

Development of Microcontroller/ FPGA-based systems*

A. H. G. AL-DHAHER

*School of Information Technology and Engineering, Faculty of Engineering, University of Ottawa,
800 King Edward Avenue, Ottawa, Ontario, Canada K1N 6N5. E-mail: dhaher@site.uottawa.ca*

Microcontrollers and field programmable gate arrays (FPGAs) both are widely used in digital system design. Microcontroller-based instruments are becoming increasingly widespread. On one hand, their high speed, power and falling prices make them an obvious choice. On the other hand, the fast growing popularity of FPGAs, the availability of powerful development tools, and the increase in speed and high density have made FPGA-based systems an alternative choice. Due to the importance of both technologies to undergraduate students, this paper discusses the importance of both technologies and presents the experience in teaching both. The design, implementation and testing of a typical project completed, based on both a microcontroller and a FPGA is presented. Students' reactions, encountered problems and skills gained are also reported.

INTRODUCTION

TRADITIONALLY, undergraduate digital design courses are taught by presenting the concepts in the lecture and further enforcing these concepts in a lab. The purpose of the lab is usually to build a prototype circuit based on schematics. Students usually have no problems handling a small circuit that requires a handful of components; however, the problems arise when the circuit becomes more complicated. To speed up the prototyping process, teaching laboratories usually use breadboards, and in some cases, wire-wrapped breadboards. This technique introduces noise as well as invisible and hard-to-trace bad contacts. These imperfections cause great frustration as a result. Moreover, students find it difficult to develop and debug more substantial logic circuits, especially those involving medium and large-scale integration devices. Consequently students end up placing more emphasis on connection debugging techniques rather than on the actual digital design [1]. Therefore, there is a need for development and debugging technologies that allow students to focus on design rather than physical implementation.

As the technologies advance and the demand for more capabilities increases, the microprocessor plays a more important role in advancing digital design. The importance of microcontrollers and their wide use in different instruments has lead many universities to offer at least one course in microcontrollers to undergraduate students of electrical and/or computer engineering and related disciplines [2]. The ability to view a hardware/software system as an integrated whole is becoming a crucial aspect of an engineering student's education. Courses offered in universities covering

both hardware and software are very popular among students. These courses cover basic principles of microcontrollers and interfacing of external peripherals.

The increasing capabilities, speed and significant decrease in cost have contributed to the wide use of microcontrollers. Recently, microcontrollers have been embedded in intelligent sensors. There has been a steady increase in the use of intelligent sensors and microcontrollers in the areas of automation, security, hazard detection and comfort. Dealing with the microcontroller itself is not a concern. The problems arise when the microcontroller needs to be interfaced with the outside world. In an academic environment, the time and resources given to a student are limited. Therefore, interfacing must be done in a fast, prototype method, resulting in wiring problems. In contrast, the high tech sector does not face these problems because it is not limited by time and resources. Due to this limitation, universities have to find an alternative to cope with this issue.

Designing with FPGAs eliminates the need for debugging large wire-wrapped circuits [3]. The FPGA allows the designer to view the design from a system level. Designing with FPGAs usually results in more complex projects that are completed in a shorter period of time. A sizable design project may be completed in one academic semester. Implementing the design requires little knowledge of the FPGA architecture and implementation programs. Designing digital systems using FPGAs may be performed easily and quickly on a desktop computer [4]. Designs may also be changed rapidly and efficiently, and students will get an immediate feedback on changes made [5]. To develop a sophisticated system, students need to model real designs, simulate, and then verify their correctness. The flexibility of the architecture and the simplicity of the design method have

* Accepted 28 September 2003.

encouraged universities to move towards the use of FPGA technology. Academic communities are finding that FPGA technology is significantly changing the way traditional logic design has been taught. The increasing popularity of field programmable gated array (FPGA) technology has led many universities to adopt them, and to a large extent, replace TTL technology. Among many of their advantages, FPGAs offer a good solution to previously mentioned problems, making them a suitable choice for teaching digital design and computer architecture courses [6, 7].

Available development tools and packages are becoming very sophisticated and highly automated. Proper development tools and equipment offer full capabilities for the development of working prototypes. The choice of design entry depends on the preference of the designer and the application. The design may be entered using different methods including schematics and waveform entries. Many development systems provide a hardware description language. This allows the design to be entered in text format expressing Boolean equations, truth tables or state machines. For many applications, the very high-speed integrated circuit hardware description language (VHDL) will provide the most efficient description of the design for accommodating future changes and documentation [8–9].

