M.Tech.
Programme Curriculum

(Effective from academic year 2018-19)

International Institute of Information Technology
Bangalore – 560100
July 2018

Table of Contents

1 Overall M.Tech. Programme Structure

The following Table provides a summary of the credit distribution in the M.Tech. programme.

Table 1: Overview of the curriculum

Term	Duration	Credits	Courses
Preparatory Term	2 weeks	Not applicable	Orientation sessions on various topics
Semester 1	16 weeks	16 credits	Foundation Courses
Semester 2	16 weeks	16 credits	Electives
		0 credits	Technical Communication for those found deficient in a test conducted in Semester 1 (Pass /Fail)
Semester 3	16 weeks	16 credits	Electives
Semester 4	26 weeks	16 credits	Masters Project / Thesis
Total			**64 credits**

2 Areas of specialization

The M.Tech. (CSE) curriculum has four areas of specialization:

- Theory and Systems(TS)
- Data Science(DS)
- Networking & Communication(NC)
- Signal Processing and Pattern Recognition(SP)

The M.Tech. (ECE) curriculum has three areas of specialization:

- VLSI Systems (VL)
- Networking & Communication(NC)
- Signal Processing and Pattern Recognition(SP)

See section 5 for details on criteria for award of specializations.

3 Preparatory courses

Students entering the M.Tech. programme are expected to come with some prior knowledge of programming. While we do not wish to conduct full-fledged programming courses at the Masters level, we will provide an opportunity for the students to hone up their programming skills in a structured way as part of the preparatory term. In the preparatory term, M. Tech (CSE) students will cover topics in programming (Python, C and Java) and some basic Data Structures.

In the preparatory term, M. Tech (ECE) students will cover topics in programming in C and some basic Electronics. The Programming courses will be taught with emphasis on hands-on programming exercises and projects.

4 Foundation Courses

M.Tech (CSE) students in the first semester will chooses courses worth at least 16 credits, from the following course. Each course here is of 4 credits.

1. Algorithms
2. Networking and Communication
3. Machine Learning
4. Mathematics for Machine Learning
5. Software Systems
6. Discrete Mathematics and Computability

M.Tech(ECE) students in the first semester will choose courses worth of least 16 credits, from the following list of courses.

1. Digital CMOS VLSI Design (4 Credits)
2. Analog CMOS VLSI Design (4 Credits)
3. Machine Learning (4 Credits)
4. Mathematics for Machine Learning (4 Credits)
5. Networking and Communication (4 Credits)
6. System Software (2 Credits)
7. System design with FPGA (2 Credits)
8. Principles of Embedded Systems (2 Credits)

Details regarding objectives, syllabus and lecture hours for each course are provided in the Appendix-A.

5 Electives

The number of electives to be completed by each student is **eight**. Thus the total number of credits that can be accumulated through electives is now 32 credits. Each elective will be associated with one or more areas of specialization.

If at least five of the eight electives (in the second and third semester) is from a particular specialization, say Data Science, then M.Tech (CSE) with specialization in DS is awarded. The name of the specialization (if any) is noted in the transcript. If these five electives are drawn from various specializations, then MTech is awarded without any mention of specialization.

Design of Elective course will be addressed in detail by the faculty concerned. This design of the course will be presented to faculty-meeting/Senate before being offered to students. Some of the Elective Courses may require a specific course(s) as a pre requisite.

6 Project Electives / Reading Elective

1. There are two forms of special electives called: Project Elective (PE) and Reading Elective (RE). These electives are intended for experiential and guided learning.

2. Every PE course at least have the following characteristics:
   - Overall Plan
   - Visible Output
   - Direct Supervision

3. PE and RE follow the usual letter grading pattern available to other courses.
4. MTech students may opt for at most one PE and at most one RE course, per semester. The total number of PE/RE courses a student can enroll for shall **not exceed three** in the entire programme duration.
5. Involvement of external institutional entities if any, as part of a PE course, should be expedited within the framework of the existing collaboration and IP policies of the Institute.

7 Thesis/Masters Project

Thesis/Masters Project shall be of 24 weeks duration and a student can accumulate 16 credits on successful completion of thesis or Masters Project.

