An FPGA Experience in ASIC Design

Liang Dong and Massood Z. Atashbar
Department of Electrical and Computer Engineering
Western Michigan University
Kalamazoo, MI 49008

Abstract

This paper accounts the development of the class projects for a graduate level course on application-specific integrated circuit (ASIC) design. As the use of field programmable gate arrays (FPGAs) in digital systems is on the rise, and they are increasingly competitive when compared to traditional ASICs, these class projects assure that the course provides students with an experience in FPGA design methodology and implementation.

Introduction

ASICs allow enhanced functionality and superior performance of electronic equipment due to the steadily decreasing cost per function and the increasing integration density [1]. As part of a graduate ASIC design course, students participate in projects that use FPGAs as target chips. The design methodology for ensuring successful and reliable FPGA designs is less stringent than that of traditional ASICs. This is due to the ease and low cost of implementing design changes in FPGAs and because FPGAs normally host smaller, simpler, and less-critical functions.

ASIC/FPGA design and verification techniques are discussed in class. Hardware description language (HDL) inference of FPGA resources and coding examples are provided. Advanced techniques using the Xilinx implementation tools are also introduced. The class projects consist of regular assignments and a final team project. The regular assignments are sequenced to facilitate comprehension, with each successive exercise building on concepts demonstrated previously. The final project is more challenging, with innovation and creativity in design being emphasized.

The FPGA-based development boards that were used for the projects include the Digilent D2SB-DIO4 Combination Board and the Spartan-3 Starter Board. The D2SB-DIO4 Board features a 200K-gate Xilinx Spartan 2E XC2S200E FPGA in a PQ208 package that provides 143 user I/Os. The Spartan-3 Board features a 200K-gate Xilinx Spartan-3 FPGA, on-board I/O devices, and 1MB fast asynchronous SRAM. Figure 1 shows the block diagram of the Spartan-3 FPGA board. These boards make good platforms of experiments with designs from a simple logic circuit to an embedded processor core.

Copyright ©2005, American Society for Engineering Education

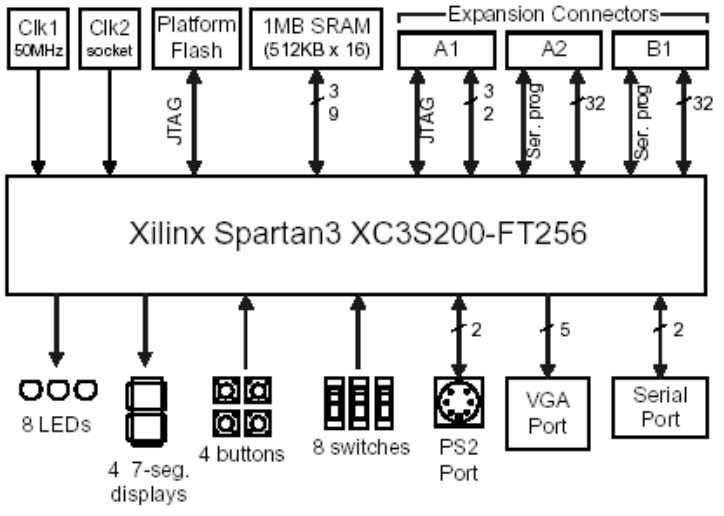

Fig. 1. Spartan-3 Starter Board block diagram. (Source: Digilent product brochure)

Class Assignments

VHDL designs of standard combinational and sequential logic modules are provided as class examples [2, 3]. The class assignments require FPGA implementations of simple systems built on these blocks. We use integrated design environment (IDE) to target the FPGA on the development board, and blend ModelSim® into a comprehensive, HDL-based FPGA design flow. Design verification and testability analysis are woven through the entire design circle.

(1) Counter and Stop Watch

The first part of the assignment is to implement a four-digit decimal counter on the four seven-segment displays of the FPGA development board. A push-button on board resets the displays to "0000". A switch SW1 determines whether the counting step is +1 or -1 on the fly. That is, when SW1 changes, the counter will start increment or decrement at the next clock edge from the current displayed value.

This exercise familiarizes the students with a synchronous design. Students learn how FPGA responds to the inputs and controls the peripheral devices. Using the on-chip clock, students are required to design a clock divider to provide constant clock rate for the customer use.

