DIGITAL VLSI SYSTEMS DESIGN

Digital VLSI Systems Design

A Design Manual for Implementation of Projects on FPGAs and ASICs Using Verilog

By

Dr. S. Ramachandran

Indian Institute of Technology Madras, India

Springer

A C.I.P. Catalogue record for this book is available from the Library of Congress.

ISBN 978-1-4020-5828-8 (HB)
ISBN 978-1-4020-5829-5 (e-book)

Published by Springer,
P.O. Box 17, 3300 AA Dordrecht, The Netherlands.

www.springer.com

Printed on acid-free paper

Contents

Preface

This book deals with actual design applications rather than the technology of VLSI Systems. This book is written basically for an advanced level course in Digital VLSI Systems Design using a Hardware Design Language (HDL), Verilog. This book may be used for teaching undergraduates, graduates, and research scholars of Electrical, Electronics, Computer Science and Engineering, Embedded Systems, Measurements and Instrumentation, Applied Electronics, and interdisciplinary departments such as Biomedical, Mechanical Engineering, Information Technology, Physics, etc. This book also serves as a reference design manual for practicing engineers and researchers. Although this book is written for an advanced level course, diligent freelance readers, and consultants, especially, those who do not have a first level exposure of digital logic design, may also start using this book after a short term course or self-study on digital logic design. In order to help these readers as well as regular students, the book starts with a good review of digital systems design, which lays a solid foundation to understand the rest of this book right up to involved Project Designs unfolded gradually.

Contents of the Book

The book presents new source material and theory as well as synthesis of recent work with complete Project Designs using industry standard CAD tools and FPGA boards, enabling the serious readers to design VLSI Systems on their own. The reader is guided into systematic design step by step starting from a buffer to full-fledged designs right up to 120,000 gates. At every stage, the reader's grasp of the developing subject is challenged so that he or she may stand on his or her own feet after completing the course. Easy to learn Verilog HDL is made use of to realize the designs. A bird's eye view on what the reader is going to learn shortly is shown as follows:

- Introduction to VLSI Systems Design
- Features and architectures of latest FPGAs of leading vendors
- Detailed review of Digital Systems Design
- Introduction to Verilog Design
- Verilog coding of Combinational and Sequential Circuits
- Writing a Test Bench
- RTL Coding Guidelines
- Design Flow for VLSI Systems and Design Methodology
- Simulation using industry standard Modelsim Tool

- Synthesis using industry standard Synplify Tool
- Place and Route and Back Annotation using industry standard Xilinx Tool
- Verilog Coding of Memories and Arithmetic Circuits
- Development of Algorithms and Verification using High Level Languages such as Matlab
- Architectural Design
- VLSI Systems Design for a couple of projects as examples: PCI Arbiter and the Discrete Cosine Transform and Quantization Processor for Video com pression applications
- Complete Hardware Implementations using FPGA Board: A Traffic Light Controller and a Real Time Clock as examples
- Suggestion of Projects for Implementation on FPGAs/ASICs

Approach

The reader is taken step by step into designing of VLSI Systems using Verilog. To start with, an overview of VLSI Systems is presented. Features and architectures of the latest FPGAs of leading vendors are also presented. The design starts right from implementing a single digital gate to a massive design consuming well over 100,000 gates. Following a review of basic concepts of digital systems design, a number of design examples are illustrated using conventional digital components such as Flip-flops, Multiplexers, De-multiplexers, Decoders, ROM, Programable Array Logic, etc. With these foundations, the reader is introduced to Verilog coding of the components mentioned earlier as well as designing systems for small end applications. These designs are tested using Test Benches, also written in Verilog. All HDL codes the reader wishes to develop for various applications must conform to Register Transfer Level (RTL) Coding Guidelines, without which no chip can work satisfactorily. These guidelines are presented at length.

The sequel to the design is to simulate, synthesize, and place and route. Since our sincere interest lies in making the reader a full-fledged engineer, we use the same popular development tools used in the Industries and Research Laboratories such as Modelsim (a simulation tool of Mentor Graphics), Synplify (a synthesis tool of Synplicity Inc.), and Place and Route and Back Annotation of Xilinx Inc. Equipped with these powerful tools, the elucidation of design progresses into more and more complex designs such as Memories and Arithmetic Circuits extending into VLSI realms.

Complex Project Designs usually need the development of new Algorithms and Architectures for Optimum realization. These issues, which stimulate creative thinking and indispensable for researchers, are thoroughly analyzed and solved in this book. Armed with all the features mentioned earlier, complete Project Designs are presented. This is followed by hands-on experience in designing systems using FPGA and Input/Output Boards. Once the path is shown for designing VLSI Systems systematically, numerous Project Designs are suggested for FPGA/ASIC Implementations.