<u>For the students pursuing Masters Project:</u>

- Masters Project to be considered as six months (not less than five months) of supervised project work carried out at industry or academic institutions.
- The Masters Project committee will ensure that a mid-term feedback is collected (for every student pursuing internship) to ensure smooth progress towards completion.
- At the time of Masters Project completion the Masters Project committee will also collect the certificate (satisfactory/unsatisfactory) from concerned person of the organization. If the certificate is unsatisfactory then the institute internship committee will review the matter and if they agree with the certificate given, and then the student has to carry on the internship again at same or different place. If the certificate is satisfactory then the student full fills the requirement of internship.

<u>For students pursuing thesis</u>, the following guidelines hold:

- There is an M.Tech. thesis committee comprising of the supervisor and at least two more faculty members. Members of this thesis committee will serve thesis and oral examiners for each student pursuing thesis.
- The thesis style rules should be available in LMS for all thesis students to use. Additionally we should make available both Word and LaTeX style files, which comply by these rules. If a student chooses to use a word processor, other than the ones above, (s)he is welcome to do so as long as the rules are met.
- A soft copy of the thesis in pdf format should be sent to IIITB librarian, a week before the final submission of thesis date according to the institute's calendar (which will be after the thesis's oral exam). The soft copy of thesis format must be officially approved by the librarian before the thesis goes in print and for binding.
- The M.Tech. academic calendar will have dates fixed for the following tasks specific to thesis evaluation: constitution of thesis committee, submission of draft to the committee(s) (a week before the oral examination), a week dedicated for all the M.Tech. thesis defenses, date for submission of soft copy to the librarian, and a date for final submission of the hardbound thesis to the library.

Appendix-A

This section provides on the details of the Foundation Courses in the curriculum. All the four credit courses will have 3 hour of lecture per week and one hour of tutorials per week throughout the semester.

Algorithms

Objectives:
This course will covered some of the advanced data structures like Fibonacci Heaps, Treaps, AVL and red black trees. It covers the algorithms design techniques like Divide and Conquer, Greedy algorithms and Dynamic Programming. It also covers Graph algorithms including shortest path problem and Minimum Spanning tree and Network flows.

Prerequisite: Basic Data Structures like Arrays, stacks, queues, linked lists, trees, binary trees and travels methods, binary heaps, hashing and graph representation**.**

Contents: The course covers

- Algorithmic analysis : Revive of Asymptotic notations for algorithms, recurrence tree methods, complexity classes
- Abstract Data Structures: Binomial and Fibonacci Heaps, Balanced Binary Search Trees, AVL Trees and Red Black Trees and their applications
- Algorithmic paradigms: Divide and conquer, Dynamic Programming, greedy algorithms including metroid's:
- Graph Algorithms: Graph traversals: DFS and BFS, shortest path problem and the spanning tree problems. Network Flow and applications.
- Randomized Algorithms: Las Vegas and Monte Carlo paradigms, some example randomized algorithms

Text Book / References
• Introduction to Algorithms by Thomas H Cormen, Charles E Leiserson, Ronald L Rivest and Clifford Stein, MIT Press, 3rd Edition 2009.

Discrete Mathematics and Computability

Objectives: This course will provide an introduction to topics from Discrete Mathematics and Computability theory towards pursuing further electives in Computer Science (Theory) and research electives in this area. The course will cover many topics in Combinatorics and also discuss Turing machines, computability and complexity theory, especially Class-P, Class-NP, NP-complete problems and class PSPACE.

Contents:

Discrete Mathematics:

- Propositional logic, sets, functions, relations, partial orders, countability.
- Combinatorics: sum rule, product rule, permutations and combinations, inclusion-exclusion principle, pigeon-hole principle, recursion, generating functions, number of onto functions, partitions and Stirling numbers of second kind.

Computability:

- Computability: Turing machines, equivalent models of Turing machines, decidable and undecidable problems, reductions, Rice's theorem
- Complexity theory: Class P, Class NP, NP-complete problems, PSPACE completeness, Savitch's theorem.

Text Book / References
• Kenneth Rosen, Discrete Mathematics and its applications, 7th edition, McGraw Hill, 2012. • Ronald L. Graham, Donald E. Knuth, and Oren Patashnik, Concrete Mathematics: A Foundation for Computer Science, 2nd edition, 1994. • Ralph P. Grimaldi, Discrete and Combinatorial Mathematics: An Applied Introduction, 5th edition, 2003 • Dexter C. Kozen, Automata and Computability, Springer, 1997. • Michael Sipser, Theory of Computation, Cengage Learning, 2007.

