The Impacts of Hardware Description Languages and Large-Scale Programmable Logic Devices on Digital Hardware Design Education in Engineering Technology Programs

by

Hong "Jeff" Nie, Recayi "Reg" Pecen
Hong.nie@uni.edu pecen@uni.edu
Electrical and Information Engineering Technology Program
University of Northern Iowa

Abstract

With the development of computer-aided design and semiconductor technologies, the combined applications of Hardware Description Languages (HDLs) and large-scale programmable logic devices, such as Complex Programmable Logic Devices (CPLDs) and Field Programmable Gate Arrays (FPGA), have become dominant design and implementation tools for digital circuits and are widely used by both industry and academia. As a result, they will lead to many significant impacts on the architectures, contents, and methodologies of digital circuit design courses in Electrical Engineering Technology (EET) programs. In this paper, by comparing in detail the design procedures of three typical digital circuits when the traditional design methodology and the HDL and CPLD/FPGA combined methodology are employed respectively, we will clearly demonstrate the deep impacts of the HDL and CPLDs/FPGA combined methodology on digital hardware design courses, such as why some traditional skills are outdated and what the newly emerged skills are. From the design procedures of those typical digital circuits, we can conclude that the best way to teach the HDL and CPLD/FPGA combined methodology in EET programs is to integrate the methodology into all digital circuit design courses, from entry level to advanced level, rather than open an independent senior-level course.

I. Introduction

Digital circuit design has been considered as core education contents for Electrical Engineering Technology (EET) programs for more than two decades. As a result, a whole set of methodologies, such as Boolean logics, truth tables, canonical sum-of-products expressions, Demorgan's gate equivalents, and Karnaugh maps for combinational circuit, and finite state machines (FSMs), state transition diagrams, state excitation tables, redundant state minimization, and FSM implementation with D or JK flip-flops for sequential circuit, have been well developed to analyze and design digital circuits [1]. However, because the complexity to design a digital circuit increases exponentially with the number of gates used in the circuit, only small-scale digital circuits can be explored in digital circuit design courses when those traditional methodologies are employed.

In order to analyze and design large-scale digital circuits, Hardware Description Languages (HDLs), such as Verilog and VHDL (Very High Speed Integrated Circuits HDL), have been developed to describe the model and behavior of digital hardware [2]. Although HDLs are similar to a computer programming language in format, HDL codes are not programs to be executed on a computer. HDLs were originally developed for two purposes: first, as a documentation language to describe the structure and behavior of complex digital circuits designed by multiple designers; second, as an input to computer simulation software to simulate the operation of circuits [3]. Since HDL was first established as the IEEE standards in 1987, and then improved in 1993, many Computer-Aided Design (CAD) systems adopt HDL to provide documentation and simulation functions. Furthermore, with the development of CAD technologies, more and more CAD system also use HDL as design entry to provide synthesis functions, i.e. converting HDL codes into a hardware implementation of the described circuit [4]. Now HDLs have become dominant hardware developing tools to design, simulate and document large-scale digital circuits and are widely used by both industry and academia. Meanwhile, as a result of fast growth of semiconductor industry, Complex Programmable Logic Devices (CPLDs) and Field Programmable Gate Arrays (FPGAs) [5], which are large-scale Integrated Circuit (IC) chips containing millions of programmable logic gates and internal interconnects, become more and more economically available, and have completely superseded small-scale generic digital IC chips (e.g. 74xx and 4000 series) in today's digital circuit design [6]. The employment of CPLDs and FPGAs is strongly connected with HDLs. As shown in Fig. 1, first, HDL codes are written to describe the model and behavior of the desired digital circuit; second, those HDL codes are synthesized and converted to a programming file by CAD systems; finally, the logic gates and internal connections of CPLDs and/or FPGAs are programmed by the programming file to implement the desired digital functions.