The book gives complete RTL Compliant Verilog codes for several projects, which are synthesizable and works on the hardware based on FPGA or ASIC. Most of the Verilog codes developed in this book can be readily used in new projects the reader may undertake in his or her study or career. The advantages accruing out of this strategy are two fold: One, the students are trained to suit industries/R&D Laboratories and, therefore, industries and other employers need not spend time and money to train them when they are hired. The other advantage is to provide in-plant training in industries, etc. based on this book to retrain old (as well as new) personnel in the new technologies.

Brief Description of the Topics Covered

The following is a brief description of the topics covered in each chapter of this book:

Chapter 1 presents an introduction to Digital VLSI Systems Design. The evolution of VLSI Systems over the years has been described. This chapter also outlines a number of applications for the VLSI Systems, thus motivating the reader to undertake a serious study of the subject and in turn be a contributor. This chapter shows how FPGA based system designs score over processor based systems. The features and architectures of latest FPGAs available in the market are presented.

Chapter 2 provides a detailed review of digital systems design so that the reader may understand the rest of the book without any difficulty or needing to refer to logic design fundamentals from another book. This chapter starts with the number systems, design of combinational circuits followed by designs using Programmable Logic Devices such as ROM and Programmable Array Logic. Thereafter, it deals with the design of sequential circuits. It shows how to design systems using the conventional state graphs and ASM Charts. Digital system designs are illustrated with a number of examples.

Chapter 3 presents a brief introduction of the evolution of Hardware Design Language. Verilog is introduced as a tool for realizing digital systems design. The advantages of Verilog coding over the traditional schematic circuit diagram approach are established, especially when the design of VLSI circuits crossing over 50,000 transistors mark is encountered. A number of design examples using Verilog are illustrated for both combinational and sequential circuits. These examples cater to the frequently used digital circuits in a system design, especially in industries. Only the cores of the designs are initially presented to expedite the learning process. Later on, full-fledged codes are presented so that the reader may simulate, synthesize and place and route. Register Transfer Level (or Logic) coding, vital for designing chips that work successfully, is the main emphasis of this design book and is discussed in a later chapter.

Chapter 4 shows how to write an effective test bench. The basic concept of a test bench is shown by presenting a simple design and a model test bench for testing the design exhaustively. For bigger designs, an elaborate test may prove to be difficult. In such cases, the test may be carried out for a range of inputs covering minimum, maximum, center, and few other input values applied judiciously.

In the previous chapter, designs for combinational and sequential circuits were dealt. Test benches for the same are presented in this chapter. Usually, the test bench size will be smaller than that of the design. This chapter is especially useful for verification engineers.

Chapter 5 deals with the Register Transfer Level Coding Guidelines. Every designer is vitally interested in making his or her design work when implemented as a hardware, which uses FPGA or ASIC. For successful working of a system, RTL coding techniques are inevitable. This requires a high degree of discipline or care while designing such systems. This chapter discusses RTL coding techniques in depth, which is basically adhering to synchronous design practices. RTL approach deals with the regulation of data flow, and how the data is processed using register transfer level as the primary means. Since we deal with a synchronous design, it should naturally run smoothly through various tools such as simulation, synthesis, and place and route, which tools are described at length in succeeding chapters.

Chapter 6 presents the VLSI Design flow along with the design methodologies that may be gainfully employed so that one may become conversant with various steps involved in designing a product. Several designs were considered in Chapter 3 and their test benches in Chapter 4. This was followed by RTL coding guidelines in Chapter 5. The next logical step in the design flow is the simulation, vital for testing one's design. All the Verilog designs presented in the third chapter are analyzed using waveforms in the present chapter. Industry standard Modelsim tool of Mentor Graphics is employed for the simulation. A command summary of the Modelsim tool, which serves as a quick reference while using the tool, is also furnished. Even though the functionality of a design is checked using simulation, it does not test time critical paths or furnish insights into gate delays, unless back annotated, since simulation does not map a target chip such as an FPGA, where the design will have to reside ultimately. These features are possible with synthesis tool, which is presented in the next chapter.

Chapter 7 covers the synthesis of designs using Synplify tool, widely used in industries. The salient features of synthesis are mapping of an FPGA device, logic optimization, and viewing schematic circuit diagrams of the Verilog code. The tool creates optimized Verilog file and Electronic Data Information Format (EDIF) file, which may be used for simulation and vendor specific place and route respectively. EDIF file is exported to the next tool, the Place and Route tool, for creating a bit stream of the design. Synplify tool supports all types of FPGAs. In order to learn and fix compilation errors and simulation errors in Modelsim and Synplify tools, errors are created deliberately and known correction applied. A command summary of the Synplify tool is furnished for quick reference while using the tool.

