Teaching Computer Architecture and Organization using Simulation and FPGAs

Aws Yousif Fida El-Din and Hasan Krad

Abstract—**Computer architecture is often taught by using software to design and simulate hardware modules and then using individual components to implement them. Our aim in this paper is to share our teaching experience of this subject in a way to enhance student learning outcome by developing projects for the computer architecture lab to help students better understand the theoretical concepts of the subject and to gain hands-on type of experience and apply that for more realistic projects. As a result, we have noticed that students show better understanding of the subject over the last few semesters. We present in this work an ALU computer module design exercise as we used it in our computer architecture course in the Department of Computer Science and Engineering at Qatar University. The approach can be well adopted for a first course in digital logic design, computer organization, and/or computer architecture. In specific, we designed and implemented an 8-bit arithmetic and logic unit, which performs 14 different arithmetic and logic operations. We did the design, simulation, and FPGA-based implementation of the proposed ALU module using QUARTUS II design software and Altera DE2 FPGA Board.**

Index Terms—**Computer Architecture Education, FPGA, VHDL, ALU, Hardware Modeling.**

I. Introduction

Computer architecture is a core design course that is offered in our department at Qatar University. Teaching this course for computer science, computer engineering, and electrical engineering students in a conventional way is inefficient and insufficient if the teaching methods focus only on the theoretical aspects in the class room and using only design and simulation software tools. Many computer science departments offer such hardware courses using software tools to simulate different hardware modules. However, students who take such courses would not have the chance to do some experiments using real Field Programmable Gate Array (FPGA) boards and be able to acquire hands-on type of experience and skills to enhance their knowledge and expand their imagination. Hence, we used QUARTUS II design software and Altera DE2 FPGA Board in our computer architecture lab to accomplish this objective.

Quartos II design software and Altera DE2 FPGA board can be used in the laboratory for digital design, computer organization, and computer architecture courses. FPGAs offer the potential to design high performance systems with low cost [1]. In this paper, we present an FPGA design and implementation of an Arithmetic and Logic Unit (ALU). The FPGA-based optimized design of the ALU can be adopted as a way of enhancing computer architecture education [2]. It is important to note that we are not the first group to place an optimized design on an FPGA chip. Kassim Al-Obaidy [1], Mark Holland, Jamis Hafrris, and Scott Hauck [2], Yamin Li and Wanming Chu [3], and Andrew Koch and Ulrick Golze [4] had done similar approaches to improve the quality of teaching such a subject.

The goal of the teaching tool is to expose our computer science students with state-of-the art technologies and modernize the laboratory component of our computer architecture course by introducing system and component modeling using VHDL and FPGA programmable logic for mapping designs [5]. Teaching computer architecture using VHDL and synthesis tools helps students design much more complex systems within a shorter design cycle. Furthermore, by finally loading their code to the FPGA board, students in the computer science program gain strength in hardware implementation besides their deep knowledge in software development. As a result, this will enable CS students to develop complete systems, gaining both hardware and software experience, which usually CS students lack.

II. Methodology and Tools

The design and implementation of the proposed ALU is an important architecture design problem. The main objective of this paper is to design and implement an optimized 8-bit ALU that performs 14 different arithmetic and logic operations and a control input of 4-bits. The ALU unit was modeled using Very High Speed Integrated Circuits – Hardware Description Language (VHDL) and it was simulated and synthesized using Quartus II design software and finally the architecture is downloaded to an FPGA board using programmer option in QUARTUS II via USB blaster port as shown in fig1.

Fig. 1. Teaching Tool Layout.

The experiment involves hardware equipment and design

Manuscript received July 21, 2011 ; revised August 10, 2011

Aws Yousif Fida El-Din is with the Department of Computer Science & Engineering, College of Engineering, Qatar University, Qatar (e-mail: altaie@qu.edu.qa).

