VHDL Based Circuits Design and Synthesis on FPGA: A Dice Game Example for Education

Sarah Toonsi, Miznan G. Behri, S. M. Qaisar, Enas Melibari and Sarah Alolyan

Electrical and Computer Engineering Department, Effat University, Jeddah, KSA

e-mail:{satunsi, mgbahri, emelibari, salolyan}@effat.edu.sa, sqaisar@effatuniversity.edu.sa

***Abstract*—The DSP (Digital Signal Processing) and digital control have many advantages over the analog processing and control. Therefore, with the recent advancements of technology most of the signal processing and control tasks have been transferred from the analog to the digital domain. In order to keep the system efficient in terms of resources and power consumption, the specific system designs are opted. In this context, the VHDL (VHSIC Hardware Description Language) is extensively employed for these systems modelling and design. This paper presents an illustrative example of system modeling with VHDL for the education purpose. A synthesizable VHDL code is developed for a dice game. The system functionality is also verified with a *C*++ based system modeling and simulations. Unlike conventional programming methodologies, a relationship between the hardware and the developed software in kept. The developed system board level testing is performed. Results show a proper system functionality.**

Keywords-VHDL, C++; dice game; circuit synthesis; qartuas; fpga

I. INTRODUCTION

The DSP (Digital Signal Processing) and digital control have many advantages over the analog processing and control [1]–[4]. Therefore, with the recent advancements of technology most of the signal processing tasks have been transferred from the analog to the digital domain. The modern systems are becoming more and more complex. In order to cope with this complexity, while keeping the system efficient in terms of resources and power consumption, the specific system designs are opted. These designs are based on the specific parallel processing architectures along with concurrent system modules functionality. In this context, the VHDL (VHSIC Hardware Description Language) is extensively employed for these systems modelling and design. It is a hardware description language used in electronic design automation to describe digital systems such as field-programmable gate arrays and integrated circuits. VHDL can also be used as a general purpose parallel programming language [5].

The game development contributes to the field of programming in a very significant way[6]. A lot of algorithms can be tested by writing games, where developers can test different combinations and possibilities [6], [7]. It is more interesting to apply these codes on hardware, instead of just seeing output on the console [5]. In this context, an illustrative example of dice game modeling with VHDL is described for the education purpose. In order to test the system functionality at hardware level, a synthesizable VHDL code is developed for this system.

Section II describes the proposed system principle. The Section III discusses the *C*++ based system modeling and simulations based system functionality verification. Section IV describes the system VHDL based design. The proposed system implementation details are given in Section V. The developed system board level test results are presented in Section VI. Section VII finally concludes the article.

II. THE DEVISED SYSTEM

The proposed system block diagram is shown on Figure 1.

Figure 1. The proposed system block diagram

Rather than using basic discrete electronics based kits and spending time on soldering and putting the modules together. A sophisticated development board, DE2 from Terasic, is employed [8]. The DE2 board level diagram is shown on Figure 2. The Altera DE2 board consists of a set of Cyclone-IV FPGA, LEDs (Light Emitting Diodes), push buttons, switches, seven-segment displays and LCD screen [8]. There are SRAM, SDRAM, and Flash memories. For experiments that require a processor and simple I/O interfaces, it is easy to instantiate Altera Nios II processor and use standard embedded interface drivers such as RS-232, SPI, I2C, USB, Ethernet, etc. [8]. Moreover, cost effective, specific embedded interface drivers can also be developed [9]. Moreover, for applications that require sound or video signals, there are also standard connectors provided on the board. It allows to rapidly develop effective video, memory and audio embedded drivers for such applications [9]. These various peripherals, allows to implement an infinite number of games, based on the DE2 board [8].

978-1-5386-0969-9/17/$31.00 ©2017 IEEE

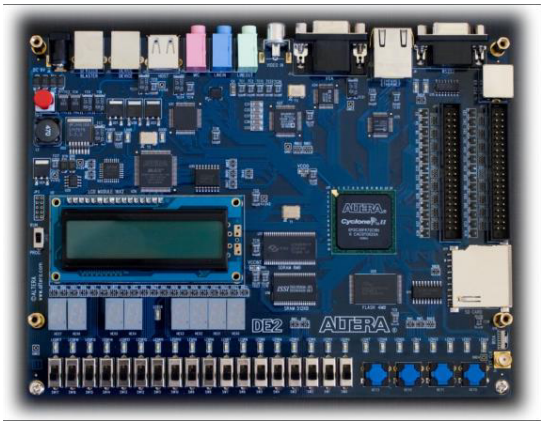

Figure 2. The DE2 board diagram [8].