VHDL synthesis and simulation provide a new tool for computer architecture classes [10–11]. Certain development systems allow complete simulation, which include timing waveform analysis of the resulting design. Functional and timing verification are typically performed using simulation and in-circuit testing. Report files and static timing analyzers are also available. FPGA simulation gives the designer confidence in the design before programming the device. When programming, the compiler configures the designated FPGA by specifying the internal connections through logic synthesis. Also, due to their widespread use and recent popularity, many FPGA vendors offer training material to university students at little or no cost. This material includes tutorials and training kits. These kits not only help students enormously during the training phase, but they also significantly simplify the teaching process.

FPGA technology has not yet replaced microcontrollers. As a matter of fact, for many years, microprocessors, and lately microcontrollers, continue to have a dominant presence in the industry and academic community. For the last decade FPGAs have demonstrated a significant amount of growth. In many instances, one wonders whether to favor one technology over the other, and what the students' reactions to this choice would be. One would also be concerned about the possible problems that are likely to be encountered.

This paper presents the experience gained in teaching both technologies to electrical and

computer engineering undergraduate students. Both technologies have been taught at several academic and technical institutions for many years and have been reported in the literature. This paper is not intended to introduce the FPGA technology to the engineering curriculum. This paper is also not meant to duplicate material or training offered by vendors. The purpose of this paper is to show the importance of teaching both technologies to undergraduate students. Due to the advantages of FPGAs and the flexibility of microcontrollers, it is essential to offer both technologies in undergraduate courses. An example of the development of an integrated circuit error detector is used to demonstrate the students' reaction toward these technologies. In both cases, students learned extensively about the design and implementation of these systems. From a teaching perspective, both technologies are equally important and beneficial to students.

MICROCONTROLLERS AND FPGAs

Modern microcontrollers are becoming complex, powerful and extremely efficient. These microcontrollers allow designers to incorporate many hardware functions on a single chip. Although high performance microprocessors are used in powerful computers, microcontrollers are embedded in many products, such as air bags, car engine controls, appliances, audio, video, telephones, fax machines, modems and many more devices. More than forty microcontrollers are located in homes [12]. In addition to the CPU, microcontrollers contain other modules, such as communication, data conversion, and timing functions. The timing module is quite complex and sophisticated. It allows for the creation and measurement of waveforms. Thus, the use of microcontrollers help designers improve the overall performance and reduce the cost of the system. The selection of a specific microcontroller and its associated circuits depends mainly on the required performance, complexity and flexibility of the system [13]. Software development systems used for microcontrollers allow programmers to code, debug and test the software. The language used may be either an assembly language or a high-level language such as C. The program may be tested on the development system. If at that point it executes correctly on the development system it may then be loaded into the memory of the microcontroller.

In some applications, the microcontroller timer may not be fast enough to meet system requirements. In such instances the timer module is not fast enough for rapid signal changing. The designer must then choose alternatives to create or measure timing waveforms, thus adding circuitry to implement these specific functions. Due to their high speed and flexibility, FPGAs often offer a viable choice for implementing such functions [14].

The selection of the best implementation for a design requires skillful knowledge of both microcontrollers and FPGAs. Systems based on these devices are usually developed by programming, which then allows the designers to modify during the design and implementation phases. The development is usually performed using more suitable development systems.

FPGAs are often used as co-processors for existing processors, accelerating time-critical operations [15] or extending operations of the standard ALU [16]. For example, with microcontrollers computing expressions with more than two terms requires a sequence of arithmetic and/or logical operations, and in many cases, there is a need for assigning a temporary variable. FPGAs can increase the performance of such systems by performing such computations in parallel with a reasonable amount of combinational logic. They are also largely used for custom-computing systems in which a whole computational device is customized and implemented in a FPGA.

Single chip microcontrollers are often used and are sufficient for specific application requirements. In this case, only the software has to be developed for the required application. In developing such systems, hardware interfacing is required, which is usually implemented using standard specialized chips. This leads to a fixed system, which cannot be altered without hardware redesign. FPGAs are very versatile in such needs [17, 18].

The increasing densities of FPGAs and the decrease in cost allow for the integration of embedded systems in a single FPGA device. This reduces the system's development time for hardware. For microcontrollers, software development compatibility should be considered. When a microcontroller application is being developed, most of the time is spent on software development. Unfortunately, the cost associated with using a FPGA to replace the microcontroller in special-purpose embedded systems is considerable. FPGAs may be more expensive than a general-purpose microcontroller. Again, this depends on the size and the quality involved. Another concern is the time required to learn VHDL and the development tools. Microcontrollers, having been around for longer than FPGAs, have well-developed tools and are familiar to many. Nevertheless, the present development tools for FPG-based systems are very powerful, sophisticated and easy to use. Students who have experience in developing software for microcontrollers and have knowledge of digital systems will find no problem adapting to FPGA development tools. To this effect, FPGAs are introduced by giving projects to students and hence the objectives are met in the way of learning by doing.