Hasan Krad is with the Department of Computer Science & Engineering, College of Engineering, Qatar University, Qatar (e-mail: hkrad@qu.edu.qa).

software. Students have been trained for four lab sessions on using the board and the design software prior the experiment. Materials used in the ALU design project are:

1) QUARTUS II design software.
2) ALTERA DE2 FPGA board.
3) Desktop or laptop with Windows 7 or SUSE Linux.
4) USB cable.

Students were required to perform the following tasks:

Task-1: Model the ALU using VHDL

Task-2: Simulation and synthesis.

Task-3: Loading the VHDL code in to the FPGA board.

Task-4: Testing the functionality of the ALU using LEDs and switches on the FPGA board.

III. Project Design

An 8-bit arithmetic and logic unit (ALU) design that takes two inputs A and B and then performs the following operations depending on the values of a control signal C [6]:

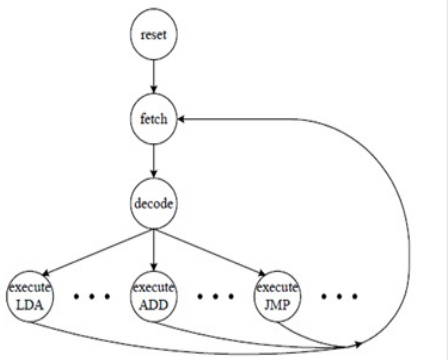


Fig. 2. State diagram for the control unit of Simple Processor

TABLE I. Arithmatic and Logic Unit Operatins

8-bit Arithmetic Operations	
Control Signals C	***Operations***
0000	A plus B
0001	Increment A by 2
0010	A Minus B
0011	Decrement A by 3
8-bit Comparison Operations	
Control Signals C	**Operations**
0100	Minimum of A and B
0101	Maximum of A and B
0110	A AND B
0111	A OR B
8-bit Shift Operations	
Control Signals C	**Operations**
1000	Circular right shift of A
1001	Circular left shift of A
1010	Right shift of A with feed in 0
1011	Left shift of A with feed in 0
1100	Right shift of A with MSB replication
1101	Left shift of A with LSB replication

The functionality of the ALU design is categorized to three main groups:

1- Arithmetic Operations: Inputs are A, B, and OP.
2- Comparison Operations: Inputs are A, B, and OP.
3- Shift Operations: Inputs are A, B, OP, and SB.

A typical design process starts with describing the circuit in VHDL. The VHDL compiler compiles the VHDL source code into a configuration file. The simulation tool can be used to verify the correctness of the design. The last step is to download the configuration file from the PC into the FPGA [7].

The project was extended to perform the Fetch – Decode – Execute Cycle using the Finite State Machine (FSM). The idea of finite state machine is to synchronize the transition of the three states. The finite state machine for the control unit basically cycles through four main states: reset, fetch, decode, and execute.

IV. Veda Coding

Mainly the VHDL code consists of two parts: The entity part and the architecture part. The entity part involves defining the input and output signals and the direction of the signals. The second part involves the functionality of the design [8]. A VHDL code for the proposed ALU is listed as follows:

VHDL Code for the ALU

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

ENTITY alu8bit IS
  port(a, b : in std_logic_vector(7 downto 0);
    op : in std_logic_vector(3 downto 0);
    SB : in std_logic;
    zero : out std_logic;
    f : out std_logic_vector(7 downto 0));
END alu8bit;

ARCHITECTURE behavioral of alu8bit IS
BEGIN
  process(a,b,SB,op)
  variable temp: std_logic_vector(7 downto 0);
    BEGIN
      case op is
            when "0000" =>
                  temp := a + b;
            when "0001" =>
                  temp := a + 2;
            when "0010" =>
                  temp := a - b;
            when "0011" =>
                  temp := a - 3;
            when "0100" =>
                  if a < b then
                  temp := a;
                  else
                  temp := b;
                  end if;
            when "0101" =>
                  if a > b then
                  temp := a;
                  else
                  temp := b;
                  end if;
            when "0110" =>
                  temp := a and b;
            when "0111" =>
                  temp := a or b;
            when "1000" =>
                  temp(6 downto 0) := a(7 downto 1);
                  temp(7) := a(0);
            when "1001" =>
                  temp(7 downto 1) := a(6 downto 0);
                  temp(0) := a(7);
            when "1010" =>
                  temp(6 downto 0) := a(7 downto 1);
                  temp(7) := '0';
            when "1011" =>
                  temp(7 downto 1) := a(6 downto 0);
                  temp(0) := '0';
            when "1100" =>
                  temp(6 downto 0) := a(7 downto 1);
```

