A first year, VHDL based, digital electronics course

José Daniel Muñoz Frías and Sadot Alexandres Fernández
Escuela Técnica Superior de Ingeniería (ICAI)
Universidad Pontificia Comillas
Alberto Aguilera 23, Madrid, 28015
Email: {daniel,sadot}@dea.icai.upcomillas.es

***Abstract*—Most classical digital electronics textbooks have addressed the design with hardware description languages (HDL) as an addition to the traditional curricula. This is true, due that the design of digital circuits and the HDL specification are not taught simultaneously, but separated through time. Usually both are studied in different courses. The goal of this paper is to present a novel approach that simultaneously performs the introduction to digital electronics with the introduction to the specification of circuits using the VHDL language. In addition, the course is designed for the Telecommunications Program first year (Spring) with option in telematics, such that the previous knowledge of the students is reduced to a fundamental course in electrical circuits.**

I. Introduction

In Europe, especially in Spain, the Bolonia Process has been an opportunity to redesign the curricula of the engineering studies. In this contribution it is documented our experience on the implantation of a Digital Systems course in the first year of the Telecommunications Program of our School of Engineering.

In the industry, the majority of digital design now performed is based on programmable logic, because the reduction on board space, less power consumption and greater flexibility. Today, discrete logic has been relegated to some gates per board acting as "glue" logic. In addition, the design based on programmable logic is done using Hardware Description Languages (HDLs) like VHDL or Verilog.

However, at least in Spain, the majority of introductory Digital Design courses are based on discrete logic, wasting a high amount of time teaching circuits based on 74XX logic families. Normally, this introductory course is followed by an advanced course based on HDL using programmable logic. We think that this approach ends with a waste of time that is unacceptable, given that the number of lecture hours has been reduced with the Bolonia Process.

There are in the literature some articles describing Digital Design courses based in Hardware Description Languages. For example, in [1] is introduced a course in which VHLD is used. However, the language is only used to describe combinational circuits. Also, in [2] an introductory course based in Verilog language is presented.

In this paper we present a course developed to address this problem. The course teaches digital fundamentals using programmable logic and HDL languages (VHDL) instead of discrete logic. After four years of experience, we can say that the results are excellent.

In the next section, a description of the course contents, course material and lecture pedagogy is given. In Section III the laboratory assignments used to reinforce the concepts learn on the theory is introduced. The paper ends showing the student results in Section IV, the results of the surveys given to the students to evaluate the course in Section V, some ideas to adopt the course for other institutions on Section VI and finally, on Section VII some conclusions and further work.

II. Course development

The course is scheduled during the second semester of the first year of the Telecommunications Program. The 7.5 ECTS credits are divided in three theoretical sessions of one hour and one laboratory session of two hours. In this section lecture content is discussed first, then lecture pedagogy. The section ends with a discussion of the laboratory sessions taught during the course.

A. Lecture content

The course starts as any other basic course in digital electronics. The first lectures deal with an introduction to the digital world, where concepts like sampling and logic levels are explained. This introduction gives also some historical background about the evolution of the technology used to implement digital systems: from vacuum tubes to integrated circuits.

The next part of the course deals with Boolean Algebra, where the student learns how to manipulate logic expressions and how to simplify them using Karnaugh maps. After that, the course deals with number systems and digital codes. With these lectures, the student learns how to represent numbers in binary and how to perform mathematical operations in this system. However, given the introductory character of this course, only basic operations are studied: addition, subtraction and multiplication for unsigned and two's complement integer numbers. In addition to that, BCD[1] numbers are introduced, showing also how to perform addition and subtraction. However, the study of floating point numbers and his arithmetic is relegated to more advanced courses.

To end this first section of the course, an introduction to VHDL is given. This part begins with the motivation to use the hardware description languages: independence of the implementation technology, productivity of the designer, etc. Then, the typical design flow is presented and then we focus on the language, showing the basic file structure, basic data types and concurrent constructs; that is, the fundamentals to

[1]Binary Coded Decimal.

describe combinational circuits using the most common levels: structural and behavioural levels. After this introduction the student can describe in VHDL basic combinational circuits using the conditional assignment **`when ... else`** and the **`with ... select`** construct, more useful to implement truth tables.