PROJECT LABORATORY

The project course laboratory is a senior level course that is offered to students before their

graduation. The course is one semester in length. Running this project course for many years resulted in several improvements and extensive knowledge was gained. The design project is the last course that students are required to take in the series of courses and labs.

Enrolment and students' backgrounds

Students enrolled in the engineering programs specialize in different disciplines. These disciplines consist of Computer Engineering, Electrical Engineering and Software Engineering. By the time they come to a senior project laboratory, the students have all completed a course in digital design, which includes basic logic circuits, minimization, sequential and combinational circuits and state machines. They may have also completed a course in VLSI, which would have given them some basic knowledge of VHDL. The microcontroller course they completed provides sufficient knowledge of software and hardware. The lab for the microcontroller course has several experiments on basic assembly language programming, structure assembly language, basic interfacing and interrupt handling [13].

In addition to these directly related courses, students, depending on their major, may have completed courses in high-level language programming (C/C++ and Java), operating systems and other computer science courses. Students enrolled in the engineering program gained quite a background in basic digital circuits design, as was previously mentioned. They started out by designing simple digital systems using TTL chips laid out on a breadboard and then transitioned over to the microcontroller. The transition provided them with in-depth experience in the design and implementation of microcontroller-based systems [13].

Because of the advantages and versatility of the FPGAs, students were offered this technology in their senior project laboratory. The projects offered were all microcontroller and/or FPGA oriented. The projects aimed to use all the of the knowledge students gained in previous courses.

Project laboratory organization

In this design project course, a team of three to five students worked together to complete a selected project within one semester. Students selected their project from options offered to them at the beginning of the semester. They were scheduled to meet twice a week for a period of three hours. It was found that students usually ended up spending much more than the allocated time. The project was conducted with limited supervision. By the completion of the project, students were required to present their working project with an oral presentation and by submitting a formal report. The report was intended to include all of their performed work; therefore, students spent a great deal of time producing high quality reports and projects.

PROJECT DESIGN EXAMPLES

To offer students the knowledge of microcontrollers and FPGAs, the project selected was to be developed using both technologies. Among the many projects performed, one example was to build an IC tester using either a microcontroller or a FPGA. This project was selected as an educational model because it provided immediate results. The following sections present the two approaches of implementing the project.

Microcontroller-based IC tester

The system was developed using general-purpose equipment to avoid accumulating additional cost and delay time. A personal computer and a general-purpose microcontroller were the main items in this development. Fig. 1 shows the tester components. The graphical user interface was developed on a host PC operating on the MS Windows 98 operating system. The user can select the chip to be tested from a list of devices and may choose to test a combination of inputs, or the user may choose to perform a full test which includes all of the possible input combinations. A window showing the output status was to display whether or not the test was successful. The graphical user interface was developed using the Visual Basic language. This was selected for its ease of use and suitability for developing rapid prototypes.

Hardware

The hardware exerciser was developed on a Motorola MC68HC12 microcontroller. The hardware unit was a compact 2.25-inch by 3.25-inch printed circuit board built around a general-purpose Motorola microcontroller type

MC68HC812A4. The board has all the timing circuitry and it operated on 3–5 volts. The Motorola MC68HC12 family of microcontrollers is a highly integrated, general-purpose family. Motorola makes different versions; the most widely used ones are the MC68HC812A4 and MC68HC912B32. These two versions are compatible in their instruction set, but differ in some features, such as on-chip peripherals and on-chip available memory [19–20]. The MC68HC812A4 chosen for this project was a 16-bit architecture, non-multiplexed 8- or 16-data, and 16-bit address bus. It has a bus speed of 8 MHz using 16 MHz crystals. It also has an on-chip memory of 4K EEPROM and 1K SRAM. In addition, the microcontroller board has two asynchronous serial ports (SCIs) and one synchronous serial port (SPI). The MC68HC812A4 is a programmable microcontroller, making it very versatile and offering flexibility in the implementation. All functions are controlled by software loaded to the microcontroller. Details of programming the microcontroller may be found in the Motorola documentation [21–22].

Software

The software developed for this project was quite sizable and complex. To facilitate this development another board (MC68HC912B32) was used. This board has on-chip memory of 768 bytes of EEPROM, 1 KRAM and 32 K flash memory. The board has a monitor/debugger program (D-Bug12), which resides on the 32 K flash memory. The board was connected to a PC using a serial interface. The HyperTerminal communication program available with MS Windows 98 was used to access the program. The D-Bug12 was designed to help with the developing and