Software Systems

The course has two major modules:

- System Software
- Enterprise Software Development

MODULE 1: SYSTEM SOFTWARE

This module starts with the introduction of the computer architecture, operating system and kernel architecture. Different types of kernel design namely monolithic, micro and hybrid architecture are analyzed. File, process, signals and memory management has been explained with the suitable live examples. Signals, inter process communication and synchronization mechanisms are explained in a practical point of view. The implementation of soft real time
systems according to POSIX standard are analyzed. Finally the difference between application program and kernel module are discussed. During Lab, the students will be asked to write code from scratch for more than 30 real time/live exercise. This comprehensive hands-on course provides the knowledge and skills of system programming and most of the concept such as File, Process, Signals and IPC are compatible with the UNIX variants like UNIX, Linux, Solaris, HP-UX and AIX.

MODULE 2: ENTERPRISE SOFTWARE DEVELOPMENT

In this module, student is exposed to the elements of enterprise software development with primary focus on web application development and mobile application development.

Course Content

MODULE 1: SYSTEMS SOFTWARE

Topic 1: Computer Architecture

- Basic structure of computer hardware and software
- Process, Memory and I/O systems: CPU, RAM, Virtual Memory, I/O devices
- Types of System - Server, Desktop, Embedded and Real Time
- Operating System Vs Kernel

Topic 2: Kernel Architecture

- Kernel Subsystems (computing resource management)
- Types of Kernel: Monolithic, Micro and Hybrid Architecture
- Monolithic - Server and Desktop
- Microkernel - Embedded and Real Time systems
- Hybrid - Handle both RT and Non-RT tasks

Topic 3: System Internals

- Brief implementation of - process, file, memory and signal management
- Communication Mechanisms - pipe, FIFO, message Q, shared memory

Topic 4: Synchronization Mechanisms - File and process

- Implementation of Soft Real Time Systems - as per POSIX standard
- Application Program Vs Kernel Module

MODULE 2: Enterprise Software Development

Topic 1: Fundamentals of Object-oriented Analysis and Design

- OO concepts

- Unified Modeling Language (UML)

Topic 2: Software Architectures

- Understanding large scale systems – MVC, Web architecture, REST, mobile architecture, and hybrid systems along with relevant concepts on reference and enterprise architecture.
- Understanding quality attributes
- Introduction to design and architectural patterns

Topic 3: Web application development

- MVC for Web - Twitter Bootstrap (rendering view), jQuery, Ajax (from jQuery) and servlets (controller), REST service, back-end model - MySql, Java programming and concepts of key value pair (like mongo DB – implemented using MySql)

Topic 4: Mobile application development

- Connectivity, security, online/offline modes, integration of sensors, location services, responsiveness.
- AngularJS and related frameworks

Text Book / References

1. Operating systems - Internals and Design Principles by William Stallings, Prentice Hall. http://dinus.ac.id/repository/docs/ajar/Operating_System.pdf
2. Architect Korner: Challenges in Platform Selection by Gururajan and Thangaraju, Linux for You, June 2010, pp.51-56
3. Software Architecture in Practice by Bass and Clements, Addison Wesley.
4. Ajax - https://www.youtube.com/watch?v=f46WEeM8HTA
5. REST Services - https://www.youtube.com/watch?v=xkKcdK1u95s
6. Jquery Tutorial - https://www.youtube.com/watch?v=8mwKq7_JIS8

Machine Learning

Objectives: This course is intended to be an intense, in-depth course in Statistical Learning methods.

Contents:

Module 1 (Machine Learning, Unsupervised Learning)

- Introduction to Learning from data – Unsupervised, Supervised
- K-means
- Hierarchical Clustering

Module 2 (Supervised Learning, Linear Models)

- Statistical Decision Theory
- Regression: Linear Regression
- Classification: K-NN, Bayes Classifier, Logistic Regression, Linear Discriminant Analysis, Perceptron

Module 3 (Non linear Models, Regularization and Model Selection)

- Non linear Models: Non linear features, Neural networks, Kernels
- Bias-Variance Trade off
- Cross Validation
- Advanced Regression – L2, L1 regularization (LASSO)

Module 4 (Kernel Methods and Support Vector Machines)

- Kernel Trick
- SVM Classification

Suggested Reading:

a) Nello Cristianini and John Shawe-Taylor. "Support Vector Machines". Cambridge University Press. 2000. ISBN 978-0-521-78019-3.
b) Bernhard Scholkopf and Alexander Smola. "Learning with Kernels". The MIT Press. 2012. ISBN 0-262-17475-9.