After this introductory section, the course deals with the study of combinational systems. This section begins with the study of arithmetic circuits. Given the introductory nature of the course, we study only the binary adder with ripple carry and carry look ahead, the adder/subtractor using two's complement numbers and the combinational multiplier. The section ends showing BCD adders, in order to demonstrate that they need more hardware and, in consequence, it is normally more efficient the use of binary numbers when complex arithmetic operations are needed.

With regard of the introduction of VHDL, the arithmetic circuits allow the natural introduction of some constructs of the language that have not been shown in the introduction. The first construct is the **`for ... generate`**, that is introduced to specify an n–bit adder by instancing a one–bit adder n–times. This methodology allows the student to learn the language constructs associated to cases where these constructs are more appropriated. In addition, if a circuit can be described in various terms, we try to show both methods. Following with the adder example, after showing how to build it at low level instancing n one–bit adders, it is shown how to describe it using the + operator and the IEEE numeric package.

The combinational section of the course ends with the study of basic combinational blocks: multiplexers, demultiplexers, encoders, decoders and comparators. All circuits are designed in the first time using logic gates, then the same circuit is described in VHDL. In this way the student learns the use of hardware description languages as something natural.

The next part of the course deals with sequential circuits. The section starts studying the fundamental concepts as feedback, state, synchronism, etc.; and basic memory elements as latches and flip–flops. Then state machines and its VHDL specification are described. To do that, it is necessary to introduce new VHDL constructs: the **`process`** and the sequential statements **`if`** and **`case`**. As with the **`for ... generate`**, the introduction to the process and sequential statements is done here to show immediately how they are used to describe synchronous circuits. To end with this part, several state machines are studied. We begin with a simple edge detector and implement it using a Moore and a Mealy machines. These designs are done in the first time using logic gates and then the state machines are described in VHDL. Then, a sequence detector is studied to show the design of more complex machines.

The end of this part consist of the study of registers and counters. Like the rest of the course, this section begins showing how to build parallel and shift registers from flip–flops and then how to describe them in VHDL. Then, ascending and descending binary counters and modulo–n counters are shown. This section ends with the study of cascaded counters and arbitrary sequence counters.

The last part of the course consists of an introduction to semiconductor memories and the design of complex digital systems. The introduction to semiconductor memories begins studying the internal structure of SRAM, DRAM and ROM memories, showing how a bit is stored in each kind of technology and also how the decoding is performed. Then, we study how to specify those memories in VHDL and how to use the memories integrated in the FPGAs. Next, it is shown how to use a ROM to implement a combinational circuit by storing the truth table in the ROM, and how to describe this in VHDL. Lastly, in the design of complex digital systems, an emphasis is given in the division of the system in datapath and control circuits and in hierarchical design. This part is developed using some examples, like a parking barrier, a binary to BCD converter using the double–dabble algorithm and a SPI bus master.

B. Course material

Unfortunately, the majority of classical textbooks [3], [4] and [5] are heavily based on discrete logic. Some of them, like [5], offer some chapters on VHDL, but they are an "after thought", such that in our opinion, are not well integrated with the rest of the material. Others authors offer an advanced textbook to cover HDL design [6]. Finally, some textbooks could be appropriated to our approach, like [7], which introduces digital logic and HDL design. Unfortunately, this book was not available when we begin to teach this course and also is based on Verilog HDL, which is less used in our environment than VHDL. In addition, given that this course is taught to first year students, another requirement was that it must be in Spanish. However, the last editions of these books, which cover HDL design, are not translated yet. All of this have motivated the need to write a book which incorporated our course philosophy. This book is covered by a creative commons license (by–nc–sa) and available on–line on http://www.dea.icai.upcomillas.es/daniel/ApuntesED.pdf.

C. Lecture pedagogy

The theoretical sessions consist of lectures given to a reduced group of 35 student approximately. In these lectures the concepts shown in section II-A are taught using the blackboard for the explanation of the theoretical concepts and some slides to show the VHDL listings, which will require a lot of time to write on the blackboard.

At the end of each chapter, some exercises are proposed to the students. To encourage the resolution of this exercises by the students, a test consisting of one of the proposed exercises is done in the next theoretical session. This test takes only the first 15 minutes of the session. These tests are computed in the final grade as a 6 %.