Chapter 8 covers the Place and Route (P&R) tool of Xilinx, also widely used in industries. The salient features of Xilinx P&R are the creation of a 'bit' file from EDIF file created by the Synplify tool or from source file directly, specification of user constraints such as clock speed and FPGA pins, remapping of the target FPGA device if desired, back annotation and floor planning. The back annotated file, which reflects the actual gate delays, is simulated again in Modelsim to ensure that the design is working correctly at the maximum frequency reported by

the Place and Route tool. Report file generated by the P&R tool gives the maximum frequency of operation possible as well as the gate count (chip complexity) for the design. A command summary of the Xilinx P&R tool is furnished for ready reference.

Chapter 9 presents the memory design, which is one of the most important aspects of a VLSI System Design. This chapter shows the way to design various types of on-chip ROMs and RAMs, some of them unconventional, in order to meet the special requirements of a particular application. The size of memory that could be incorporated on-chip is usually limited by the order of a few tens of Kilo Bytes with the currently available FPGAs. This limitation is fast changing with the advances in the technology. In applications, where large memories are called for, external memories such as the commercially available static RAMs, ROMs, Flash RAMs, dynamic RAMs, etc. may be used. Towards this end, a controller design that interfaces with commercially available external memories is presented in this chapter. However, the access speed of external memory falls by a factor of two when compared to on-chip memory. On the other hand, on-chip memory increases the chip area consumed. Therefore, the designer must consider carefully the pros and cons before making the choice for on-chip memory or external memory in a system design.

Chapter 10 presents arithmetic circuits such as add/subtract, multiply, etc. These circuits are computationally intensive and, therefore, conventional methods are not sufficient for real time implementations on FPGA or ASIC. In order to speed up the processing considerably, we will have to base our designs on massively parallel circuits and heavy pipelining. This chapter presents arithmetic circuit designs such as signed adders and multiplier with a high degree of parallelism and pipelining for computationally intensive applications such as video compression.

Chapter 11 deals with the development of algorithms for various projects so that they are suitable for implementation on FPGAs/ASICs. Complex applications such as video codecs have involved algorithms at their core, which need to be adapted or developed depending upon how we wish to implement the system. The design methodology or strategy would depend upon whether we need to implement the system using software such as C or by a HDL such as Verilog. The viability of the project is decided by the successful working of the algorithms or the design methodology using Matlab or C, which is also discussed at length. While developing algorithms for hardware implementation, we need to keep the actual hardware in mind and subsequently, design the architecture. Only then, the algorithm can be converted into an actual working product. This chapter also covers the verification of concepts and algorithms developed earlier using a high level language such as Matlab. This must be carried out before taking up the architecture and Verilog design. Designers, in general, have a tendency to bypass this vital step and get into trouble later on, after completing Verilog codes, at the time of debugging. This is especially true in large designs, and is worthwhile to take little more trouble of coding in Matlab or C, preferably in Matlab, to save lots of trouble and time.

Chapter 12 presents the development of architectural designs. In the last chapter, development of algorithms were presented and verified for a number of applications in the field of video processing as examples. These algorithms were developed in such a way that the applications may be mapped onto an FPGA or an ASIC. The next logical step is to work out a detailed architecture keeping the actual hardware such as registers, counters, combination circuits, etc. in mind. In this chapter, the architectural designs are developed for the same applications that we had undertaken in the previous chapter.

Chapter 13 presents VLSI System Design examples for two projects, namely, PCI Arbiter and the Discrete Cosine Transform and Quantization Processor for Video compression applications. While presenting these designs, emphasis is laid on systematic design. This comprises identification of a project based on need, formulating detailed specifications, development of algorithm and verification, or proving an algorithm or concept using a high level language such as Matlab or C to establish its feasibility, development of detailed architecture based on actual hardware components, Verilog RTL coding, simulation, synthesis, place and route, and back annotation. The methodologies adopted in these designs are to use highly parallel and heavily pipelined circuits in order to increase the throughput and to be platform independent, whether an implementation uses an FPGA or an ASIC. No vendor specific modules are used and, hence, these designs are universal and can work on any FPGA or ASIC. The design methodologies presented in this book are equally applicable to other HDLs such as VHDL.

Chapter 14 presents a couple of complete hardware implementations, namely, a traffic light controller and a real time clock, as examples. These applications are based on ready made boards available such as an FPGA board and a digital input/output board. The design methodologies adopted in these designs may be extended to any other Project Design or any other FPGA or ASIC and I/O boards. These system designs are presented in a systematic manner, starting from detailed specification. The need for formulating the right type of architecture is emphasized and designed with actual hardware components in mind. The signal nomenclatures adopted in the architectures are actually used as it is in realizing their designs in Verilog conforming to the RTL coding guidelines. Simple test benches are developed and simulated using Modelsim tool to ensure the correct functioning of the designs. These are followed by running the synthesis tool and the place and route tool to get the bit stream files. The hardware for each of the designs is subsequently setup and the corresponding bit streams downloaded into the FPGA. Elaborate testing is thereafter conducted on the actual hardware to ensure the correct working of the systems designed.