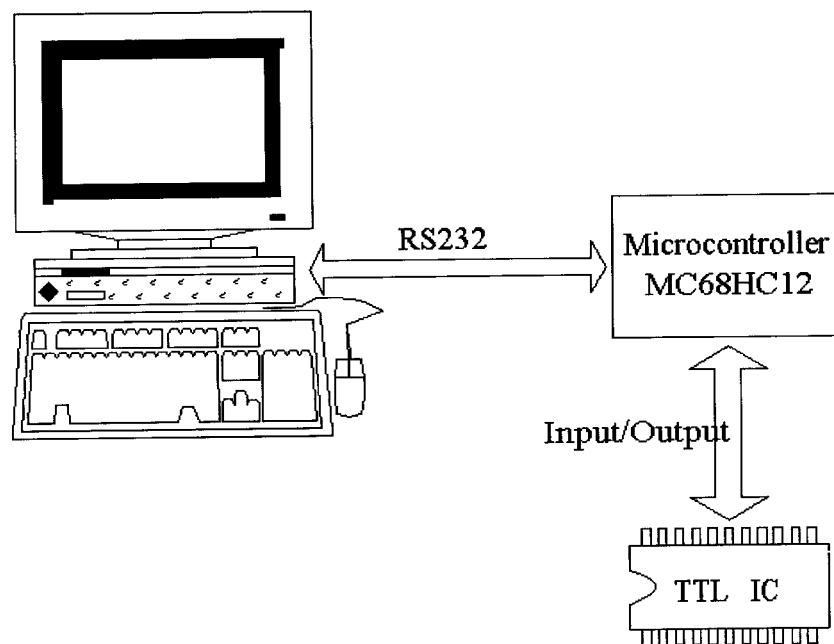


Fig. 1. Schematic of microcontroller-based digital IC tester.

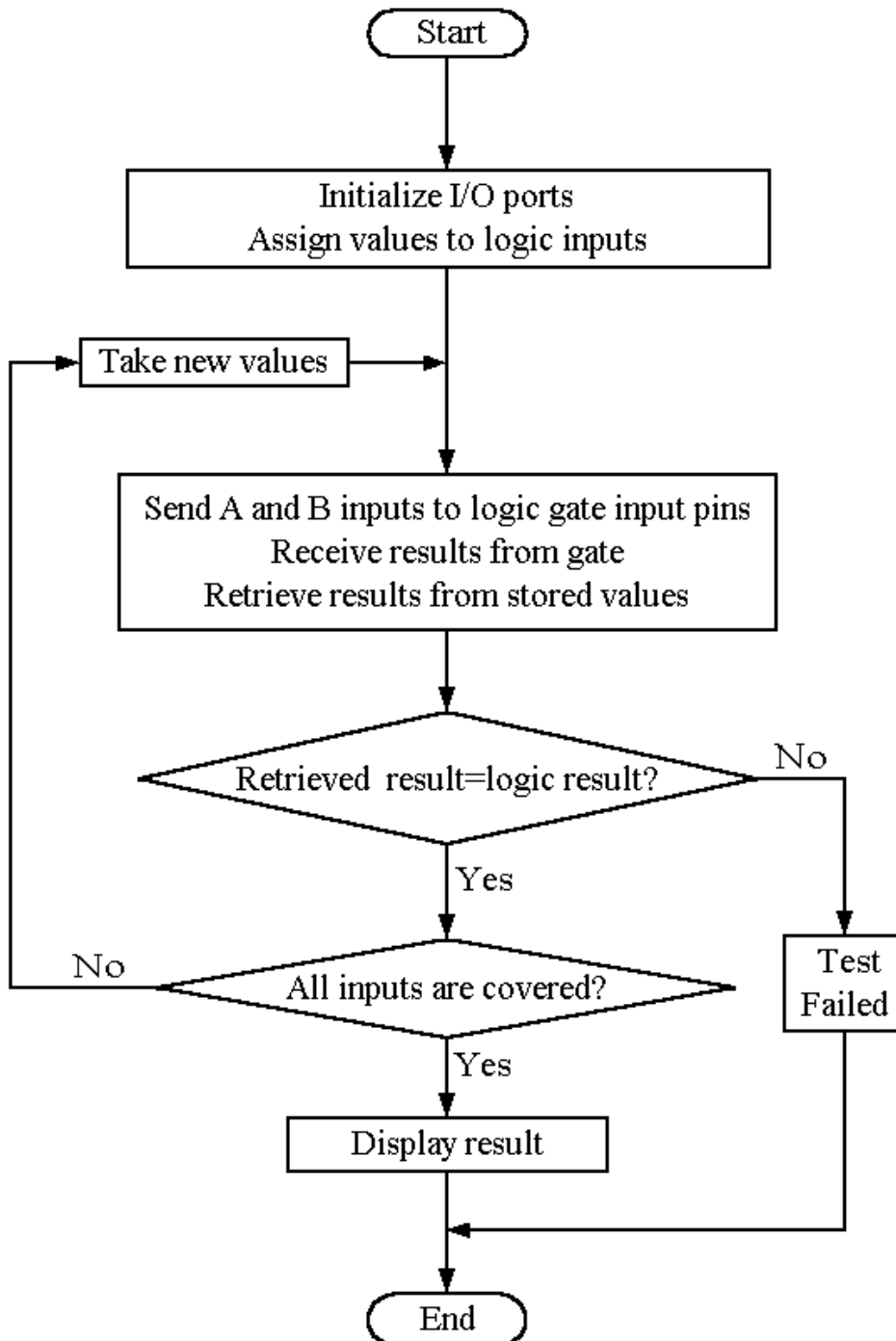


Fig. 2. Block diagram of algorithm used for testing digital ICs.