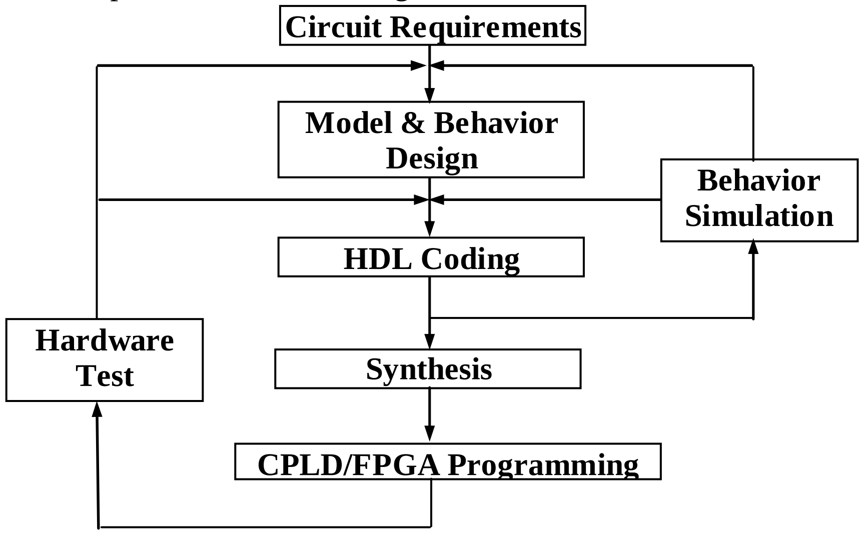

Fig. 1. Digital Circuit Design Flow with HDL and CPLD/FPGA

Due to the combined application of HDL and CPLD/FPGA, the emphasis of modern digital circuit design has shifted from how to realize digital circuits with minimum gates and interconnections to how to describe the model and behavior of digital circuits so that CAD systems can be employed to minimize design time [6]. Correspondingly, many traditional digital circuit design methodologies become outdated and many new design methodologies emerge. Inevitably, all those changes will lead to many significant impacts on the architectures, contents, and methodologies of digital circuit design education. To the best of our knowledge, most electrical engineering programs in US universities have integrated the combined application of HDL and CPLD/FPGA into their digital circuit design courses. However, EET programs in US universities are relatively slow in moving from the traditional digital circuit design education to an HDL and CPLD/FPGA integrated one. As it is shown in [7], less than 40% universities and colleges offering EET or similar programs cover or plan to cover within two years the topics about HDL and CPLD/FPGA in their digital circuit design courses. Furthermore, among those universities that cover or plan to cover the topics about HDL and CPLD/FPGA, some offer an independent senior-level course for HDL and CPLD/FPGA instead of integrating the topics into their existing digital circuit design courses.

In this paper, in order to clearly demonstrate the deep impacts of the combined applications of HDL and CPLD/FPGA on digital circuit design courses, i.e. why some traditional skills are outdated and what the newly emerged skills are, we will compare in detail the design procedures of three typical digital circuits when the traditional design methodology and the HDL and CPLD/FPGA combined methodology are employed respectively. From the three case studies, a straightforward conclusion is obtained that the best way to teach HDL and CPLD/FPGA in EET programs is to integrate them into all digital circuit design courses, from entry level to senior level, rather than open an independent course. The organization of this paper is as follows. After this introduction, the design of a ballot counter counting the number of ballots supporting or against a motion, which is a typical combinational logic circuit, is studied in Section II. Then the design cases of a sequence detector and a clock counter with specific output waves, which are typical sequential logic circuits, are studied respectively in Section III and VI. Finally, the effective ways to integrate the HDL and CPLD/FPGA combined methodology into digital circuit design courses in EET programs are discussed, and conclusions are given in Section V.

II. Design of a Ballot Counter

The ballot counter to be designed has five inputs and three outputs. The number of ‘1’ of the five inputs should be indicated at the three outputs in a binary format. According to the traditional digital circuit design methodology, the design procedures consist of the following four steps.

x4	*x3*	*x2*	*x1*	*x0*	*y2*	*y1*	*y0*	*x4*	*x3*	*x2*	*x1*	*x0*	*y2*	*y1*	*y0*
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
0	0	0	0	1	0	0	1	1	0	0	0	1	0	1	0
0	0	0	1	0	0	0	1	1	0	0	1	0	0	1	0
0	0	0	1	1	0	1	0	1	0	0	1	1	0	1	1
0	0	1	0	0	0	0	1	1	0	1	0	0	0	1	0
0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1
0	0	1	1	0	0	1	0	1	0	1	1	0	0	1	1
0	0	1	1	1	0	1	1	1	0	1	1	1	1	0	0
0	1	0	0	0	0	0	1	1	1	0	0	0	0	1	0
0	1	0	0	1	0	1	0	1	1	0	0	1	0	1	1
0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	1