Chapter 15 suggests a number of projects for the reader to design and implement on FPGAs/ASICs, category-wise. Issues involved in Digital VLSI Systems Design are discussed at length in order to aid the reader to quickly develop products. A brief introduction of embedded systems design is presented. Detailed specifications and basic architectures for a couple of applications for FPGA/ASIC implementations are furnished for the reader to make a start and gain hands-on experience in Digital VLSI Systems Design.

How to Use This Book?

The material in this book has been developed over several years as a result of teaching students of various disciplines and guiding practicing engineers and students in their projects/thesis work. Teaching/training styles vary among countries, universities, colleges, and industries and may, therefore, be suitably organized. In the light of imparting VLSI Systems Design to students and engineers of various disciplines and, research over the years, the following pattern of education is recommended:

Second Year Under Graduates of Electrical/Electronics/Computer Engineering

Specializations such as Power systems, Electronics, Communication, Power electronics, Embedded Systems, Controls, Measurements and Instrumentation, Applied Electronics, etc. are included in Electrical/Electronic Engineering. Chapter 1 presents a general view of what VLSI Systems are and Chapter 2 is a review material, which the students may study independently if they have just completed a first level digital design course. Optionally, they may be taught Chapter 2, especially if there is a semester gap after they have completed their first level digital design course. Thereafter, they may be taught Chapters 3, 4, and 6. Chapter 5, which presents RTL coding guidelines, may be deferred to the third year.

Third Year Under Graduates of Electrical/Electronics/Computer Engineering

Chapter 5, and review of Chapter 6 if there is a semester gap after they have completed their second year course mentioned earlier, may be taught to start with. Thereafter, they may be taught other tools such as synthesis and place and route from Chapters 7 and 8 respectively. Finally, Memory designs presented in Chapter 9 may be taught.

Fourth Year Under Graduates of Electrical/Electronics/Computer Engineering

After a brief review of simulation, synthesis, and place and route tools, Arithmetic circuits design (Chapter 10) may be taught. A simple Project Design, PCI arbiter, may be taught from Chapter 13. More involved design such as the DCTQ processor may be by-passed. Thereafter, hardware designs based on FPGA boards may be taught from Chapter 14. This may be followed by Chapter 15. Finally, mini projects may be assigned to students in small groups encouraging team work, yet clearly defining individual student goals. The second half of a semester may be used for the mini projects. These projects may be based on FPGA boards, if feasible. Students may be assessed by asking them to present their progress from time to time. These presentations may be strategically scheduled as follows:

First week: Detailed specification of the project.
Second week: Algorithmic development, if any, and Hardware Architecture.
Fifth week : Verilog coding of the design and test bench.
Sixth week : Simulation/synthesis/place and route results and
Eighth week : Demonstration, documentation, and final presentation.

Many projects are suggested throughout the book, and many more may be created by the instructor and the students.

First Year Post-Graduate Students and Ph.D. Scholars of Electrical/Electronics/Computer Engineering

Chapters 1 and 2 are review materials, which the students may study independently. Chapters 3 to 5 may be taught in detail, if the students have not covered these topics in their earlier studies. A quick exposure to three industry standard tools in Chapters 6 to 8 may be made using command summary of these tools, leaving the students to gain proficiency with the tools in the laboratory. This may be followed by Chapters 9 and 10. Chapters 11 and 12 are important for developing new algorithms and their verification and the design of architecture, especially for those doing research (MS or Ph.D.). If hard pressed for time, one of the three applications in these chapters, say, the DCTQ alone need be taught. Thereafter, Chapters 13 to 15 may be taught. Individual mini projects may be assigned to students in the second half of a semester as detailed for the fourth year students earlier. The above recommendation, although appear to be too crammed for a semester study, has been actually tested for post-graduates and Ph.D. scholars of Electrical/Electronics engineering. If there is room, the entire book may be taught spread over two semesters: Chapters 3 to 10 in the first semester and the rest in the second semester.

Third Year under Graduates of Information Technology/Computer Science and Interdisciplinary Departments such as Bio-medical, Mechanical Engineering, and Post-graduate Students of Physics

Chapters 1 to 4 and Chapter 6 may be taught spread over a semester. The Chapter 5 on RTL coding guidelines may be deferred to the fourth year.

Fourth Year Under Graduates of Information Technology/Computer Science and Interdisciplinary Departments such as Biomedical, Mechanical Engineering and Post-graduate Students of Physics

Chapters 5, 9, 10, and all topics of Chapter 11 for developing new algorithms and their verification may be taught. PCI arbiter design in Chapter 13 and Traffic light controller design in Chapter 14 may also be taught. Applications from Chapter 15 can be taught finally. Individual mini projects may be assigned to students in the second half of a semester as detailed for the fourth year electrical engineering students earlier.