```
                temp(7) := SB;
            when "1101" =>
                temp(7 downto 1) := a(6 downto 0);
                temp(0) := SB;
            when others =>
                temp := a - b;
        end case;
        if temp="00000000" then
          zero <= '1';
        else
          zero <= '0';
        end if;
        f <= temp;
        END process;
END behavioral;
```

V. SIMULATION

Once the VHDL code is compiled with zero errors, the proposed ALU design can be entirely simulated with the simulator included in the QUARTUS II design software.

Fig 3, 4, and 5 shows the trace windows for all signals generated by the simulator. We can examine the inputs A and B and the control input C as well as the final results in order to determine whether or not the prototype is correct.

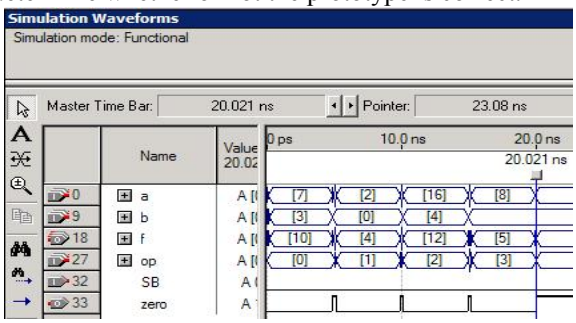


Fig. 3. Trace Window for the Arithmetic Operations.

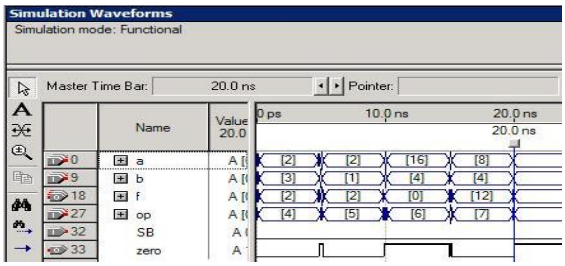


Fig. 4. Trace Window for the Comparison Operations.

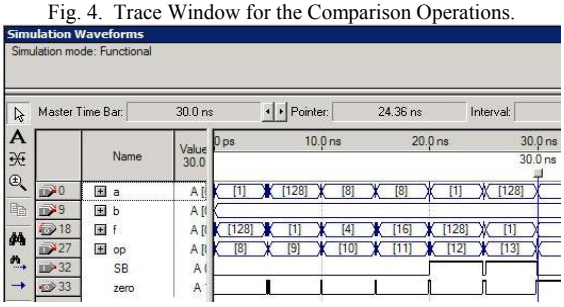


Fig. 5. Trace Window for the Shift Operations.

VI. SYNTHESIS

Once we have determined the correctness of our design functionality, we are ready to proceed with the Register Transfer Level (RTL) description of the design. RTL Viewer provides a block diagram view of a circuit at the level of registers, flip-flops and functional blocks that constitute the design. The displayed image, see Figure 6, is the circuit generated after the analysis and initial synthesis steps [9] [10].

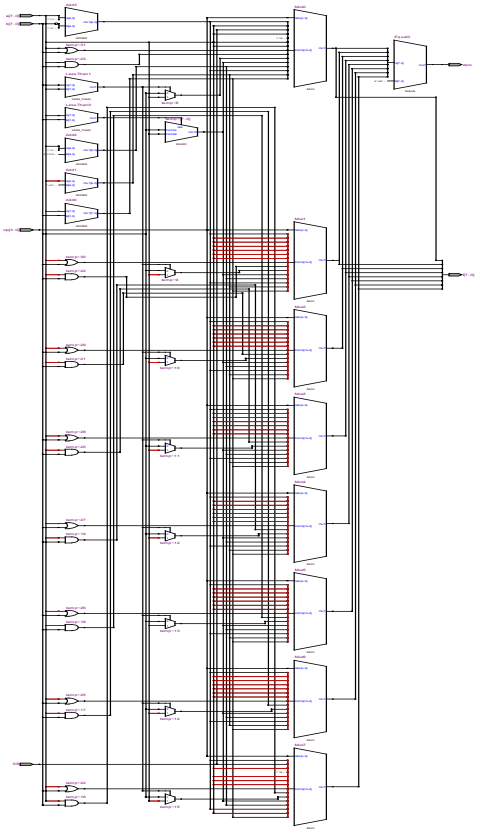


Fig. 6. The Top Schematic of ALU.

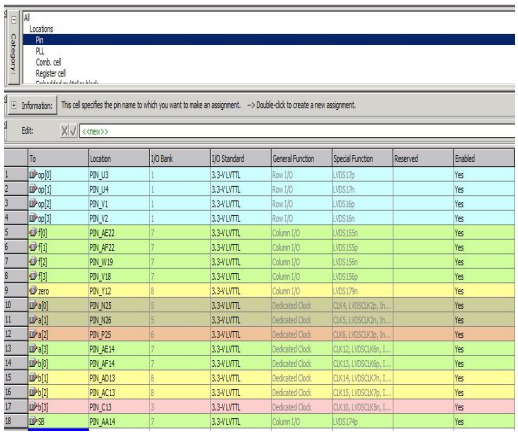
VII. FPGA IMPLEMENTATION

An FPGA chip has thousands of programmable gate arrays. VHDL is used to configure hundreds of these logic blocks and interconnections of the blocks [7]. The two major FPGA vendors, Altera [11] and Xilinx [12], distribute their software free of charge over the web and they do have a special arrangement for educational institutions so they can get free licenses for the commercial version of the software through Altera's and Xilinx's University Programs [13].

The Altera DE2 FPGA board was used to implement the proposed ALU. The board supports several features. The Altera DE2 development and educational board is a perfect tool for teaching different computer hardware courses [14]. We use this board as a tool for different educational and research activities.

The board has eighteen slide switches (SW0 – SW17). Switches (SW0 - SW3) are used for input A, switches (SW4 – SW7) are used for input B, and switches (SW14 – SW17) are used for the control input. There are eighteen red LEDs (LEDR0 – LEDR17) and nine green LEDs (LEDG0 – LEDG8). LEDs (LEDG0 – LEDG3) are used for displaying the results and the LED (LEDG9) is used for showing if there was a zero result (ZERO FLAG). Four bits are only used instead of eight bits, because of the switch and LED limitations. Figure 7 shows the pin assignments for the switches and LEDs and Figure 8 shows the Altera DE2 board.