Module 5 (The Probabilistic Method – Primer on Randomized Algorithms)

1. Common Distributions and their characteristics
2. Chernoff Bounds, Martingales
3. Lovasz Local Lemma
4. PAC (Probably Approximately Correct) Algorithms
5. Probabilistic Decision Theory
6. Frequentist vs Bayesian Approaches

Suggested Reading:

- James O. Berger. “Statistical Decision Theory and Bayesian Analysis”. 2Nd Edition. Springer. 1980. ISBN 3-540-96098-8.

Sheldon M. Ross. “Introduction to Probability and Statistics for Engineers and Scientists”. Elsevier Academic Press 2009. ISBN 13: 978-0-12-370483

Text Book / References
Peter Flach. “Machine Learning” Cambridge University Press. ISBN 978 – 1- 316 – 50611-0 Sergios Theodoridis, Konstantinos. “Pattern Recognition” Academic Press, Fourth Edition. ISBN 978-1-59749-272-0 Christopher M. Bishop. “Pattern Recognition and Machine Learning”. Springer. 2006. ISBN 978-0387-31073-2. Kevin Murphy. “Machine Learning – A Probabilistic Perspective”. The MIT Press 2012. ISBN 978-0-262-01802-9.

Mathematics for Machine Learning

Objectives: This course intends to provide the advanced math background essential for Machine Learning and other advanced courses, and can be viewed as a combination of three main topics: Convex optimization, Advanced Linear Algebra and Advanced Probability.

Contents:

Module 1 (Convex optimization)

- Convex sets, Convex functions
- Unconstrained minimization
- Equality constrained minimization
- Inequality constrained minimization
- Lagrange multipliers
- KKT Multipliers
- Primal form, Dual form

Suggested Reading:

Boyd, Stephen, and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

Module 2 (Matrix Diagonalization)

- Matrix Transformation, Trace, Determinant
- Eigen decomposition
- Singular Value Decomposition
- Moore Penrose Pseudoinverse

Suggested Reading:

a) David C. Lay; Steven R. Lay; Judi J. McDonald "Linear Algebra and its applications, 1997."

b) Golub, Gene H.; Van Loan, Charles F.. "*Matrix Computations*" (3rd ed.), Johns Hopkins, 1996. ISBN 978-0-8018-5414-9.

Module 3 (Random Walks and Markov Chains)

- Random Walks, Markov Chains – properties, stationarity, convergence
- Markov Random Fields
- Linear Dynamical Systems – Kalman Filters

- MCMC Sampling
- Applications – Simulated Annealing, Page Ranking

Suggested Reading:

α) Lazlo Lovasz. "Random Walks on Graphs: A Survey". In: Combinatorics — 'Paul Erdos is Eighty'. Vol. 2. Bolyai Society Mathematical Studies. Keszthely (Hungary): Bolyai Society, 1993, pages 1–46.

β) Olle Haggstrom. "Finite Markov Chains and Algorithmic Applications". Cambridge University Press 2002, ISBN 0 511 01941 6 virtual (netLibrary Edition).

χ) Ross Kindermann and J. Laurie Snell. "Markov Random Fields and Their ApplicationsMarkov Random Fields and their Applications" (1980) American Mathematical Society, ISBN 0-8218-5001-6

δ) Eds. Andrew Blake, Pushmeet Kohli, Carsten Rother. "Markov Random Fields For Vision And Image Processing". MIT Press. July 2011. ISBN: 9780262015776. Introductory Chapter on MRF.

Module 4 (Latent Variable Models)

- Expectation Maximization
- Gaussian Mixture Models (GMM)
- Hidden Markov Model (HMM)

Suggested Reading:

Kevin Murphy. "Machine Learning – A Probabilistic Perspective". The MIT Press 2012. ISBN 978-0-262-01802-9.