Altera-Quartus is the CAD (Computer Aided Design) tool, needed to implement the VHDL system descriptions on the DE2 Board [10].

III. C++ Based System Modeling

The dice game C++ based modelling is much easier and rapid compared to the VHDL based modeling [11]. Therefore, at first step a *C++* based, dice game model is implemented for a simulations based functional verification.

The most important part of designing a dice game is to generate random numbers. There are many ways to produce pseudo random numbers. In C++ modeling, a built in function '*rand'* along with another built in function '*modulo*' are employed for the random numbers generation with a specified range. The developed dice game model is compiled and executed by employing the online C++ shell [12]. Results have shown a proper system functionality. A trial result of the Dice game C++ model is shown on Figure 3.

options | compilation | execution

```
Hello this is a new dice game :)
please enter number 1 to roll the dice
1
your dice number is:2
The computer dice number is:5
please enter number 1 to roll the dice
1
your dice number is:4
The computer dice number is:2
```

Figure 3: A Trail of the Dice game C++ Model.

IV. The VHDL Based System Development

The VHDL stands for VHSIC Hardware Description Language. Where VHSIC stands for Very High Speed Integrated Circuits. It is a language for describing digital electronic systems [5]. The VHDL is originally developed by the US department of defense. Later on it is evolved and subsequently adopted as a standard by the IEEE (Institute of Electrical and Electronic Engineers). It can be used for many purposes such as: modeling, simulating and describing hardware. A variety of CAD tools are available to synthesize the VHDL system descriptions for a possible implementation on FPGAs or ASICs (Application Specific Integrated Circuits)[10], [13], [14].

A VHDL code mainly consists of an entity description and architecture. The entity description is where all input and output ports are declared along with their types. On the other hand, the architecture is where the functionality is declared along with all operations and functions [5]. Unlike conventional coding, the sequence of code is not important in the VHDL. Codes in VHDL are compiled in a parallel fashion. The complete code will be compiled at once, and all commands will be executed in a concurrent fashion [5]. Therefore, specific coding style is required to introduce sequencing in a VHDL code.

A. Sequential Code

Random numbers are generated for both user and computer. To display numbers on seven-segment display an if statement is used. Then a comparison between both user and computer dice number is required. The one with higher score is the winner and his accumulative score is incremented. The game has only three rounds, and after that the winner will be declared. To avoid using loops, not synthesizable in VHDL, except the for loop [5]. Each game cycle is represented by the click of a push button. Therefore, three push buttons are employed. Each deals with a game phase or round. A process, with three push buttons in the sensitivity list, will do the job [5], [8]. Each time a button is pushed, new random numbers will be generated for both the player and the machine.

To be able to achieve a sequential implementation, a process is described. It is the only way to write sequential codes in VHDL. Different processes can be declared in one architecture [5]. Each process has a sensitivity list in which all parameters that will activate the process are added [5]. In the dice game description, the three push buttons and clock are added in the sensitivity list of the random numbers generator [5]. It allows the system to snoop the status of clock and these push buttons. Each time the clock or one of these push buttons, change their status the random numbers generation process is activated.

B. Arithmetic Operations

By including the two IEEE library packages, *NUMERIC_STD* and *STD_LOGIC_1164*, a lot of arithmetic operations become accessible [5]. However, while synthesizing, certain among these operations do not act according to the designer intension. Therefore, during the VHDL modeling of dice game. A special attention is paid to

the choice of appropriate synthesizable arithmetic operations [15].

C. *Random Number Generator*

In VHDL the random numbers can be generated, for simulation purpose, by employing the operation of '*Uniform()*'. However, this operation is not synthesizable. There are various ready algorithms to generate random numbers, however they possess a higher computational complexity and require a relatively higher implementation effort [16], [17]. In this context a low complexity, easy to implement and synthesizable random number generator is used. It employs the clock, on the DE2 board, to increment a pseudo value on each rising edge of the clock. However, this approach can generate a limited number of patterns. Therefore, the generated patterns my repeat through time.