The FPGA is configured either through the USB interface (USB Blaster) or by storing the configuration in the flash memory (4-MByte), which enables stand-alone operation [15].



	To	Location	I/O Bank	I/O Standard	General Function	Special Function	Reserved	Enabled
1	LEDR[0]	PIN_U3	1	3.3-LVTTL	Row I/O	LVDS12p		Yes
2	LEDR[1]	PIN_U4	1	3.3-LVTTL	Row I/O	LVDS12n		Yes
3	LEDR[2]	PIN_V1	1	3.3-LVTTL	Row I/O	LVDS16p		Yes
4	LEDR[3]	PIN_V2	1	3.3-LVTTL	Row I/O	LVDS16n		Yes
5	F[0]	PIN_AE22	7	3.3-LVTTL	Column I/O	LVDS155n		Yes
6	F[1]	PIN_AF22	7	3.3-LVTTL	Column I/O	LVDS155p		Yes
7	F[2]	PIN_W19	7	3.3-LVTTL	Column I/O	LVDS156n		Yes
8	F[3]	PIN_Y18	7	3.3-LVTTL	Column I/O	LVDS156p		Yes
9	zero	PIN_Y12	8	3.3-LVTTL	Column I/O	LVDS179n		Yes
10	a[0]	PIN_N25	5	3.3-LVTTL	Dedicated Clock	CLK9, LVDSCLK2p, In...		Yes
11	a[1]	PIN_N26	5	3.3-LVTTL	Dedicated Clock	CLK5, LVDSCLK2n, In...		Yes
12	a[2]	PIN_P25	6	3.3-LVTTL	Dedicated Clock	CLK6, LVDSCLK3p, In...		Yes
13	a[3]	PIN_AE14	7	3.3-LVTTL	Dedicated Clock	CLK12, LVDSCLK6n, I...		Yes
14	a[4]	PIN_AF14	7	3.3-LVTTL	Dedicated Clock	CLK13, LVDSCLK6p, I...		Yes
15	a[5]	PIN_AD13	8	3.3-LVTTL	Dedicated Clock	CLK14, LVDSCLK7n, I...		Yes
16	a[6]	PIN_AC13	8	3.3-LVTTL	Dedicated Clock	CLK15, LVDSCLK7p, I...		Yes
17	a[7]	PIN_C13	3	3.3-LVTTL	Dedicated Clock	CLK10, LVDSCLK3n, I...		Yes
18	LEDR8	PIN_AA14	7	3.3-LVTTL	Column I/O	LVDS174p		Yes

Fig. 7. Pin Assignment for LEDS and Switches.

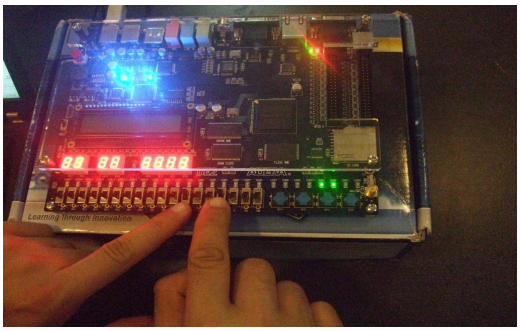


Fig. 8. Altera DE2 Board.

VIII. Observation

One of the biggest challenges in the course was linking the theory part to the practical part. The previous lab session for the architecture course was using assembly language, during which students faced several difficulties in understanding the assembly language sessions and linking them to the theory concepts. Relative to the previous years, students showed better performance and understanding using VHDL modeling and hardware implementation using FPGA boards.

Students were consulted on regular basis (every one week). The consultation helped the students overcoming their problems. Most of the problems faced by the students were either posted on the course website (http://mybb.qu.edu.qa) or sent via email to the students. Extra tutorials were arranged to overcome these problems and the students appreciate it enormously. Regular tutorials and assignments helped highlight many of the concepts.

Variety types of projects have been implemented, including, but limited to Dedicated Processor, General Purpose Processor, General Register Organization, Fast Multiplier, which enhances the level of students understanding.