III. Laboratory

The whole course is complemented with experimental sessions in a laboratory where the student designs digital systems using a powerful CAD system (Quartus II from Altera e.g.) and then implements a complete set of digital circuits using programmable logic devices (FPGA). The laboratory stations (see figure 1) are equipped with the standard electronics laboratory instrumentation (oscilloscope, function generator, power supply and a Personal Computer to execute the CAD system).

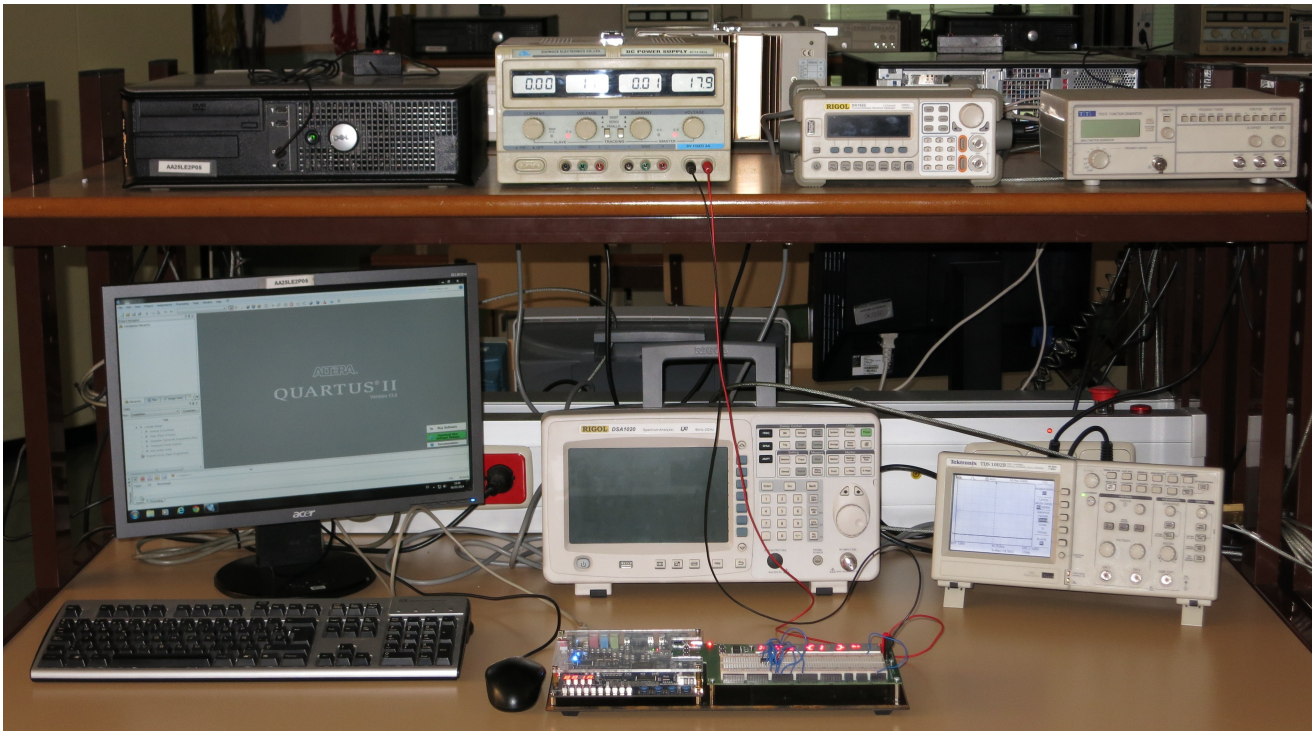

Fig. 1. Laboratory station

The laboratory part of the course consists of hands–on assignments which accompany the lectures described above. The goal of this lab is to illustrate the concepts discussed in class and to give students the opportunity to build, design and test real systems.

Laboratory experiments make use of the Altera–Quartus II–Modelsim tool–chain. This is a powerful state–of–the–art CAD system for designing and implementing digital systems on Field Programmable Gate Array based boards. This framework consists of an integrated set of tools that allows one to capture designs with schematic entry or to describe them with a Hardware Description Language (HDL). The design can be simulated in the computer and then implemented in the FPGA development board. The use of programmable logic devices takes away the tedious task of wiring individual gates on the insertion boards and allows one to concentrate on the creative part of designing the circuits. This approach also allows the design of systems with a greater complexity and, consequently, more near to the real circuits designed in the industry.