Each of the above recommendations is to be taught over a semester.

In-plant Industrial Training

Experience shows that in many industries, R&D laboratories, etc., there are no periodic and systematic orientation programs or training, either for new recruits or for existing personnel. As a result, with rapid technological strides, employees do not come up to the expected level of the employer. The lack of proper and regular training has also rendered the 'attrition management' increasingly difficult. These problems can be alleviated to a great extent by conducting regular orientation programs for the new recruits even if they come with experience, and periodical hands-on training based on this book to retrain old (as well as new) personnel in the new technologies. Depending upon the level of personnel, the management can form their own curriculum for orientation programs and training as per the recommendations made earlier for various categories of students using this book. Designers may be encouraged to create a library of developed codes in Verilog/VHDL/Matlab/C with proper documentation on-line so that the on-going product designs are completed quickly without reinventing the wheel, thus improving the productivity and hence profitability dramatically.

Assignments and Laboratory Work

Merely studying the text would not make the reader an adept in designing VLSI Systems. On the other hand, one is sure to become proficient if every assignment given towards the end of every chapter is sincerely solved. These must be supplemented by inventing more number of assignments and solving them. Most of the assignments, especially those presented in the second chapter, are based on industry related problems and placement questions faced by a large number of students globally. Chapters 3 to 10, 13, and 14 contain a large numbers of illustrated examples including full-fledged projects, all of which can be used as source materials for laboratory work. Chapters 6, 7, and 8 respectively present the simulator (Modelsim), synthesis (Synplify), and place and route and back annotation (Xilinx) tools and may be thoroughly studied before starting their respective laboratory work. As laboratory assignments, the problems/assignments presented in each of the Chapters 3 to 15 may be selectively allocated, depending upon the categories of students as presented earlier.

Verilog and Matlab Source Codes Supplied in CD

All Verilog codes, be they designs or test benches, illustrated in Chapters 3 to 5, 7, 9, 10, 13, and 14 are in their respective folders in the CD provided in the book. Similarly, the source codes of Matlab illustrated in Chapters 11 and 13 are in different folders. These may be copied in hard disks and tested using the tools. 'Readme' files explain the usage of various files. Command summaries for Modelsim, Synplify and Xilinx tools are also included. Also, a summary of the usage of RTL

Verilog codes is provided in separate files. Reader's technical skills may be enhanced by going through the PPT: How to make oral/written presentations?

Mini Project and Project Work

Some assignments in Chapters 7, 8, and 11 to 15 are suitable for mini projects for students of fourth year undergraduate and post-graduate students as recommended earlier. Over 100 projects are listed in Chapter 15, which may be selected for FPGA/ASIC implementation by both undergraduate and post-graduate/research students. Many mini projects may also be carved out of these projects. Many more mini projects, full-fledged projects and research projects, may be created on similar lines as illustrated in above mentioned chapters.

Solution Manual

Solution Manual for the assignments presented towards the end of each chapter is available to teachers from the publishers on a CD or on their website. The solution manual contains all source codes and reports of solved assignments in the book and presentations for quick and easy dissemination of the subject.

Acknowledgment

The author is thankful to various industries he has been associated with over decades and Indian Institute of Technology, Madras for providing excellent resources and working environment. Heart-felt thanks are due to Prof. S. Srinivasan, IITM, who has been a great source of inspiration for writing this book. Thanks are also due to numerous co-designers in industries, undergraduate to research students for their lively discussions on projects and staff for their timely help in preparing the manuscript of this book. Special thanks are due to the publishers in bringing out this book quickly, yet maintaining high quality.

S. Ramachandran

Chapter 1

Introduction to Digital VLSI Systems Design

The electronics industry has achieved a phenomenal growth over the last few decades, mainly due to the rapid advances in large scale integration technologies and system design applications. With the advent of very large scale integration (VLSI) designs, the number of applications of integrated circuits (ICs) in high-performance computing, controls, telecommunications, image and video processing, and consumer electronics has been rising at a very fast pace. The current cutting-edge technologies such as high resolution and low bit-rate video and cellular communications provide the end-users a marvelous amount of applications, processing power and portability. This trend is expected to grow rapidly, with very important implications on VLSI design and systems design.

Information technology (IT) focuses on state of the art technologies pertaining to digital information and communication. The IT sector is the fastest growing industry in recent times. With the world growing smaller day by day and business going global, the need for better devices and means for communications becomes all the more important. One of the most important characteristics of IT is its increasing need for very high processing power and bandwidth in order to handle real-time applications: video, for example. This has led to the need for faster and increasingly more efficient products to enable better telecommunications. It is this ever-growing demand in the modern world that is making many countries invest heavily in VLSI systems design. Manufacturing VLSI systems on chips is an involved process and comprises a number of activities: VLSI systems design using electronic design automation (EDA) tools; computer aided design (CAD) in the manufacture of VLSI chips; foundry activity starting from base wafer to packaged and tested ICs; and design, development, and manufacture of capital equipments for producing VLSI chips. All these activities except VLSI systems design using EDA tools are capital intensive. The latter, however, is knowledge intensive. Since applications are numerous and growing rapidly, challenging as well as interesting, VLSI system application designers are in greater demand than professionals working on chip technology.