The second part of the assignment is to implement a stop watch on the displays. The left two digits show seconds from "00" to "59". The third digit shows the tenth of a second, and the last shows the hundredth of a second. The on-chip clock was divided to provide the stop watch with a resolution of 10 ms. Therefore, the range of the stop watch is one minute, such that "59.99" will increase and change to "00.00". A decimal point between the second and third digits indicates this time scale.

Copyright ©2005, American Society for Engineering Education

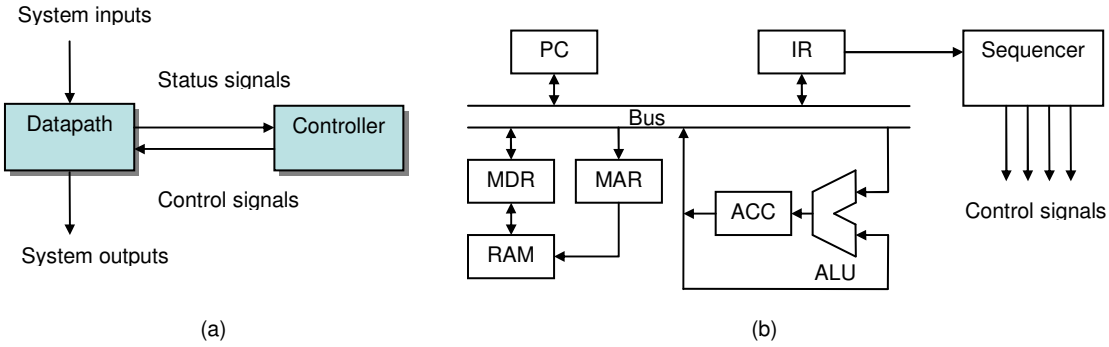

Fig. 2. (a) Datapath/controller partitioning. (b) Microprocessor block diagram.

When switch SW1 turns on, the watch resets and starts to count from "00.00". When SW1 turns off, the watch stops and the stop time remains on the displays. In addition, a push-button on board can be used to freeze the displays without stopping the watch. The split/lap times are stored and can be recalled. Students are required to discuss the clock resolution and possible clock drift and compensation.

(2) A Simple Microprocessor

It is convenient to partition a sequential system into a datapath and a controller (Figure 2a) [3]. The datapath consists of the components that store and manipulate data. Those components can be previously designed or easily adapted. The controller controls the functioning of the datapath components. This assignment uses a simple microprocessor to demonstrate the optimal partitioning of the state machine. The VHDL model is synthesized and implemented on an FPGA.

As shown in Figure 2b, the datapath components of the microprocessor include arithmetic and logic unit (ALU), accumulator (ACC), random access memory (RAM), memory address register (MAR), memory data register (MDR), program counter (PC), and instruction register (IR). They are connected to a single bus, and each uses three-state buffers to avoid bus access conflict. As the functionality of the system is mainly contained in the datapath, the system can be described in terms of register transfer operations. A single reset initializes all sequential blocks, and a single clock drives all blocks to keep the system synchronous.

A Sequencer is operated as the controller, which is a design-specific state machine controlling the overall functioning of the microprocessor. It takes instructions as inputs, such as load, store, add, sub, and generates control signals as outputs. The control signals may include: load IR from bus, load MAR from bus, load MDR from bus, drive bus with ACC, drive bus with PC, drive bus with MDR, load ACC with ALU, increment PC, perform addition in ALU, perform subtract in ALU, etc.

Copyright ©2005, American Society for Engineering Education

(3) FIR Filter

This assignment is to implement an FIR filter on the FPGA. It provides students with a grasp on design description, VHDL coding, verification, synthesis, place and route, and postlayout verification. Using a single FPGA, students are required to determine the number of FIR filter taps and the register width. If two or more FPGAs can be cascaded into an ASIC chain, students are required to design the connections. In order to support a fault-tolerant architecture, the design should be able to bypass any FPGA in the chain to allow that FPGA to be ignored if it fails.

To improve the synthesis run time, design partitioning is applied. The design is partitioned hierarchically and functionally such that the architecture is intended for synthesis. It also allows the building blocks, such as the multiplier and the adder, to be easily substituted with different architectures. The individual lower-level blocks are combined using the glue logic. Students are also exposed to how generics and signals are mapped.