debugging of software. It has a line assembler and commands to modify the contents of the memory. It also allows for stepping through memory and displaying contents of one or more memory locations. Another advantage of the D-Bug12 is that it has several routines the user may use to develop assembly language. This board was used for developing the program only, but because of its limited EEPROM space (only 768 bytes), the program was loaded to the first board. This approach was adopted to utilize these cost effective boards, which were used in the teaching laboratory of the

microcontroller course. The software was developed in assembly language using WinIDE Integrated Development Environment assembler supplied by P&E Microcomputer Systems, and it resides on the PC. After debugging and fixing all the errors, the program was loaded to the EEPROM on the board.

Operating and testing

Using this system to conduct a test involved several steps. First, from the graphical user interface, the user chooses the IC and selects which function will

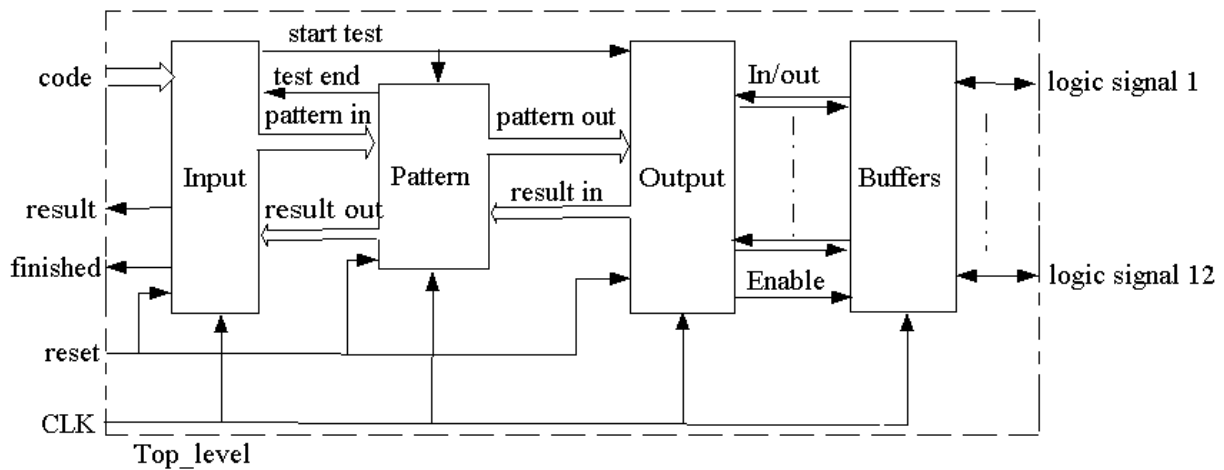


Fig. 3. FPGA-based test unit.

be tested in the current test session. An exhaustive test may be selected, in which case every function will be tested with all possible input combinations. Second, the test information determined from the GUI was transmitted to the Motorola MC68HC12 microcontroller. The MC68HC12 applies logic inputs using I/O ports to the device to be tested. The MC68HC12 then reads the output from the device. By comparing this output to the truth table stored in the microcontroller memory, the system then decides whether the test was a pass or fail. Results of the test are then displayed on the graphical user interface.

Different types of logic devices were tested on the system. Two-input and three-input logic gates (twins and threins) were tested. Faulty gates were mixed with functional ones and the system was able to identify the two groups successfully. The test takes into consideration all different combinations of inputs and identifies the faulty gate within the chip. Figure 2 shows stages of the algorithm of testing adopted. An arithmetic logic unit (ALU) was also tested and the system proved to be successful in testing this type of integrated circuits. With the arithmetic logic unit, the user has the choice of testing single or combinations of functions offered in this device.

FPGA-based IC tester

This system consisted of three separate components: the software controller was based on a PC operating MS Windows 98. Test unit interfacing with the PC was achieved by using the parallel port of the PC and digital test platform connected to the FPGA unit using bi-directional I/O pins.

Software controller

Interactions between different parts of the system were guided by subroutines included in the software controller. The integrated circuit under test was identified by a certain code. The FPGA completed the test for all the gates and finally, the software retrieved the result from the FPGA, interpreted the code and displayed the results.