V. The System Implementation

The system VHDL code is developed. The entity ports description is shown below.

```
Entity DiceGame is
port(
Reset, Game1, Game2, Game3, Clk : in std_logic;
Rled, Gled: out std_logic;
User_Score: out std_logic_vector(6 downto 0);
Computer_Score:out std_logic_vector(6 downto
0);
Computer_Total:out std_logic_vector(6 downto
0);
User_Total:out std_logic_vector(6 downto 0)
);
end Dicegame;
```

It shows that the entity '*Dicegame*' has two control ports. These are respectively named as *Reset* and *Clk*.

The employed '*Reset*' is an asynchronous one. After powering on the system, *Reset* is asserted to initialize the system. The *Reset* is the highest priority control port in the system. The second highest priority port is *Clk*. Once the *Reset* is deserted, the *Clk* takes charge of the system. The system operates on each rising edge of the clock.

There are also three ports for the push buttons respectively named as Game1, Game2 and Game3.

Two output ports, ***R****led* and **G**led are also employed. The ***R****led* is asserted to glow a red color LED if the computer wins the overall game. On other hand, the ***G****led* is asserted to glow a green color LED if the user wins the overall game.

User_Score is an output bus, composes of seven wires. It displays the user score, between [1; 6] for each trial of dice.

Computer_Score is an output bus, composes of seven wires. It displays the computer score, between [1; 6] for each trial of dice.

User_Total is an output bus,composes of seven wires. It displays the accumulated user score, between [0; 3] for an overall Dice game run.

Computer_Total an output bus, composes of seven wires. It displays the accumulated computer score, between [0; 3] for an overall Dice game run.

After system powering and initialization. The user should press the push buttons, **G**ame1, **G**ame2 and **G**ame3, in order. If **G**ame1is pressed, its signal value will become one. Then the Dice game circuit generates and displays the randomly generated numbers for both user and the computer on two different seven-segment displays [18]. These seven-segment displays are piloted respectively by the *User_Score* and the *Computer_Score* ports. Moreover, the Dice game circuit also decides that who got a larger dice number, among the user and the computer. After that winner score is set to one and the loser score is set to zero. If both are equal then no one earns any point and their respective scores remain zero. These scores are also displayed on the third and the fourth seven-segment displays [18]. These seven-segment displays are piloted respectively by the *User_Total* and the *Computer_Total* ports.

In second round the user press the **G**ame2. It converts its signal value from zero to one. Then the Dice game circuit generates and displays the randomly generated numbers for both user and the computer. Moreover, a winner decision is made. After that winner score is incremented by one and the loser score remains unchanged. If both are equal then no one earns any point and their respective scores remain unchanged. A similar process is repeated for the third round. The user commences this round by pressing the **G**ame3 push button

The first implementation step is to open a new project in the Altera-Quartus project manager [10]. Then the Dice game VHDL description is added in this project. Later on, a syntax verification is done in order to remove any possible syntax errors.

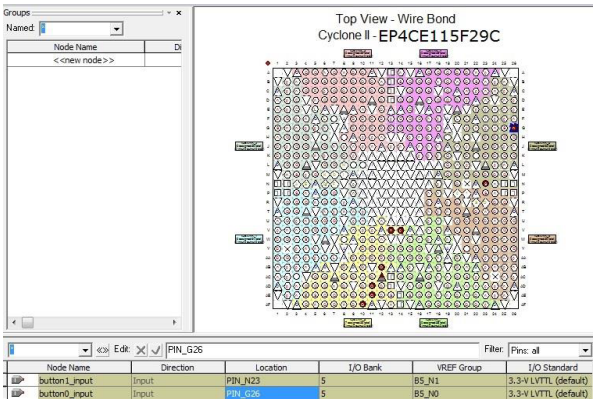

Figure 4. Pin Assignment with Quartus-Pin Planner

After successful syntax check, the design is compiled and the circuit netlist is generated. Then the circuit ports are assigned to the physically FPGA pins on the DE2 board. It is done via the Quartus-Pin Planner and with the help of the DE2 hardware design manual [8], [10] (cf. Figure 4).