Another partition consideration is for the place and route. The place and route tool is more efficient for the smaller blocks. Smaller blocks can be placed and routed first. They are then combined with the rest of the circuit and hierarchically placed and routed for the whole design. After the placing and routing process is done, a back-annotated timing file can be generated for postlayout verification.

In this FPGA assignment, we consider the clock skew and fix [4]. The clock skew may introduce setup and hold time violations. The design contains clock and reset circuits. The signal from the lowest-hierarchy buffers of the clock and reset trees is fed to every flip-flop.

Team Projects

Many course projects are proposed that are feasible with the available FPGA development boards. These may include fast ALU design, video processing, image filtering, image and audio compression, asynchronous communications, error control coding, and automatic control simulation. The following are some team projects that were completed during the semester of fall 2004.

(1) Video Processing

Digital video requires fast data handling and is therefore a good application of FPGAs. This project implements a video processing board using the Spartan-3 FPGA as the core. It has an interface to the PC for importing images. While the Spartan FPGA family has internal RAM, the large amount of video data requires the use of the external SRAM on the Spartan-3 Starter Board. A VGA generator is implemented on the Spartan-3 Starter Board, and the output is displayed on a color monitor through the VGA port (Figure 3).

Copyright ©2005, American Society for Engineering Education

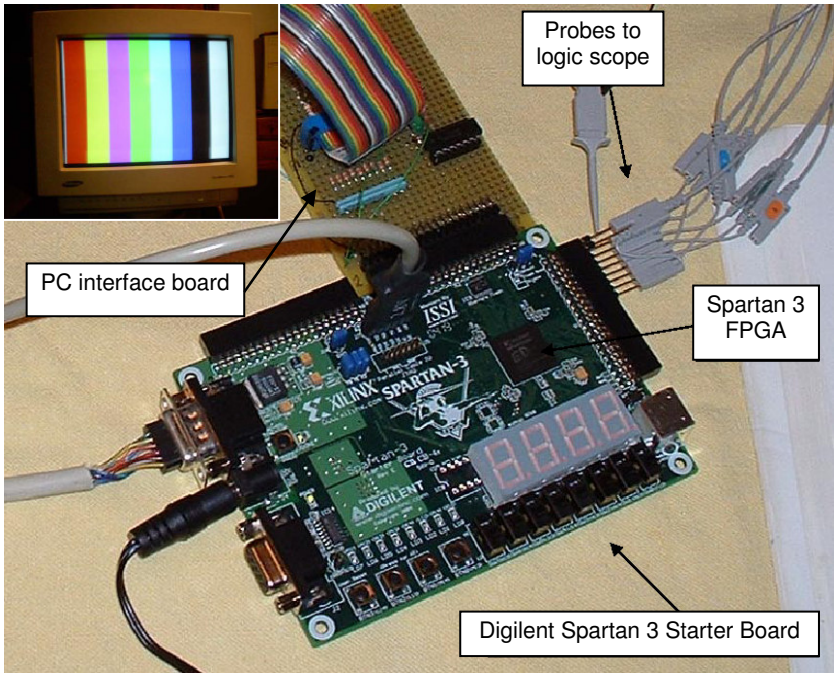

Fig. 3. Video processing board.

The design modules consist of a parallel port input, a memory controller, a VGA output, a video decoder, and a clock generator.

The parallel port interface connects to a PC to import images to the buffer. A program on the PC converts an image file to a data stream out the parallel port. The port is 8-bit wide with additional handshaking lines. The input module takes in the data and stores it in the image buffer. Counters generate the address for the memory controller.

The memory holds the video image. It accepts data from the input module and supplies data to the VGA output module. Memory access is shared between other modules. The controller assigns priorities to each access to keep the important data moving.

The VGA section pulls data from the memory and scans it out to the display. It counts the pixel location, horizontal and vertical, on screen to address the correct pixel location in memory. The data from the memory is buffered then serialized to the output pins. External resistors create an analog value to drive the RGB lines. The VGA section generates the horizontal and vertical sync to the monitor.

A decoder converts video into a digital data stream. The module decodes the data, assigning each pixel to a row and column address and converting the data to RGB.