The lab assignments introduce gradually the tools and serve to illustrate the material covered in class. The first set of labs deals with combinational circuits and the second set introduces sequential circuits. In all experiments we focus on the test of the digital system as part of problem. This course uses an FPGA board from Terasic, which has an Altera Cyclone II chip on–board. Besides, the platform includes a extension board to experiment with additional circuits (see figure 2). The extension board allows the connection of all resources of the FPGA to additional external devices mounted in the breadboard. It has also an alphanumeric display that is used in an advanced course to show how time–multiplex display is done.

Labs are done in groups of two students in a two hours per week session, with group lab reports, predesign review and individual prelabs tests. In short, the goals of this first digital system laboratory course are focused on:

- To learn combinational and sequential digital systems design starting from a text specification. To do that, the student must apply the concepts and methods discussed in the class through hands–on projects.
- To understand modern CAD tools through hands–on experimentation with programmable logic design as well as the basics of VHDL.
- To analyse the results of logic and timing simulations and to use these simulation results to debug digital systems.
- To develop skills, techniques and learn state–of–the–art engineering tools (such as VHDL or Verilog) to design complex digital circuits.

Table I shows laboratory sessions and their sequence. As mentioned before, the laboratory has two major portions (combinational and sequential) therefore there are two objectives in all experiments, one related to the hardware design and the other related to the VHDL and testing. All these experiments are mandatory and each lab is followed by specially designed assignments to cover fundamental concepts:

- Lab 1 deals with the verification of the behaviour of logic gates using his truth table and with the familiarisation with digital integrated circuits. The lab