0	1	0	1	1	0	1	1	1	1	0	1	1	1	0	0
0	1	1	0	0	0	1	0	1	1	1	0	0	0	1	1
0	1	1	0	1	0	1	1	1	1	1	0	1	1	0	0
0	1	1	1	0	0	1	1	1	1	1	1	0	1	0	0
0	1	1	1	1	1	0	0	1	1	1	1	1	1	0	1

Table 1. Truth Table for Ballot Counter

First, a truth table shown in Table 1 is generated. Then based upon this truth table, the canonical sum-of-products expressions [4] for *y2*, *y1*, and *y0* are derive as follows:

$$y2 = \sum m(15,23,27,29,30,31)\,, \tag{1}$$

$$y1 = \sum m(3,5,6,7,9,10,11,12,13,14,17,18,19,20,21,22,24,25,26,28)\,, \tag{2}$$

$$y0 = \sum m(1,2,4,7,8,11,13,14,16,19,21,22,25,26,28,31)\,. \tag{3}$$

Second, for each of the canonical sum-of-products expressions, a five-variable Karnaugh map [8] is used to reduce the number of product and sum terms in the expression so as to obtain a minimum-cost expression that uses the minimum number of gates and interconnections. Third, in order to implement the ballot counter with the minimum number of 74xx/4000 chips, Demorgan's Theorem [4] is applied on the minimum-cost expressions to convert AND, OR, NAND and NOR to their equivalent gates. Finally, according to the obtained minimum-cost and minimum-chip expressions, the ballot counter is implemented by connecting 74xx/4000 chips with wires on a breadboard. Obviously, the above-described design procedures are long and tedious. Students are easy to make mistakes during the design and implementation procedures, especially when they deal with those five-variable Karnaugh maps.

On the contrary, when the HDL and CPLD/FPGA combined methodology is employed to design this ballot counter, the design procedures are easy and straightforward. In this paper, we perform the designs with VHDL, a Xilinx ISE Foundation CAD system [9], and a Xilinx Spartan-3 FPGA development board [10]. First, the inputs and outputs of the ballot counter are defined with a VHDL construct called *entity* as follows:

```
entity ballot is
    Port (  X : in  STD_LOGIC_VECTOR (4 downto 0);
            Y : out  STD_LOGIC_VECTOR (2 downto 0));
end ballot;
```

Here the designed entity is assigned a name as *ballot*. Second, the functionality of the ballot counter is specified with another VHDL construct called *architecture* as follows:

```
architecture ballotfunc of ballot is
begin
with X select
    Y <=   "000" when "00000",
           "001" when "00001"|"00010"|"00100"|"01000"|"10000",
           "010" when "00011"|"00101"|"01001"|"10001"|"00110"|"01010"|
                       "10010"|"01100"|"10100"|"11000",
           "011" when "00111"|"01011"|"10011"|"01101"|"10101"|"11001"|
                       "01110"|"10110"|"11010"|"11100",
           "100" when "01111"|"10111"|"11011"|"11101"|"11110",
           "101" when "11111",
           "XXX" when others;
end ballotfunc;
```

Here the designed architecture is assigned a name as *ballotfunc*. In this architecture, we simply specify the expecting values of *Y* according to the input values of *X*. Since the Xilinx Spartan-3 FPGA development board has eight switches and eight LEDs, an accessory file is edited to connect *X* to five switches and *Y* to three LEDs so that the operation of the ballot counter can be verified in a straightforward way. Finally, the synthesis functions of the Xilinx ISE Foundation CAD system are employed to create a programming file by compiling the VHDL codes and the accessory file, and the obtained programming file is downloaded to the Xilinx Spartan-3 FPGA development board to implement the design. Clearly, when the HDL and CPLD/FPGA combined methodology is employed, we do not need to *i*) derive the canonical sum-of-products expressions given by Eqs. (1)-(3); *ii*) simplify the canonical sum-of-products expressions with the complicated five-variable Karnaugh maps; *iii*) derive a minimum-chip expression with Demorgan's Theorem; and *vi*) use small-scale IC chips, wires and a breadboard to implement the circuit. All those routine tasks are now implemented by the CAD systems. Consequently, with the HDL and CPLD/FPGA combined methodology, based upon our teaching experience, most EET student can design and implement a ballot counter within half hour; while, with the traditional digital circuit design methodology, an instructor may not be able to implement the same circuit for two hours.