The clock generator divides the 50 MHz oscillator to 25 MHz pixel and system clock. This is close enough to the standard 25.175 MHz VGA clock.

Copyright ©2005, American Society for Engineering Education

Number of Slices	275 out of 1920	14%
Number of Slice Flip Flops	330 out of 3840	8%
Number of 4 input LUTs	417 out of 3840	10%
Number of bonded IOBs	87 out of 173	50%
Number of GCLKs	1 out of 8	12%

Table 1. Synthesis results of the video processor design.

Table 1 gives the synthesis results of a particular design to show the scale of the FPGA implementation. The design used only a small portion of the Spartan-3 FPGA internal resources.

(2) Video Game: Ping-Pang

This project implements a video Ping-Pong game on the D2SB-DIO4 Combo Board with Xilinx Spartan 2E FPGA as the target chip. The DIO4 peripheral board is equipped with switches, push buttons, and a VGA mount. Their pins are hardwired to the pins on the FPGA. The push buttons serve the ball and reset the score, the switches control the bat movement, and the visual effect and game scoring are shown on the VGA monitor.

The design involves the two-dimensional game logic and the interface to a VGA monitor. The game is displayed on the monitor by supplying data in real time. The VHDL modules include random number generator, game controller, and VGA interface. The random number generator initiates the ball movement. The game controller directs the bat movement in response to the switch change, calculates the bounce route, and starts, stops, pauses, and scores for the game. The VGA interface module determines VGA resolution, refresh rate, color display, and provides horizontal and vertical timing signals for video.

(3) Advanced Encryption

This project implements an advanced encryption standard, the Rijndael algorithm [5], on the Xilinx Spartan 2E FPGA. Networks require advanced encryption standards to satisfy their security needs. While software implementation of the Rijndael algorithm offers flexibility and low cost, FPGA implementation provides higher speed and better performance.

This project involves the design, testing and implementation of an encryptor/decryptor. The encryptor and the decryptor exist as separate blocks on the same chip. This is essential in cases where concurrency is important. The VHDL modules consist of Byte Substitutor, Row Shifter, Column Mixer, Round Key Adder, and Key Expander. There is also a controller in the design to control these modules and round operations. The initial data is included in the VHDL code, and the encrypted and decrypted data is shown as consecutive digits on the seven-segment displays.