Ever-increasing global communications has opened up a brand new vista for people interested in a career in the information technology and VLSI design including embedded systems. The advent of advanced EDA tools gives one the freedom to innovate and experiment to develop a new product that could be the next breakthrough in all spheres of research and development. A career in these sectors will give the reader access to the technical resources to work with and design better world-class products. As a product developer, the reader will be providing solutions to international markets and, therefore, technical skills need to be

of the best quality. It requires continuous learning, systematic approach, and sustained efforts to realize one's dreams. It is both intellectually stimulating and exciting to be part of creating the technology of the future. This book is an earnest attempt to equip the reader completely for this challenging task and shape him or her as a highly skilled professional.

The rest of this chapter is organized as follows: In the next section, a brief introduction to the evolution of VLSI systems is presented. This is followed by presenting a short list of growing applications for VLSI systems in order to motivate the reader towards undertaking product designs seriously. In Sections 1.3 and 1.4, the advantages as well as limitations are discussed for processor based systems and embedded systems respectively. Section 1.5 presents field programmable gate arrays (FPGAs) based designs and their advantages over processor and embedded controller based designs. It discusses briefly the selection criteria of hardware. It also discusses in detail various issues involved in one of the latest applications, namely, video compression. Complete project design for video compression is presented in later chapters. An introduction to digital system design using FPGAs is presented in Section 1.6. Following this, detailed descriptions of features and architectures of various types of popular FPGAs manufactured by leading vendors are presented so that the readers may select the right type of FPGAs for their applications.

1.1 Evolution of VLSI Systems

With the advent of discrete semiconductor devices such as bipolar transistors, unijunction transistors, field effect transistors, etc., miniaturization started in full-swing, replacing bulky systems that used vacuum tubes. During 1950s computers that were made using vacuum tubes occupied an entire floor of a big building. Vacuum tubes are even now used in high power applications such as radio transmission and HAM radios. Gradually, attempts were made to integrate several circuits, be it analog or digital, in a single package. These attempts succeeded in producing both analog and digital ICs, as well as mixed signal ICs. Analog ICs offered operational amplifiers, multipliers, modulators/demodulators, etc., while digital ICs integrated AND, OR, XOR gates and so on.

Digital ICs are broadly classified according to their circuit complexity measured in terms of the number of logic gates or transistors in a single package. Chips falling under the category of small scale integration (SSI) contain up to 10 independent gates in a single package. The inputs and outputs of these gates are connected directly to the pins in the package with provision for connections to a power supply. With the advances in integration technology, more devices having a complexity of approximately 10 to 100 gates were packed in a single package. They were called medium scale integration (MSI) devices. Decoders, adders, multiplexers, de-multiplexers, encoders, comparators are examples of MSIs. Thereafter, large scale integration (LSI) devices emerged, which integrated between 100 and 1000 gates in a single package. Examples of this category include digital systems

such as processors, memory chips, and programmable logic devices. Finally in late 1970s, very large scale integration devices containing thousands of gates within a single package became a reality. Personal computer chips such as 80186, 80286 of Intel are examples of this category. Since then, integration has been growing by leaps and bounds crossing 10 million gates in a single package, going into realms of ultra large scale integration (ULSI), system level integration (SLI), and system-on-chip (SOC). FPGAs fall under all the above high-end categories starting from VLSI. The foregoing classifications are summarized in the following.

Category	**Date**	**Density (gates)**
Single transistor	1959	1 device
Logic gate	1960	1
Small scale integration (SSI)	1964	Up to 10
Medium scale integration (MSI)	1967	10 – 100
Large scale integration (LSI)	1972	100 – 1000
Very large scale integration (VLSI)	1978	1000 – 10000
Ultra large scale integration (ULSI)	1989	10000 and above
SLI/SOC	Late 1990s	> 10 million

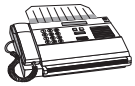
Systems were implemented in all the above categories and are still being implemented using discrete transistors for large power applications and SSIs to SOCs for progressively larger systems. In the next sub-section, we will see what applications these systems can be configured for.