A friendly graphical user interface was developed. By selecting a chip from the displayed list, an electrical connection diagram of that particular chip appeared with leads connecting to various pins. When in the idle state, all pin lights turn to the color blue. A reset button was used to reset the FPGA to make the system ready to perform the test. Selecting the test button started the test sequence and changes the pin light colors to reflect the chip's functionality. A red pin indicated that there was an error associated with that pin/gate output, and a green pin indicated a fully functional pin.

FPGA-based test unit

This unit is based on the Altera UP1 Education Board. This board is moderate in cost and is supplied by Altera as a training package. The UP1 CPLD education board includes both a MAX EPM7128S EEPROM-based CPLD, and a FLEX EPF10K20 SRAM-based CPLD. This board also includes several LEDs, four seven-segment displays, three dual-in-line package switches and a 25.175-MHz crystal oscillator. In addition, it has a PS/2 mouse or keyboard connector and a video graphics adapter (VGA) connector. The logic devices are programmed through the PC's parallel port connected via the ByteBlaster download cable.

The early design was configured on the EPM7128S CPLD, resulting in a problem due to the code size being too large for the CPLD. Also, the EPM7128S does not have internal tri-state buffers; these buffers have to be implemented with complex external circuitry on the breadboard to allow for bi-directional buses. The importance of expandability was also an issue. Later designs used the FLEX EPF10K20 chip, providing the necessary functionality for more advanced designs. More importantly, the design allowed for the implementation of internal tri-state buffers.

The top-level design implemented in the FPGA is shown in Fig. 3. All the blocks were designed in VHDL. The top-level module is responsible for

communicating with the outside world. All input/output ports declared in this module were assigned a physical I/O port on the FPGA. The input blocks were used to control the communication between the FLEX 10K chip and the software controller. The two input signals received from the PC were the reset signal and the code signal. When the reset signal was low, the blocks processed on every rising edge of the input clock which was assigned to the global clock input from a 25.175 MHz crystal oscillator on the UP1 Board.

The pattern block is used to generate a pattern of input values to the chip to be tested based on the received code. Finally, the output block was used to properly initialize the bi-directional bus to the TTL test platform. The test platform was interfaced to the output block using the bi-directional bus. The output block enabled the buffer to act as an input or an output.

Test unit platform

The hardware consisted of a test bench and the FPGA. In addition to having the IC physically planted on the board test bench, several logic buffers were placed between the parallel port pins and the FPGA pins to limit the amount of current flowing between the two devices while boosting the desired voltage levels (usually from 3.7 volts to 4.2 volts). Also several light emitting diodes (LEDs) were placed after the parallel port for verification purposes.

Operating and testing

An operation was initiated by selecting a chip and then selecting the test button. This made the reset logic low and sent the IC code signal. For every rising clock edge, the input block checked that the entered code for a valid code, which was a representation of the selected IC. If a valid code was found, a pattern of 8 bits was sent to the output block, which represented one state of the truth table of the tested IC. The IC was tested for all combinations of inputs. The input signal to the pins were held for a sufficient amount of time so that the rise and fall times of the TTL were preserved. The result for a current input combination was read from the TTL output pins through the output block. Results from the test TTL were compared to the 8-bit pattern and an error was recorded for any mismatch.

Moreover, on every rising clock edge, the input block probed the test_end signals from the output block. If the test_end signal was detected, the result of the test was placed on the result_out signal to the PC, and the finished signal was toggled high.

Different types of digital ICs were tested on this system. ICs with 2 inputs and 3 inputs were considered. Results were very consistent and reproducible. Damaged ICs were mixed with fault-free ones and were given to the system. The system accurately differentiated between the two categories.

STUDENT PERFORMANCE AND PROJECT LABORATORY EVALUATION

Difficulties faced and adopted solution

Analyzing the difficulties students faced during the project course and then implementing solutions to remedy those difficulties were key in improving the project course. Prior to the project course, all the courses and laboratories that students completed were structured with regular lab hours and clear lab assignments. During these labs, students were expected to finish the lab experiment in the time allocated and submit a lab report. Teaching assistants helped the students immensely throughout their laboratories and courses. Contrastingly, this project course is totally different and in turn, students were forced to depend on themselves and each other.

Forming teams among students presented one of the first problems most students faced in this course. Different time schedules, backgrounds and motivation levels were some of the factors that contributed to this problem. As a solution, students were encouraged to form their teams prior to the start of the project course based on their experiences, skills and abilities. Team members were encouraged to know their team members before the group formation and to choose their groups wisely. Also, a team leader was chosen as a representative of the group. The team leader was responsible for ensuring that deadlines were met, work was being carried out effectively and that conflicts were resolved within the group. Working in a group environment where each student's experience and background differ from the rest of the team was a concern raised by most students. Therefore, it was recommended that each team meets and discusses each team member's background and skills and then acknowledges each student's personal learning goals. Based on group discussions, tasks were divided between students and completed tasks were reported amongst group members upon completion.