Copyright ©2005, American Society for Engineering Education

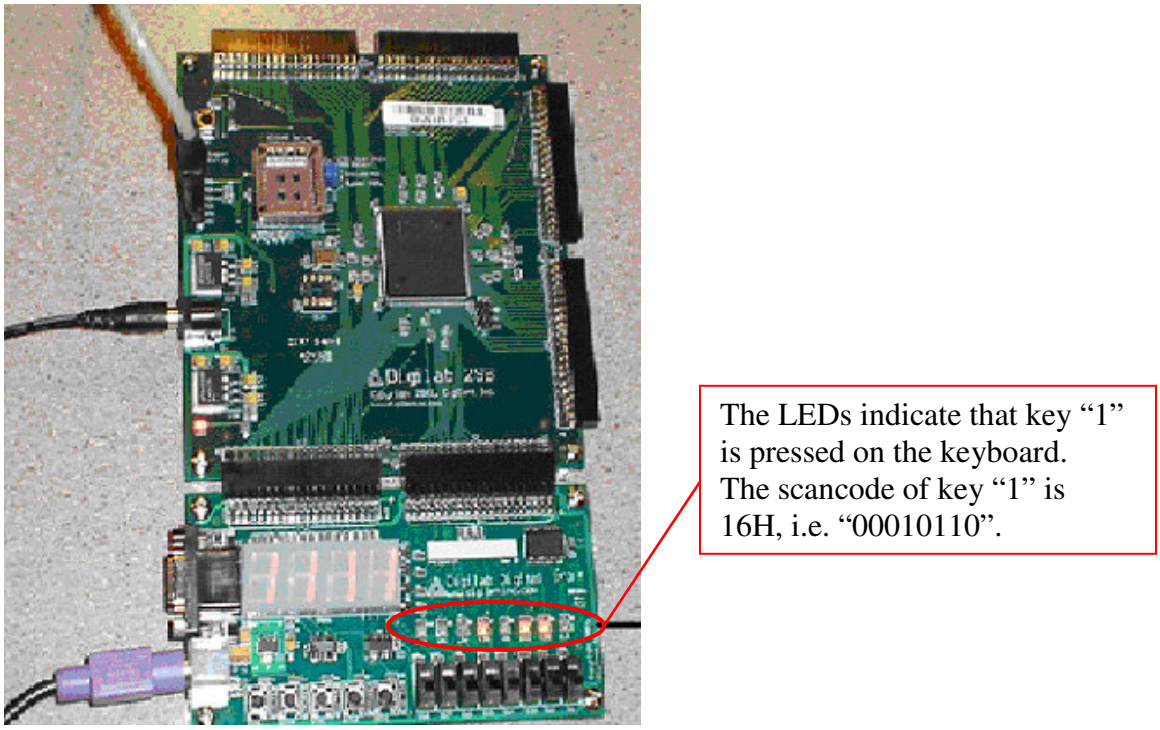

Fig. 4. Unidirectional PS/2 interface from keyboard to FPGA.

(4) PS/2 Encoder/Decoder

The PS/2 interface is used by many external devices such as the keyboard and the mouse. A PS/2 encoder/decoder is a two way handshaking technique that can be used to receive information from and send commands to a PS/2 device. This project implements a PS/2 interface for unidirectional data transfer from a keyboard to the Xilinx Spartan 2E FPGA on the D2SB-DIO4 Combo Board.

PS/2 keyboards use scancodes to communicate key press data. The scancode is displayed as a binary number using the eight LEDs on the DIO4 board (Figure 4). When a numerical key (0-9) is pressed, the number is retrieved and shown on the seven-segment displays. When any other key is pressed, an “E” is shown. An internal register queue is used to buffer up to 16 scancodes. One can request that a previous keyboard press be displayed using an “undo” switch on board.

The keyboard interface modules consist of Keyboard Verification, Scancode Buffer, Parity Logic, and Reader Logic. The Keyboard Verification periodically verifies if a PS/2 keyboard is present and if keys are pressed or released. The Scancode Buffer converts serial data from the PS/2 port to bytes and provides a register queue to store and shift them out. The Parity Logic checks the parity bit of each serial data frame. The Reader Logic starts reading as soon as there is data in the queue, and continues until the queue is empty. It displays the binary scancodes on the LEDs and the corresponding

Copyright ©2005, American Society for Engineering Education

numbers or "E"s on the displays. Other issues that need to be taken into account include the debounce handling and the delays for the PS/2 signals.

Summary

Class assignments and team projects using FPGA-based development boards have been developed for use in a graduate level ASIC design course. These exercises introduced students to design methodology of microelectronic systems, as well as familiarized them with the FPGA implementation techniques.

The assignments and projects are organized in accordance with the layout of the course lectures. They are built upon concepts and examples given in class, thus facilitating comprehension and enhancing the learning experience.

Acknowledgements

Development of the ASIC design course at Western Michigan University was made possible by the equipment donations from Xilinx, Inc. and Digilent Inc. through the Xilinx University Program.

References

1. Smith, M.J.S., Application-Specific Integrated Circuits, Addison-Wesley Professional, 1997
2. Yalamanchili, S., Introductory VHDL: From Simulation to Synthesis, Prentice Hall, 2001
3. Zwolinski, M., Digital System Design with VHDL, Prentice Hall, 2000
4. Chang, K.C., Digital Systems Design with VHDL and Synthesis: An Integrated Approach, Wiley-IEEE Computer Society Press, 1999
5. Daemen, J. and Rijmen, V., The Design of Rijndael: AES – The Advanced Encryption Standard, Springer-Verlag, 2002

LIANG DONG
Dr. Dong currently serves as an Assistant Professor of Electrical and Computer Engineering at Western Michigan University. His research interests include integrated circuit for communications, array signal processing for communications, channel estimation and modeling, space-time coding, and wireless networking.

MASSOOD Z. ATASHBAR
Dr. Atashbar currently serves as an Assistant Professor of Electrical and Computer Engineering at Western Michigan University. His research interests include physical and chemical microsensors development, wireless sensors, and applications of nanotechnology in sensors, digital electronics, advanced signal processing, and engineering education.

Copyright ©2005, American Society for Engineering Education