Networking and Communication

Objectives: This is an era of networking between computers, mobiles and Internet of Things. Hence, this course covers the fundamental concepts of networking and communication. At the completion of the course, the student should be able to understand the following topics:

- End to end Architecture (topology + protocols) of Data networks.
- Functionalities of various layers in ISO model and interaction between them.
- Principle aspects of communication

Main Modules:

- Data Network Architecture (3 week): In depth conceptual understanding of all the topologies and layers of the ISO models and the associated protocols.
- Application Layer (2 weeks): HTTP protocol, SMTP protocol (email), DNS, socket programming
- Transport Layer (3 weeks): TCP concepts, UDP concepts, congestion/flow control, multiplexing/de-multiplexing
- Network Layer (2 weeks): Routing algorithms, internet signalling, IP addressing
- Datalink Layer (2 weeks): ARP protocol, MAC protocol, error correction/detection
- Physical Layer (3 weeks): Channel capacity, modulation and basics of FEC.

Analog CMOS VLSI Design

Prerequisites : Kirchoff's Laws(KCL/KVL) in electrical networks, Linear circuits: Thevenin/Norton theorems, phasor analysis. Some exposure to diodes/transistors, biasing and small-signal analysis would be useful.

The course has two objectives :

(1) To introduce how CMOS VLSI chips are fabricated (VLSI Technology)

(2) To explain how robust Analog MOS circuits can be designed with a good understanding of VLSI Technology and MOS Device Physics.

The course will discuss how Analog circuits are designed in a VLSI chip environment starting from an understanding of VLSI technology and fabrication. The methodology adopted for teaching this course is to first provide a simple physical model of the MOSFET transistor that is capable of abstracting the essential electrical behavior of the device. Following this a related small-signal MOSFET model can be derived. The application of DC and small-signal analysis methods on MOSFET circuits can then follow.

The main aim of the course will be to learn how to analyze and build CMOS amplifiers that are the building blocks of almost all VLSI mixed-signal systems. At every stage of the course the students are expected to design, on paper as well as simulation, the circuits discussed in the class. An important aspect of the course will be a project in which the students are expected to design and simulate (using Spice simulator).

Topics : VLSI Technology, MOS device physics, Common-source, common-gate, common-drain, and cascode stages, Differential amplifiers, Current mirrors, Frequency response of amplifiers, One and two-stage operational amplifiers, Stability and frequency compensation, feedback networks, Memory design. The course will be useful for those interested in VLSI Design, mixed-signal embedded hardware and is a pre-requisite for RF Design.

References:

1. CMOS : Circuit Design, Layout and Simulation, R. Jacob Baker, IEEE Press/Wiley Student Edition.
2. Silicon VLSI Technology Fundamentals, Practice and Modeling, J. D. Plummer, M. D. Deal, and P. B. Griffin

Digital CMOS VLSI Design

Topics : The theory part includes CMOS logic, latches, flip-flops, CMOS layout, MOSFET Current and Capacitances, Non-ideal MOSFET Effects, CMOS Delay Estimation, Logical effort, Delay optimization and logical effort, Power estimation: Static and Dynamic, Low-Power design, Static Combinational CMOS Logic Styles, Dynamic Combination CMOS Logic styles, Static and Dynamic Sequential Circuit Design, Technology scaling, and VLSI design methodologies. The course will include a lab component of 1credit hour. This will require students to spend 2 hours per week in the lab. Lab component includes Schematic and layout of Digital circuits using Electric. HDL simulation, and synthesis using Mentographics/Xlinix/LASI digital design software tools. Digital prototyping on FPGA board is also included in this course.

References:

1. Neil H. E. Weste and David Harris, CMOS VLSI Design: A circuits and systems perspective, 4^{th} edition, 2011.
2. Verilog HDL: A guide to digital design and synthesis, S. Palnikar, 1996.
3. J. Rabaey, A. Chandrakasan, and B. Nikolic, "Digital Integrated Circuits," 2nd Edition, Pearson Education, 2003.

Principles of Embedded Systems (2 Credits)

Description : Embedded systems are everywhere and most of the electronic systems have a computer inside to do smart things. Due to great demand a large number of embedded systems are available in the market from many companies. Purpose of this course is to help students understand existing architectures of embedded systems and also understand principles involved in designing such systems. In this course we will learn various issues involved in designing embedded systems meeting performance, cost, physical size and weight as well as power consumption requirements. Complex algorithms, user interface along with real time constraints make embedded computing more challenging than normal computing without any constraints on time. The course will start with Shannon's paper on switching circuits, simple microcontrollers and all the way up to distributed embedded computing. In order to understand the engineering aspects better each student or groups of students will study one of the existing platforms and share the knowledge with the class and also do some experiments on embedded systems. The course will involve more open discussions to discover principles and lab to get hands on experience in working with embedded systems.