Finding time to meet was another problem that students faced, given their hectic schedules. Once a team leader was chosen and the tasks were divided the leader was encouraged to set up a weekly or bi-weekly meeting session with the rest of the group to discuss the project's progress. Difficulties encountered and suggested solutions were discussed in the meeting. These sessions gave the team members a chance to bring forth innovative and challenging ideas to each other. They allowed the students to exercise the problem-solving skills that they acquired over the past three years at University. Moreover, each student was encouraged to work at home at his/her own pace, since software development and design could easily be done from any PC with the required development tools.

Learning the development tools was another problem that students have reported. As a solution,

teams were given the opportunity to learn the development tools prior to the start of the actual project. A comprehensive tutorial was provided to each student that allowed him or her to learn every aspect of the MAX+PLUS II environment. Moreover, the students were given a library full of resources such as books, manuals, journal papers and Internet sites, all there to help address any of their hardware and/or software concerns.

Knowing where and how to start a project may be the most challenging problem at the beginning. Before diving full force into the project itself, students were encouraged to browse through previous projects that are stored on CDs for convenience, after they familiarized themselves with the development tools. The previous projects gave students an idea of the amount of work that needed to be done and the expected quality. Also, the previous projects were provided to all groups so that students need not re-invent the wheel when it came to the development of their work. Students were highly encouraged to use some of the developed components and build on them with their own innovative ideas.

Acquired skills

The microcontroller and FPGA technology offered students new design methods in digital systems. Learning the new digital design techniques proved to be a challenge that motivated students. The idea of learning by developing a prototype and functional systems attracted the students' interest. With limited backgrounds in VHDL, students were excited to gain more knowledge in more recent technologies. Within one semester, students managed to become accustomed to the development tools, learn VHDL, and design, develop and present a project.

The Altera development environment, including both software and hardware, was found to be very suitable for educational purposes. The student version, which was offered at no cost, helped students significantly. Most of the students' work was done outside the lab, making it easy for both the students and the teaching assistants. Students started the development by entering the design using the graphical editor in order to get acquainted with the development tool. The development of the project was carried out using VHDL, from which the students gained tremendous experience. Using this development method, students realized that the VHDL description had to be kept as simple as possible in order to develop the FPGA application with the available synthesis tools. The general reaction was very favorable, and the end result was very beneficial. Also, students learned a great deal about working as a group and managing their time effectively.

SUPERVISION OFFERED TO THE STUDENTS

The students conducted the projects under very limited supervision, which helped them depend on themselves and use each other as resources. The course was organized such that it was a lab intensive course. A weekly one-hour lecture was offered to assist the student's tackle any design or project-related issues with the supervisor. This weekly lecture also allowed administrative and technical problems to be resolved. Students performed some of their work in the laboratory and technical assistance was offered to them by the technical staff and teaching assistants.

Students' assessment and evaluation

At the beginning of the course, students were given a course outline, deadline schedules and guidelines on writing the progress report and final report. Even though students were given the freedom to work on their project at any time, the course was evaluated throughout the semester. During the third week into the semester, students were required to submit a project plan outline. This brief plan outlined the objective and the methodologies to be used. It was made clear to the students that the plan could and should change during the period of the project; therefore, the plan served its purpose in making the students focus on their project from the very start. Towards the middle of the semester, students were required to submit a progress report showing the performance and achievements up to date and discussing problems faced and solutions adopted up to that point. During the last week of the semester students were asked to demonstrate their working project. Moreover, at the end of the semester, a formal presentation was required in which all team members showed their contribution and work performed. The presentations were attended by the entire class and in the presence of the course professor, the teaching assistants and the technical staff.

The group was required to submit one formal report at the end of the semester that complied with the guidelines given to the students at the beginning of the course. This formal report was a comprehensive report that analyzed and discussed in detail the development and implementation of the project. Each report was to include a background section to show all the relevant research conducted. The report was to contain flow charts and diagrams explaining all of the algorithms developed. In order to prove their full understanding of their project and to show that they are indeed design engineers, the students were asked to introduce a section in the final report that discussed possible improvements and future enhancements. This section helped the students understand the complete cycle of a design project. Students are to submit the developed code as well

as a soft copy of the report on CD along with the hard version of the report.

CONCLUSION

The need for automation places an increased demand on the knowledge of new engineers. The use of microcontrollers and FPGAs is widely spreading and requires students to be familiar with and trained in both these technologies. The approach adopted in teaching these technologies proved to be a very effective one. Students

benefited a great deal from their experiences in the project course. The projects developed were applicable and their immediate results motivated students to pursue these technologies. The students showed a strong interest in this field, which is an encouraging indication of their potential success in the academic setting.