1.2 Applications of VLSI Systems

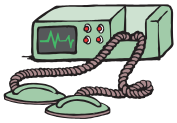
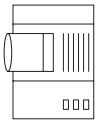
VLSI system applications have become all pervasive in various walks of life like communications including internet, image and video processing, digital signal processing, instrumentation, power, automation, automobiles, avionics, robotics, health and environment, agriculture, defense, games, etc. There is hardly anyone who does not know what a cell phone is. From MP3 players, camera cell phones and GSM to Bluetooth and Ipods, everyone wants all the features squeezed into a single device as small as possible. Some of the ever-growing applications are as follows:

- Digital cameras
- Digital camcorders
- Digital camera interface
- Digital cinema
- Digital display

- Digital TV and digital cable TV
- Digitizer for analog NTSC/PAL/SECAM cameras
- Display interface
- Mobile phone
- FAX machine
- PDA
- Scanner



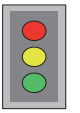
- Anti-lock brakes
- Automatic transmission
- Cruise control
- Global positioning system for automobiles
- Electro cardiograph
- Life-support systems
- MRI/CT scan
- LCD projector

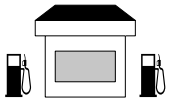

- Low-cost computer
- Mobile phone personal computers
- Scan pen and PC notes taker
- Automated baggage clearance system in airports
- Avionic systems
- Flight simulator
- Instrument landing system
- Ship controls
- Driverless shuttle
- Cruise controls
- Traffic controller
- Washing machine







- Petrol/diesel dispenser
- Demodulator for satellite communication
- Encryption/decryption
- Network card
- Network switches/routers
- Quadrature amplitude modulator (QAM) and demodulator
- Wireless LAN/WAN



More applications are presented in the final chapter, which may be taken up for implementation by the readers. They are classified into various categories, such as automotives, avionics, control system applications, medical applications, and video processing applications, to name a few. Brief descriptions are also presented. Curious readers may have a peep into those applications before going over to the next section.

VLSI systems can be designed using any of the following: 8/16/32/64 bit general-purpose processors, microcontrollers, DSPs, FPGAs, or ASICs depending upon the applications, throughput, market potential, etc. The advantages and limitations for each of these categories are discussed in the following sections.

1.3 Processor Based Systems

Designers have wide choice of selecting processors, which include general- purpose processors, microcontrollers, application specific instruction processors (ASIP), reduced instruction set computers (RISC), complex instruction set computers (CISC), digital signal processors (DSP), etc. ASIPs are optimized for specific class of applications such as telecommunications, digital signal processing, embedded controls, etc. Each of these processors has an instruction set with a specific class of applications in mind. Nevertheless, they are all processors executing instructions sequentially, differing only in performance, processing speed, effectiveness, power, cost, etc. Most of these processors fall under the category of VLSI. Over the years, a wide variety of systems have been designed with these processors. These cover an impressive spectrum: data processing systems, data acquisition systems, programmable logic controllers and numerous industrial control systems, measuring instruments, image processing systems, etc.

Selection of a processor for an application is not an easy task, especially when numerous processors are available. The designer is overwhelmed by numerous instruction sets. With a change of processor for a new project, the designer is forced to learn a new instruction set, which is as arduous as learning a new language, if the designer is not already familiar with the processor. Often, there is great confusion and struggle if instruction sets of processors have conflicting meaning such as

the position of source and destination in an instruction. For example, Intel microprocessor instructions start with the destination first, followed by the source, whereas Motorola microprocessor instructions start with the source first, followed by the destination. Even in the hardware realms, process timings of processors vary widely. The associated peripheral chips also differ, necessitating major redesign of software/hardware, if processors are changed. All these introduce considerable delays in the project. In order to avert disaster, designers try to bend the new project towards their favorite processor(s). For small performance requirements, this tactic may serve the intended purpose. However, in time critical applications, the favorite processor may be a misfit.

Earlier, we discussed about the difficulties a designer undergoes while designing systems using processors, especially when the need arises to unlearn instruction set of the familiar processor and, instead, learn a new processor assembly language instructions. This difficulty may be eliminated if the designer has knowledge of a high level language such as C or C++. Of course, the designer has to learn C if he or she has no knowledge of the same. It is a well known fact to designers that C codes generate longer codes than assembly language instructions do. This, in turn, would lower the processing speed considerably and may prove to be a bottleneck in real-time applications.

1.4 Embedded Systems

General-purpose processors as well as microcontrollers are popular in embedded systems due to several good features such as low cost, good performance, etc. If hardware is already available, the designer needs to concentrate only on software development and integration of the system. Therefore, for small quantity of end products, it will be cost-effective as well as reduce the development cycle time dramatically if the embedded systems are built using bought out populated electronic cards such as STD/VME bus cards and the like. The designer may write some part of the programs in C and other parts in assembly language and link them together. Many tools are available to aid designers in product development – hardware: troubleshooters, emulators, logic analyzers, and programming units and software: assemblers, compilers, linkers, and C. These tools are a must if the embedded system designer is to bring out a working product into the market fast. Designers must try to achieve the goal of designing complete systems by treating hardware and software in a unified way. Hardware/software co-design emphasizes this unified view that enables the co-development of systems using both hardware and software, especially during synthesis [1, 2].