The various DE2 board modules, employed in this project are:

A. *Clock*

A 50MHz Clock, generated on the DE2 board is employed to pilot the input port *Clk*.

B. *Push Buttons*

Four push buttons are employed to drive respectively the Reset, **G**ame1, **G**ame2 and **G**ame3 buttons.

C. *LED*

One Red and One Green LEDs are employed. These LEDs are respectively piloted by the ***R****led* and **G**led ports.

D. *Seven-segment Display*

Four seven-segment displays are used. These are respectively piloted by the User_Score, Computer_Score, User_Total and Computer_Total ports.

After pins assignment, the circuit is synthesized and a binary configuration file is generated for the Cyclone-IV FPGA. This FPGA is mounted on the DE2 board [8]. In order to configure the FPGA, the configuration interface is launched (cf. Figure 5). The USB-Blaster setup is configured and the binary configuration file is chosen [19]. The programmer configures the Cyclone-IV FPGA according to the configuration file via the USB-Blaster interface.

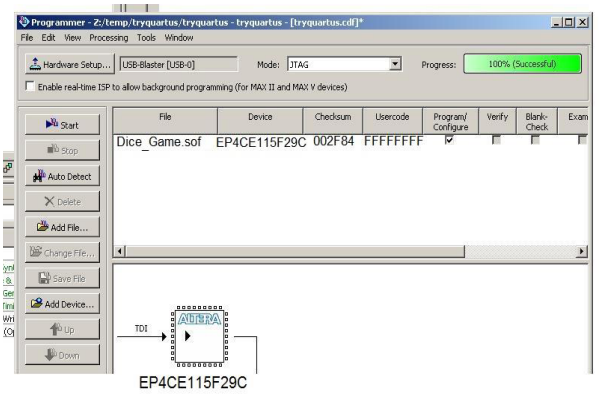

Figure 5. The FPGA Programmer Interface.

VI. RESULTS

The following pictures represent two trails of the dice game on DE2 board. In one trail the user wins (cf. Figure 6-A), while in the other the computer wins (cf. Figure 6-B).

Different test trails confirm a correct functionality of the developed system. The system performance can be improved by refining the random number generator. However, it will cost more in terms of the increased computational complexity, power consumption and the implementation efforts.