III. Design of a Sequence Detector

The sequence detector under consideration is a sequential circuit with three inputs (*clock*, *reset*, and *w*) and one output *z*. If *reset*='0' and the previous four values of *w* were "1001" or "1111", *z*='1'; otherwise, *z*='0'. An example of the desired behavior is shown as follows:

w: 010111100110011111,

z: 000000100100010011.

When the traditional digital circuit design methodology is employed, the follow four design steps must be performed.

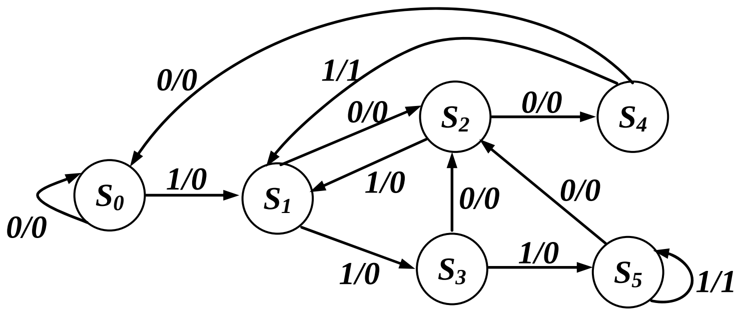

Fig. 2. State Transition Diagram of the Sequence Detector

First, a state transition diagram as Fig. 2 must be derived to describe the behavior of the FSM for the sequence detector. Second, the six states in Fig. 2 are encoded as Table 2. Assuming that D flip flops are used to implement the FSM, a state excitation table shown in Table 3 is generated according to the state transition diagram given by Fig. 2 and the state encoding table given by Table 2. Third, a combinational circuit with $Q_2Q_1Q_0$ and w as inputs and with $D_2D_1D_0$ and z as outputs is designed based upon the Boolean logic relation defined in Table 3. Finally, the sequence detector is implemented by wiring 74xx/4000 chips on a breadboard. Obviously, when the traditional digital circuit design methodology is employed, the design procedures of the sequence detector are even more complicated than those of the ballot counter, since both combinational and sequential circuits must be considered during the design.

State	S_0	S_1	S_2	S_3	S_4	S_5
Binary Code	000	001	010	011	100	101

Table 2. State Encoding Table of the Sequence Detector

Present State ($Q_2Q_1Q_0$)		000	001	010	011	100	101
w=0	Next State ($D_2D_1D_0$)	000	010	100	010	000	010
	z	0	0	0	0	0	0
w=1	Next State ($D_2D_1D_0$)	001	011	001	101	001	101
	z	0	0	0	0	1	1

Table 3. State Excitation Table of the Sequence Detector

On the other hand, when the HDL and CPLD/FPGA combined methodology is employed, the core design procedure is to write the VHDL codes as follows to describe the state transition diagram shown in Fig. 2:

```
entity sequence is
    Port (  clock : in  STD_LOGIC;
            reset : in  STD_LOGIC;
            w : in  STD_LOGIC;
            z : out  STD_LOGIC);
end sequence;

architecture Behavioral of sequence is
    type State is (S0, S1, S2, S3, S4, S5);
    signal PresentState, NextState: State;
begin
    seq: process (clock, reset) is
    begin
        if reset = '1' then
            PresentState <= S0;
        elsif rising_edge (clock) then
            PresentState <= NextState;
        end if;
    end process seq;
    com: process(PresentState, w) is
    begin
        z <= '0';
        case PresentState is
            when S0 =>
                if w = '0' then
                    NextState <= S0;
                else
                    NextState <= S1;
                end if;
            when S1 =>
                if w = '0' then
                    NextState <= S2;
                else
                    NextState <= S3;
                end if;
            when S2 =>
                if w = '0' then
                    NextState <= S4;
                else
                    NextState <= S1;
                end if;
            when S3 =>
                if w = '0' then
                    NextState <= S2;
                else
                    NextState <= S5;
                end if;
            when S4 =>
                if w = '0' then
                    NextState <= S0;
                else
                    NextState <= S1;
                    z <= '1';
                end if;
            when S5 =>
                if w = '0' then
                    NextState <= S2;
                else
                    NextState <= S5;
                    z <= '1';
                    end if;
            end case;
    end process com;
end Behavioral;
```

Similar to the ballot counter design, once the VHDL codes for the sequence detector is ready, the rest design tasks will be implemented by the CAD systems.