IX. Conclusion

The paper addresses the importance of using simulation tools and FPGA development board to enhance classroom experience for computer architecture education at Qatar University.

The optimized design of an 8-bit ALU with fourteen different operations was an excellent motivating opportunity for computer science and engineering students not only to have hands-on type of experience in computer architecture, but also to be exposed to the software/hardware actual environment.

Using a simple hardware description language like VHDL and rapidly growing inexpensive FPGAs, a computer architecture course can be more fun to be taught to both computer science and computer engineering students.

It is worthwhile to mention that this teaching tool has been developed and implemented using a popular Altera DE2 board found in many universities. It has been tested in Qatar University for the 2nd year in an undergraduate computer architecture course and for senior design projects as well.

Computer science and computer engineering students showed better performance and understanding by using the Quartus simulation software and the FPGA boards. The researchers highly recommend using this FPGA board in teaching different computer hardware courses.

References

[1] K. Alodaidy, "Teaching Computer organization and architecture using simulation and FPGA applications". Journal of Computer Science, USA, 2007, pp624-632.

[2] M. Holland, J. Harris and S. Hauck, "Harnessing FPGAs for computer architecture education", Proceedings of the IEEE Intr. Conf. on Microelectronic Systems Education (MSE03), USA, 2003.

[3] Y. Li and W. Chu. "Using FPGA for computer architecture/organization education", IEEE Computer Society Press, USA, 1996, pp.31-35.

[4] A. Kock and U. Golze, "FPGA Application in Education and Research", Proceeding of the 4th EUROCHIP Workshop, Germany, 1993.

[5] S. Areibi, "A First Course in Digital Design Using VHDL and Programmable Logic", 31st ASEE/IEEE Frontiers in Education Conference, Reno NV, 2001.

[6] W. Stallings, *Computer Organization and Architecture Designing for* Performance, 7th ed., Prentice Hall, Pearson Education International, USA, 2006, ISBN: 0-13-185644-8.

[7] K. Imamura, Digital Logic Design for Computer Science,USA, 2005

[8] S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with VHDL Design*, 2nd ed., McGraw-Hill Higher Education, USA, 2005. ISBN: 0072499389.

[9] J. R. Armstrong and F.G. Gray, *VHDL Design Representation and Synthesis*, 2nd ed., Prentice Hall, USA, 2000. ISBN: 0-13-021670-4.

[10] Z. Navabi, *VHDL Modular Design and Synthesis of Cores and Systems*, 3rd ed., McGraw-Hill Professional, USA, 2007. ISBN: 9780071508926.