Acknowledgements—The author would like to thank Professor E. Petriu for his encouragement and support, and Mr. R. Montcalm and Mr. A. Stewart for their continuing technical support. The author would also like to thank all the students in this program and in particular those whose projects were taken as examples in this article.

REFERENCES

1. Steven K. Reinhardt, Integrating hardware and software concepts in a microprocessor-based system design lab, *IEEE Computer Society Technical Committee on Computer Architecture Newsletter*, (September 2000) pp. 49–55.
2. A. H. G. Al-Dhaher, Microprocessor Teaching, *Microprocessors and Microsystems*, **5**(10), pp. 455–471, 1981.
3. D. Lam, Educational use of field programmable gate arrays, field-programmable logic architectures, synthesis and applications, *4th International workshop on Field-Programmable Logic Applications FPL94, Proceedings*, Berlin, Germany, Springer-Verlag (1994), pp. 277–279.
4. J. O. Hamblen, Rapid prototyping using field programmable logic devices, *IEEE Micro*, **20**(3), 2000 pp. 29–37.
5. J. O. Hamblen, H. L. Owen, S. Yalamanchili and B. Dao, An undergraduate computer engineering rapid systems prototyping design laboratory, *IEEE Transactions on Education*, **42**(1), 1999, pp. 8–14.
6. G. M. Brown and N. Vrana, A computer architecture laboratory course using programmable logic, *IEEE Trans. Education*, **38**(2), 1995, pp. 118–125.
7. A. Trost, A. Zemva and M. Verderber, Prototyping hardware and software environment for teaching digital circuit design, *Int. J. Electrical Eng. Educ.*, **38**(4), 2001, pp. 368–378.
8. A. Sagahyoon, From AHPL to VHDL: a course in hardware description languages, *IEEE Trans. Educ.*, **43**(4), 2000, pp. 449–454.
9. E. A. Bezerra and M. P. Gough, A guide to migrating from microprocessor to FPGA coping with the support tool limitations, *Microprocessors and Microsystems*, **23**, 2000, pp. 561–572.
10. J. O. Hamblen, Using VHDL-based modeling, synthesis and simulation in an introductory computer architecture laboratory, *Int. J. Electrical Eng. Educ.*, **33**(3), 1996, pp. 251–260.
11. W. D. Richard, D. E. Taylor and D. M. Zar, A Capstone Computer Engineering Design Course, *IEEE Trans. Educ.*, **42**(4), 1999, pp. 288–294.
12. B. Martin-Del-Brio and C. Bernal-Ruiz, A software tool for teaching microcontroller system principles, *Int. J. Electrical Eng. Educ.*, **36**(4), 1999, pp. 279–286.
13. A. H. G. Al-Dhaher, Integrating Hardware and Software for the Development of Microcontroller-based Systems, *Microprocessors and Microsystems*, **25**, 2001, pp. 317–328.
14. A. E. Clapp and T. L. Harman, Combining microcontroller units and PLDs for best system design, *IEE Micro.*, April 1994, pp. 70–78.
15. R. Halverson and A. Lew, FPGAs for expression level parallel processing, *Microprocessors and Microsystems*, **19**(9), 1995, pp. 533–540.
16. V. Gustin, An FPGA extension to ALU functions, *Microprocessors and Microsystems*, **22**(9), 1999, pp. 501–508.
17. Z. Salcic, PROTOS—A microprocessor/FPGA-based prototyping system for embedded applications, *Microprocessor and Microsystems*, **21**, 1997, pp. 249–256.
18. S. A. Ito and L. Carro, A comparison of microcontrollers targeted to FPGA-based embedded applications, *13th Symposium on Integrated Circuits and Systems Design*, 18–24 Sept. 2000, Manaus, Brazil, IEEE Computer Soc., pp. 397–402.
19. J. W. Valvano, *Embedded Microcomputer Systems—Real Time Interfacing*, Brooks/Cole Thomson Learning (2000).
20. F. M. Cady and J. M. Sibigroth, *Software and Hardware Engineering—Motorola M68HC12*, Oxford University Press, (2000).
21. Motorola, *MC68HC812A4 Advance Information*, Document No. C68HC812A4/D Rev. 3, (1999).
22. Motorola, *68HC12, CP12 Reference Manual*, CPU12RM/AD (1997).

A. H. G. Al-Dhaher received his Ph.D. 1974 in electronics from the University of Kent at Canterbury, England. He then joined Kuwait University until 1990. In 1990 he moved to Canada where he established ACTA Technologies, a private software company that specialized in software development and infectious diseases diagnostic programs. In 2000 he joined the University of Ottawa as a member of staff in the School of Information Technologies and Engineering. His research interests include microcontroller-based systems, computer architectures and intelligent sensors.