IV. Design of a Clock Counter with Specific Output Waves

The clock counter under consideration has two inputs and four outputs. One input (*CLK*) is a 50MHz clock, and the other input (*RESET*) is a reset which forces all outputs to be '1' when it is '1'. When the reset is '0', the four outputs (*AN0-AN3*) must have the specific waves shown in Fig. 3, and the duration of each output to be '0' is 1ms. When the traditional digital circuit design methodology is employed, multiple sequential and combinational logic units must be combined together to achieve the required functionality, and many 74xx/4000 chips must be wired together to implement the clock counter.

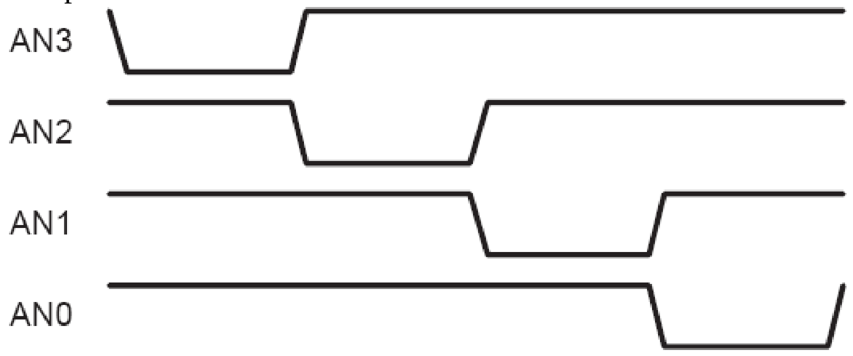

Fig. 3. Output Waves for the Clock Counter

However, when the HDL and CPLD/FPGA combined methodology is employed, the clock counter can be designed with the VHDL codes as follows:

```
entity counter is

    Port (  CLK : in  STD_LOGIC;
            RESET : in  STD_LOGIC;
            AN : out  STD_LOGIC_VECTOR (3 downto 0));
end counter;

architecture Counterfunc of counter is
begin
p0: process (CLK, RESET) is
    variable refresh : std_logic_vector(17 downto 0);
    begin
        if RESET = '1' then
            AN <= "1111";
            refresh := (others => '0');
        elsif rising_edge (CLK) then
            refresh := refresh + 1;
            if refresh < 50000 then
                AN <= "1110";
            elsif refresh < 100000 then
                AN <= "1101";
            elsif refresh < 150000 then
                AN <= "1011";
            elsif refresh < 200000 then
                AN <= "0111";
            else
                AN <= "1110";
                refresh := (others => '0');
            end if;
        end if;
    end process p0;
end Counterfunc;
```

In this VHDL codes, again instead of describing how to realize the counter with digital logic, we simply specify the desired behaviors of the counter, which can be summarized as follows:

(1) If *RESET*='1', *AN*="1111" and *refresh*=0.
(2) For every rise edge of *CLK*, *refresh* adds one.
(3) When $0 \le refresh < 5000$, *AN*="1110"; $5000 \le refresh < 10000$, *AN*="1101"; $10000 \le refresh < 15000$, *AN*="1011"; and $15000 \le refresh < 20000$, *AN*="0111".
(4) If *refresh* is other value, let *refresh*=0, and *AN*="1110".

Then the rest design tasks are simply and straightforward: editing an accessory file to connect *AN*, *CLK*, *RESET* to the suitable ports in the FPGA development board, creating a programming file with the synthesis functions of the CAD systems, and downloading to the FPGA development board.