[11] Altera Corporation, web page, http://www.altera.com/.

[12] Xilinx Corporation, web page, http://www.xilinx.com/

[13] T. Hall and J. Hamblen, "Using FPGA to Simulate and Implement Digital Design Systems in the Classroom", ASEE Southeast Section Conference, USA, 2006.

[14] Z. Obaid, N. Sulaiman and M. Hamidon. "FPGA-based Implementation of DigitalLogic Design using Altera DE2 Board", International Journal of Computer Science and Network Security, Korea, 2009.

[15] O. Vainio, E. Salminen, and J. Takal, "Teaching Digital Systems Using a Unified FPGA Platform", Electronics Conference (BEC), 2010 12th Biennial Baltic, Estonia 2010.

Aws Yousif Fida El-Din, obtained his Master of Science Degree from the Faculty of Computer Science & Information Technology – University Putra Malaysia at 2001. Aws is currently a Lecturer in the Department of Computer Science & Engineering and a Cisco Academy Instructor in the Cisco Regional Networking Academy at Qatar University.. His Major Research Areas are CRISC Architectures and High Speed Multipliers. Aws is also involved in several Department and College Level Services. Aws is currently a CISCO Certified Academy Instructor CCAI. He has both Academic and Industrial CCNA Certifications. Aws is specialized in Modeling and Simulating Digital Systems using Hardware Description Language (VHDL), QUARTUS Design Software (Altera), and Field Programmable Gate Array Board (FPGA).

Hasan Krad, earned his Ph.D. in computer Science from Illinois Institute of Technology (IIT), Chicago, Illinois, in 1987, Master degree in Mathematics/Computer Science from Ohio University, Athens, Ohio, in 1984, and a B.Sc. degree in Electrical Engineering (Electronics) from Damascus University, Damascus, Syria 1976, Dr. Krad is currently working as a professor of Computer Engineering at the department of Computer Science and Engineering, college of Engineering, Qatar University. Prior to joining Qatar University back in 2004, Dr. Krad worked at other universities in the USA for seventeen years, and taught many different courses in the area of Computer Science, at both graduate and undergraduate levels, and he participated in many research projects and presented many refereed publications in different international conferences and journals.

Dr. Krad's research areas include Digital Logic Design, Hardware verification, Computer Architecture, and Parallel Processing. Dr. Krad served as a Chairman of the Department of Computer Science at Dillard University and was honored with the rank of Barron Hilton Endowed Professor of Computer Science From 1998-to-2002. Dr. Krad served on many committees at departments, colleges and universities levels. He is also very active in community services.

`