Processor based embedded systems are quite effective for small and medium-end applications. For medium to high-end embedded systems design, field programmable gate arrays (FPGAs) and application specific integrated circuits (ASICs) are the right choice. In the near future, FPGAs may be expected to be cost-effective even for small-end applications. The development tools that will convert ideas into reality of a working system for this category are Verilog/VHDL

(hardware design language) compilers, simulation, synthesis and place and route tools and programmers. In the next section, we will discuss how FPGAs offer much higher performance than processors including DSPs and those used in embedded systems.

1.5 FPGA Based Systems

A number of software and hardware implementations have been reported for various real-time applications such as video codecs. Although software implementations are easy to realize on general-purpose microprocessors, multiprocessors, microcontrollers, or digital signal processors, their instruction sets are not well suited for fast processing of computationally intensive real time processing applications such as high resolution compression/video scaling of motion pictures, satellite communication modulator/demodulator, etc. In addition, the instructions are executed sequentially, thus slowing down the processing further. For example, one of the promising digital signal processors, which was used to implement MPEG based video codec could only process still monochrome images of resolution 512 × 512 pixels at one frame per second instead of the required frame rate of 30. In contrast to this, the hardware implementations based on FPGAs and ASICs can exploit pipelining and massively parallel processing resulting in faster and cost-effective designs.

ASIC designs are suitable if high-volume production is envisaged. However, in the research and development phase and for rapid prototyping of a new design, FPGA is the right choice. Further, FPGA implementation is cost-effective for low volume applications. In a later chapter on project designs, we will show that high resolution motion pictures of sizes 1600 × 1200 pixels can be processed (actually compressed) at 30 frames per second using FPGAs. The Verilog code developed for this application can also be implemented without any modification of the codes on an ASIC for still higher resolution pictures by over three times in the present day technology. The following sub-sections discuss briefly video compression application using FPGAs as an example.

1.5.1 FPGA Based Design: Video Compression as an Example

FPGAs offer high performance in terms of processing speed and high chip density, thus suiting every conceivable application, whether small or high end, yet remaining cost-effective. An entire VLSI system can be housed in a single FPGA device. Although many applications are possible, we will discuss the basics of video compression implementation as an example. Video compression is an application that demands high performance and high density. For example, a color motion picture of high resolution, 1600 × 1200 pixels can be compressed and transmitted or received over a serial channel at a real time processing speed of 30

frames/second using an FPGA. Complete project design for this application, in addition to many other applications, is presented in later chapters. The design methodology presented for this application is equally applicable for any other application. In the following sub-sections, we will briefly discuss the need for video compression, what standards govern the implementations, various issues involved in the design, and a review of the evolution of video compression implementations.

Need for Video Compression

Image processing applications such as high definition television, video conferencing, computer communication, etc. require large storage and high speed channels for handling huge volumes of data. For instance, one hour of color motion picture of size 1024 × 768 pixels at 30 frames per second in the raw format will need about 255 GB of memory and 566 Mbps channel speed for effective communication. In order to reduce the storage and communication channel bandwidth requirements to manageable levels, data compression techniques are imperative. Data compression in the order of 20 to 40 is normally feasible depending upon the actual picture content and techniques adopted for bringing about the compression.

It is of paramount importance that systems designed for applications communicate with one another effectively and also offer connectivity and compatibility among different services. These requirements are met if these systems are designed to conform to international standards such as JPEG, H.261, HDTV, and MPEG [3]. The development of standards by the ISO, ITU, etc. for audio, image and video, for both transmission and storage, has led to worldwide activity in developing hardware and software systems for a number of diverse applications. Although the standards implicitly address the basic encoding operations, there is enough freedom and flexibility in choosing the algorithms, the actual implementation, and devices. As such, the standards do not stifle the research and development activity, the main objective being maintaining compatibility and interoperability among the systems. The next sub-section describes briefly various standards available currently for compression of still image and motion pictures.

Video Compression Standards

JPEG has been recognized as the most popular and efficient coding scheme for continuous-tone still images. The JPEG compatible fast implementations find applications in color facsimile, high-quality newspaper wire photos, desktop publishing, graphic arts, medical imaging, digital still cameras, imaging scanners, etc. Examples of video sequence (motion picture) standards are H.261 for video telephony and video conferencing, MPEG 1 for digital storage media and MPEG 2 for generic coding of moving video and extending to television broadcasting and communication.

In recent years, additional standards such as JPEG 2000, MPEG 4, and MPEG 7 have also been introduced. The JPEG 2000 standard is intended to complement and not to replace the JPEG standard. Lossless and lossy coding, progressive by resolution and quality, high compression efficiency, error resilience, and lossless