V. Discussions and Conclusions

From the three circuit design cases demonstrated in Sections II, III, and IV, it is clear that with more and more CAD system provide synthesis functions, i.e. automatically converting HDL codes into a hardware implementation of the described circuit, the core design activities has changed from how to realize digital circuits to how to describe the model and behavior of digital circuits. As a result, among the traditional design methodologies, those focusing on the model and behavior of digital circuits, such as truth tables for combinational circuits and state transition diagrams for sequential circuits, become more and more important. Meanwhile, the methodologies focusing on circuit implementation, such as canonical sum-of-products expressions, Karnaugh maps, and Demorgan's gate equivalents for combinational circuit, and state excitation tables, redundant state minimization, and FSM implementation with D or JK flip-flops for sequential circuit, become outdated since now circuit implementation is performed by CAD systems. Furthermore, some new methodologies focusing on the combined application of HDL and CPLD/FPGA, such as HDL coding and circuit behavior simulations, emerge.

It is well known that in traditional digital circuit design courses, so much time and efforts have been spent in training students to use those traditional circuit implementation methodologies. It is for the reason that without the help of synthesis functions in CAD systems, those circuit implementation methodologies are the prerequisite methods to implement any digital circuit. In other words, unless the HDL and CPLD/FPGA combined methodology is integrated into all digital circuit design courses, from entry level to advanced level, we cannot cross out those traditional circuit implementation methodologiese from digital circuit design courses. Therefore, offering an independent senior-level course for HDL and CPLD/FPGA in EET programs is definitely not a good solution. On the contrary, by integrating HDL and CPLD/FPGA into all digital hardware design courses, we can transfer the time and efforts previously spent in studying those traditional circuit implementation methodologies to study the combined application of HDL and CPLD/FPGA, and hence the over load of the digital circuit design courses will not increased. Furthermore, by employing the HDL and CPLD/FPGA combined methodology, students can have more opportunities to explore the design of large-scale digital circuits with the same design methodologies as currently used in industry. For example, for a project designing a three-scale four-digit frequency meter, if the traditional digital design methodology is employed, even in the point of view of an instructor, it would be a quite complex digital project, and EET students would need to have a dedicated course like Senior Design to implement this project. However, when this project is used as an in-course project for the second digital circuit design course in which the combined application of HDL and CPLD/FPGA is taught, our teaching records show that most EET students can implement the project smoothly.

In summary, by integrating the HDL and CPLD/FPGA combined methodology into all digital circuit design courses, from entry level to advanced level, we can modernize our digital circuit design education in EET programs so as to keep pace with the fast developing digital electronic industry. Meanwhile, with the development of semiconductor and CAD technologies, not only the FPGA development kits, but also the CAD systems supporting HDL synthesis become economically available. For example, Digilent Inc. provides a FPGA development board, which consists of a Xilinx Spartan-3 FPGA with 200,000 gates, a fast asynchronous SRAM with 1Mega bytes, and various on-board I/O devices, with $99 per board [10]. Through Xilinx's University program, any university can apply for a donation from Xilinx to get its CAD systems for digital circuit design courses for free [9]. So it is right time for EET programs in US universities to move from the traditional digital hardware design education to a HDL and CPLD/FPGA integrated one.

References

[1] J. F. Wakerly, *Digital Design principles and Practices*, 3rd edition, Prentice Hall, 1999.
[2] J. Bhasker, *A VHDL Primer*, 3rd edition, Prentice Hall, 1998.
[3] Z. Navabi, *VHDL-Analysis and Modeling Digital Systems*, 2nd edition, McGraw Hill, 1998.
[4] S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with VHDL Design*, 2nd edition, McGraw Hill, 2005.
[5] K. Skahill, *VHDL for Programmable Logic*, Addison Wesley, 1996.
[6] M. Green, "Standing in the shadows of giants", *ElectronicsWeekly.com*, July 18th, 2005, http://www.electronicsweekly.com/Articles/2005/07/18/35857/Standing+in+the+shadows+of+giants.htm.
[7] R. Furtner and N. Widmer, "Technology Education and the New Frontier of Digital Electronics," Proceedings of ASEE Annual Conference, Chicago, IL, June 2006.
[8] M. Karnaugh, "A Map Method for Synthesis of Combinatorial Logic Circuits," T*ransactions of AIEE, Communications and Electronics* 72, part 1, November 1953.
[9] http://www.xilinx.com/univ/don_program.htm.
[10] http://www.digilentinc.com.