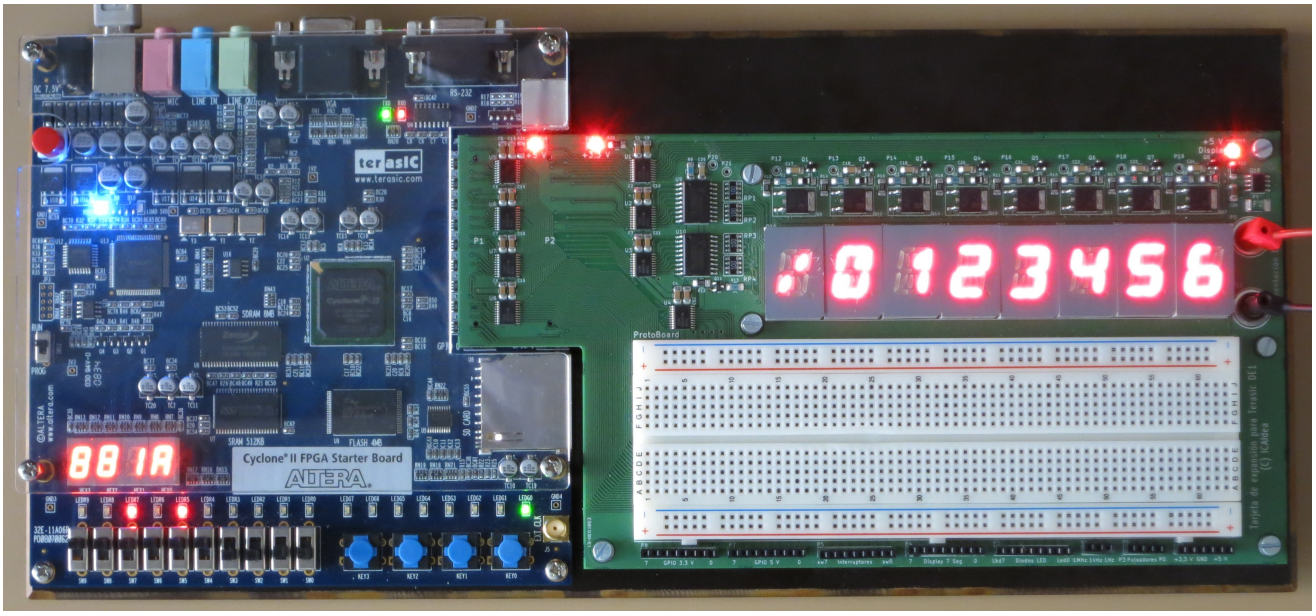

Fig. 2. Board used on the laboratory

TABLE I. DIGITAL SYSTEMS LABORATORY SYLLABUS

Session	Lab. Topic	Report (R) Predesign (PR) Previous test (PT)
1	Introduction to logic gates and the digital oscilloscope	(R)
2	Introduction to schematic capture and Quartus II compilation	(R) (PR)
3	Introduction to simulation and physical implementation with Quartus II	(R) (PR) (PT)
4	Combinational circuits design with VHDL	(R) (PR) (PT)
5	Arithmetic circuits. The 5–bit adder	(R) (PR) (PT)
6	Arithmetic circuits. The 5–bit multiplier	(R) (PR) (PT)
7	Arithmetic circuits. The 5–bit ALU	(R) (PR) (PT)
8	Introduction to flip–flops	(R) (PR) (PT)
9	Electronic lock	(R) (PR)
10	Parking control	(R) (PR)
11	Microwave oven timer	(R) (PR)

covers the timing and others technology characteristics like voltage levels. This lab is done using discrete logic, in order to measure easily the timing of the signals.

- Lab 2 is an introduction to hierarchical design methodology of digital logic circuits using CAD tools, covering: instantiation, hierarchical design and schematic entry.
- Lab 3 is an introduction to simulation and implementation, covering the following topics: top–down design, testbench and test–vector generation, test methodology and programmable logic implementation.
- Lab 4 deals with combinational design in through the implementation of basic blocks like multiplexers, decoders and interfaces with I/O components. This session covers the full basic design flow with VHDL.
- Lab 5, 6 and 7 deals with the implementation of basic arithmetic blocks (adder, multiplier and ALU design). These sessions cover the introduction to behavioural modelling through implementation of main combinational blocks including, simulation and hardware verification.
- Lab 8 is an introduction to the concepts of fundamental sequential circuits: latches and flip–flops. This lab is based on discrete logic to measure easily the timing, clocking and technology characteristics of those sequential circuits.
- Finally, labs 9, 10 and 11 study the fundamentals of complex sequential circuits with registers, counters and state machines using basic I/O components like push–buttons, switches, LEDs and displays. The systems designed on the last two sessions are complex enough to divide them on datapath and control logic. This set of sessions focus on real applications of digital systems covering aspects like clock divider, edge detection, register loading, etc.

IV. STUDENT RESULTS

The course has been taught since 2011. The results obtained by our students are summarized in table II. As can be seen, in spite of using a complex language as VHDL, which requires a great grade of abstraction, the students get good results on the course. With some oscillations, a 66 % of students pass each year, which is a good figure giving the fact that they are first grade students.

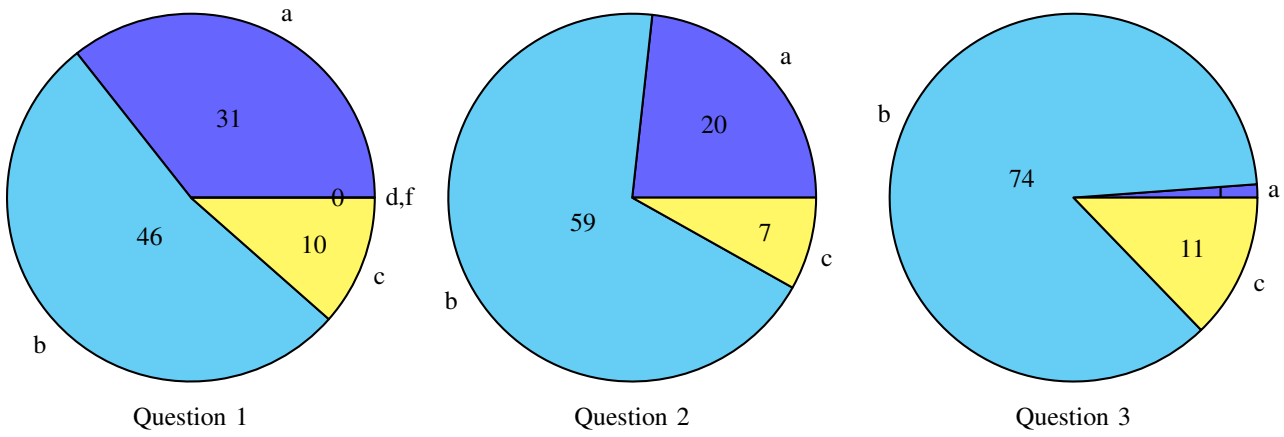

Fig. 3. Results from the survey.

