Edition design tools from Altera.
- Improve the VHDL design skills.

A set of practical exercises has been developed with the before objectives. Simple digital systems of the first exercises are part of the more complex systems of the final exercises.

- **Exercise 1**: Digital control system of a 4x4 keyboard.
- **Exercise 2**: Digital interface of a PS2 keyboard.
- **Exercise 3**: Digital control system of a LCD display.
- **Exercise 4**: LIFO and FIFO memories.
- **Exercise 5**: Basic calculator using the systems of exercise 2 and 3.
- **Exercise 6**: Manchester serial transceiver with CRC.
- **Exercise 7**: Control system of a PWM analog to digital converter.
- **Exercise 8**: Digital control system of a successive approximation analog to digital converter.
- **Exercise 9**: Home alarm emulator using the systems of exercises 2, 3, 6, 7 and 8.
- **Exercise 10**: FIR filter using the systems of exercises 7 and 8.

The design process of FPGA based systems (Figure 9) includes the main stages referred next. In the USB2-FPGA development system most of the design stages use the Quartus II Web Edition design tools from Altera.

**Description**: The system behavior or/and structure is/are defined from the design specifications. Schematics are used to describe the system structure and hardware description languages, like VHDL or Verilog, are used for the behavioral description. Usually a joint description combining both structural and behavioral ones is used to define a specific system.

**Compilation**: During compilation a netlist containing all the system components and their interconnections is obtained, the right connection of the components is verified and possible syntax errors are detected. Optionally, the netlist can be optimized in order to improve the logic and interconnection resources usage. The resultant netlist is used for the implementation and verification stages.
• **Implementation**: In this stage FPGA logic resources are assigned to the different elements of the netlist (mapping process), placed and interconnected (place & route process or fitting process). Besides, the FPGA programming file is generated and a new netlist containing the delay of all signals is obtained. Using the updated netlist a timing simulation or a timing analysis can be accomplish in order to verify the right operation of the designed system. If verification results are the expected ones the FPGA can be programmed and the design is finished.

• **Verification**: This stage can be divided into three different processes: functional simulation, timing simulation and timing analysis. By means of functional simulation the system behavior can be verified without any timing consideration. When good results are reached the system can be implemented and if not the description must be modified. Timing simulation and analysis take place after the implementation. If the system does not work properly due, for example, to excessive signal delays, the implementation/compilation options must be changed in previous stages.

• **Programming**: The FPGA is programmed using the USB2-FPGA Control Panel tool.

I. **Virtual Logic Analyzer**

Besides the hardware and software resources described above, a virtual logic analyzer intended to verify the designed system behavior have been developed. The logic analyzer combines a hardware support and a software human machine interface (HMI) that runs in the PC.

The hardware support is implemented in the FPGA. It is made up of the analyzer input pins (data acquisition channels), a data acquisition memory and the communication processor that take charge of the acquired data transfer from the FPGA to the PC through the USB2.0 connection. This hardware is a module of a design library and must be included in the system that is being designed during the description stage. It consumes few logic and interconnection resources and does not inhibit the implementation of the proposed practical exercises.

To use the logic analyzer the acquisition channels (input pins of the logic analyzer) must be connected to the desired nodes of the implemented system. The nodes can be external...
(FPGA output pins) or internal signals. In the first case the output signals of the designed system must be wired to the input channels of the analyzer, just like an external instrument. In the case of internal node verification connections must be defined during the description stage. This type of verification can not be achieved with an external measurement system.

Figure 10 shows the logic analyzer human machine interface (HMI). It is a Visual C++ application for Windows, combining a graphic editor, where acquired signals are represented, with a control panel to configure the logic analyzer operation modes.

**CONCLUSIONS**

The main characteristics of the system are:

- It provides an efficient learning method combining a multimedia tutorial system with hardware to achieve not only a theoretical education but a practical training with a good cost/performance relation.
- It is appropriate to achieve asynchronous distance learning due to the low cost of the development board components and the free available software.
- The system configurability including a high number of input/output pins.
- The diversity of internal and external resources providing a high flexibility.
- The USB2.0 interface providing a very fast communication channel between the board and the PC.
- The virtual logic analyzer to test internal and external nodes of the designed system.

**ACKNOWLEDGMENT**

This work has been funded by “Xunta de Galicia” (Spain).

**REFERENCES**