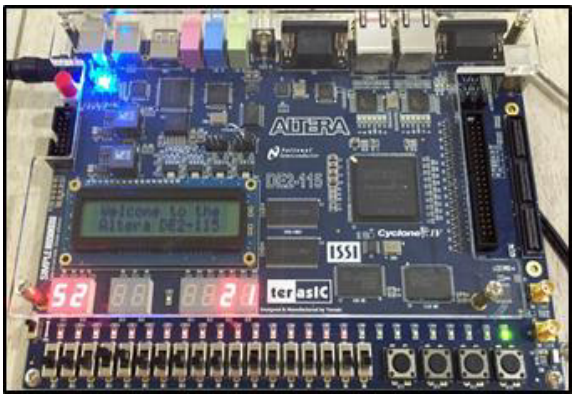

Figure 6-A. Trail of the Dice Game, where the User Wins

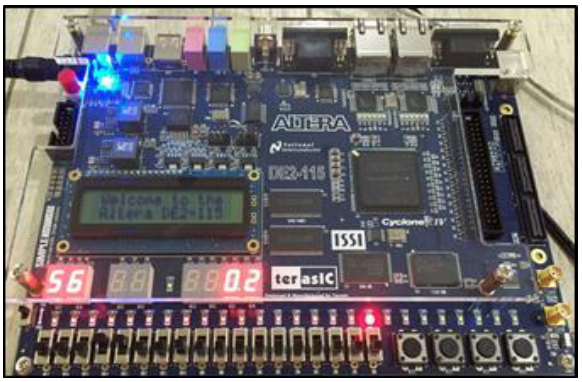

Figure 6-B. Trail of the Dice Game, where the computer Wins

VII. CONCLUSION

The VHDL is extensively employed for the design and development of specific and efficient solutions for the digital processing and digital control applications. In this context an illustrative example of system modeling with VHDL, for the education purpose, has been demonstrated. A synthesizable VHDL code has been developed for a Dice game. The system functionality has also been verified with a C++ based system modeling and simulations. The synthesizable VHDL design challenges have been discussed. A complete system implementation flow is described. The system functionality test results have been presented. It confirms a successful system implementation with a correct functionality. Employment of high level synthesis tools for the FPGA based systems development is a future work. Moreover, getting inspired from works presented in [20-23], the event driven features will be integrated in the system. It will add to the devised system performance compared to the counter

classical ones in terms of the system resources utilization like memory, power, etc

ACKNOWLEDGEMENT

Authors are thankful to anonymous reviewers for their valuable feedback.

REFERENCES

[1] E. C. Ifeachor and B. W. Jervis, *Digital signal processing: a practical approach*. Pearson Education, 2002.
[2] B. Gold, T. G. Stockham, A. V. Oppenheim, and C. M. Rader, "Digital processing of signals," 1969.
[3] S. W. Smith, "The scientist and engineer's guide to digital signal processing," 1997.
[4] J.-P. Su, *Digital control systems*. SRL Publishing Co., Champaign, IL, 1977.
[5] S. Sjoholm and L. Lindh, *VHDL for Designers*. Prentice Hall PTR, 1997.
[6] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*, vol. 1. Cambridge University Press Cambridge, 2007.
[7] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory*. SIAM, 1998.
[8] U. Manual, "Altera DE2 Board," *Altera Corporation*, vol. 72, 2006.
[9] E. Yusifli, R. Yahiaoui, S. M. Qaisar, F.-Z. Nacer, and T. Gharbi, "Design and development of camera interface controller for the OmniVision Digital HD Cameras Family."
[10] W. Kleitz, *Digital Electronics with VHDL, Quartus II Version*. Pearson Prentice Hall, 2006.
[11] G. De Micheli, "Hardware synthesis from C/C++ models," presented at the Design, Automation and Test in Europe Conference and Exhibition 1999. Proceedings, 1999, pp. 382–383.
[12] C++ Shell, "http://cpp.sh/."
[13] I. Xilinx, "Foundation," 2011.
[14] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 26, no. 2, pp. 203–215, 2007.
[15] S. K. Wood, D. H. Akehurst, O. Uzenkov, W. G. J. Howells, and K. D. McDonald-Maier, "A model-driven development approach to mapping UML state diagrams to synthesizable VHDL," *IEEE Transactions on Computers*, vol. 57, no. 10, pp. 1357–1371, 2008.
[16] D. B. Thomas and W. Luk, "Fpga-optimised uniform random number generators using luts and shift registers," presented at the Field Programmable Logic and Applications (FPL), 2010 International Conference on, 2010, pp. 77–82.
[17] D. B. Thomas and W. Luk, "High quality uniform random number generation using LUT optimised state-transition matrices," *The Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 47, no. 1, pp. 77–92, 2007.
[18] R. Steward, "Solid state displays," Mar. 1973.
[19] N. N. Joshi, P. Dakhole, and P. Zode, "Embedded web server on nios ii embedded fpga platform," presented at the Emerging Trends in Engineering and Technology (ICETET), 2009 2nd International Conference on, 2009, pp. 372–377.
[20] Qaisar, Saeed Mian, Laurent Fesquet, and Marc Renaudin. "Adaptive rate filtering a computationally efficient signal processing approach." Signal Processing 94 (2014): 620-630.
[21] Qaisar, Saeed Mian, et al. "Power efficient analog to digital conversion for the Li-ion battery voltage monitoring and measurement." Instrumentation and Measurement Technology Conference (I2MTC), 2013 IEEE International. IEEE, 2013.
[22] Qaisar, Saeed Mian, Laurent Fesquet, and Marc Renaudin. "Effective resolution of an adaptive rate ADC." SAMPTA'09. 2009..
[23] Qaisar, S.M., 2011. Electronic management system for rechargeable battery has measuring circuit measuring parameter determining variation of parameter transmitting data to electronic processing unit if variation is higher than predetermined threshold..