.

TABLE II. STUDENT RESULTS

Year	N° Students	N° pass	Mean grade
2011	34	20	5.17
2012	36	29	5.68
2013	36	20	5.29
2014	33	23	6.9

V. COURSE ASSESSMENT

At the end of the course, a survey is given to all the students of the class to get their opinion about the course. Some questions of the survey and the possible responses are the following:

1) The subject of the course has been:
   a) Very Interesting.
   b) Interesting.
   c) O. K.
   d) Boring.
   e) Very Boring.
2) The VHDL language is:
   a) Easy.
   b) Some times it is complicated.
   c) For me is the most difficult thing of the subject.
3) The laboratory practice has been:
   a) Easy.
   b) O. K.
   c) Very difficult.
4) The course material (book and web page) is (5: very good, 0: very bad).
5) Your global opinion about the subject is (5: very good, 0: very bad).

The aggregated responses to the first tree questions from all the years that we have been teaching this subject are shown in the figure 3. The average responses to the four and five questions had been 4.1 and 3.9 respectively.

In consequence, our students are very satisfied with the course. However, as the results of question 2 shows, the VHDL language is a little bit complicated for them. This result makes we think that maybe we can switch to Verilog to see if with this language the students have less difficulties. Given that Verilog is closer to the C language, which has been studied in the first semester, we hope that it is possible some improvement in this aspect.

VI. ON THE ADAPTABILITY OF THE COURSE

Given the results obtained in this course, the authors can only encourage the adoption of the ideas shown in this paper to others institutions. As it has been said before, the course book is covered by a creative commons license (by–nc–sa) and available on–line on http://www.dea.icai.upcomillas.es/daniel/ApuntesED.pdf. The book covers all the content of the course discussed in this paper. Although in our laboratory we use FPGA boards from Altera, the book is "vendor agnostic", and can be used with other laboratory environments.

The laboratory assignments are also on–line at http://www.dea.icai.upcomillas.es/daniel/asignaturas/SistDig1_1_ITL/index.html. However, these assignments are centered around the DE1 board from Terasic, which is based on a Cyclone II FPGA from Altera. This means that if this assignments must be used in another environment, the low level details needs to be changed.

VII. CONCLUSIONS AND FURTHER WORK

When we first taught this course, we had some doubts about the learning ability of first year students to deal with the complexity of hardware specification using HDL. However, we have discovered that these students have less difficulties to learn the language that when they study in a first course digital logic and then, in a next course, they study advanced logic design and hardware specification languages.

At the end of the course, the students can design complex systems like the ones implemented on the last lab sessions or the problems solved in the final exam. In addition they are satisfied with the course and do not consider the HDL learning experience as too hard.

Finally, as a further work, given that Verilog is considered easier to learn than VHDL, it will be interesting to develop a course using Verilog as HDL language instead of VHDL and then compare the learning results of our students.

REFERENCES

[1] H. Abidin, M. Kassim, K. Othman, and M. Samad, "Incorporating VHDL in teaching combinational logic circuit," in *2010 2nd International Congress on Engineering Education (ICEED)*, Dec. 2010, pp. 225–228.

[2] J. Juan, E. Ostua, and D. Guerrero, "Methodology updating experience in basic digital electronics teaching," in *Technologies Applied to Electronics Teaching (TAEE), 2012*, Jun. 2012, pp. 58–62.

[3] J. C. H. Roth and L. L. Kinney, *Fundamentals of Logic Design*, 7th ed. Mason, OH: Cengage Learning, Feb. 2013.

[4] T. L. Floyd, *Digital Fundamentals*, 10th ed. Upper Saddle River, N.J: Prentice Hall, Mar. 2008.

[5] J. F. Wakerly, *Digital Design: Principles and Practices Package*, 4th ed. Upper Saddle River, New Jersey: Prentice Hall, Aug. 2005.

[6] J. C. H. Roth and L. K. John, *Digital Systems Design Using VHDL*, 2nd ed. Mason, OH: Cengage Learning, Mar. 2007.

[7] M. M. Mano and M. D. Ciletti, *Digital Design: With an Introduction to the Verilog HDL*, 5th ed. Upper Saddle River, NJ: Prentice Hall, Jan. 2012.