Topics : Relay circuits, Boolean Algebra, Gates, Shift Registers, CPUs, Memories and Busses, Complex systems and Microprocessors, Embedded system design process and Formalisms for design, Instruction sets, CPU and Memory, I/O Devices and Component Interfacing, Program Design , Analysis and Optimization, Operating systems with real time constraints, Design Methodologies and Architecture design, Power management techniques for single and multi core systems, Multi core Embedded systems , Future Embedded systems, Neural computers and Quantum computers.

References:

1. Computers as Components, Principles of Embedded Computing System Design, Wayne Wolf, Princeton University, Morgan Kauffman Publishers, Academic Press, 2001
2. IEEE Papers as required
3. Published material from TI, ADI, ARM, Intel and others
4. Software Development for Embedded Multi-core Systems: A Practical Guide Using Embedded Intel Architecture, Max Domeika

System Software (2 credits)

This course starts with the introduction of the computer architecture, operating system and kernel architecture. Different types of kernel design namely monolithic, micro and hybrid architecture are analyzed. File, process, signals and memory management has been explained with the suitable live examples. Signals, inter process communication and synchronization mechanisms are explained in a practical point of view. The implementation of soft real time systems according to POSIX standard are analyzed. Finally the difference between application program and kernel module are discussed. During Lab, the students will be asked to write code from scratch for more than 30 real time/live exercise. This comprehensive hands-on course provides the knowledge and skills of system programming and most of the concept such as File, Process, Signals and IPC are compatible with the UNIX variants like UNIX, Linux, Solaris, HP-UX and AIX.

Course Content
MODULE 1: SYSTEMS SOFTWARE
Topic 1: Computer Architecture

- Basic structure of computer hardware and software
- Process, Memory and I/O systems: CPU, RAM, Virtual Memory, I/O devices
- Types of System - Server, Desktop, Embedded and Real Time
- Operating System Vs Kernel

Topic 2: Kernel Architecture

- Kernel Subsystems (computing resource management)
- Types of Kernel: Monolithic, Micro and Hybrid Architecture
- Monolithic - Server and Desktop
- Microkernel - Embedded and Real Time systems
- Hybrid - Handle both RT and Non-RT tasks

Topic 3: System Internals

- Brief implementation of - process, file, memory and signal management
- Communication Mechanisms - pipe, FIFO, message Q, shared memory

Topic 4: Synchronization Mechanisms - File and process

- Implementation of Soft Real Time Systems - as per POSIX standard
- Application Program Vs Kernel Module

Text Book / References
1. Operating systems - Internals and Design Principles by William Stallings, Prentice Hall.
http://dinus.ac.id/repository/docs/ajar/Operating_System.pdf
2. Architect Korner: Challenges in Platform Selection by Gururajan and Thangaraju, Linux for You, June 2010, pp.51-56

System Design using FPGA (2 credits)

This course covers the use of the hardware description language- Verilog for the design of digital integrated circuits and covers the basics of programming the design on to the field programmable devices (FPGA). We will first review in detail the basics of Verilog programming which includes: data types, operators, expressions and signal assignments, modules, nets and registers, the different styles of modeling (gate level, behavioral, data-flow) and testbench design.
Second, we focus on the basic building blocks of FPGA programming, architecture, design methodologies and best design practices.
Students will simulate/verify the design with testbenches and implement the design on to an FPGA development board (Xilinx).

Course Contents
Module 1: Introduction to Hardware description language/Verilog programming
Week 1: Overview of Digital Design with Verilog HDL, Top-down and bottom-up design methodology
Week 2: Data types, system tasks, modules and ports, compiler directives
Week 3: Gate-Level Modeling, Data flow modeling which include Gate level primitives, Continuous assignments, delay specifications.
Week 4: Behavioral modeling: blocking and nonblocking statements, procedures, conditional statements. Testbench design
Module 2: System Design using FPGA
Week 5- Introduction to FPGA, CPLD, ASIC ; Concept of logic blocks, interconnections, bitstreams. Overview of Xilinx or Altera FPGA architecture.
Week 6- Introduction to standard FPGA design flow using Xilinx Vivado Design Suite/Altera Quartus Prime development Suite.
Week 7- Programming the FPGA development boards: Write RTL, Simulate, perform timing analysis. Hands on- Configuring/Programming the FPGA.
Week8 - Projects- Programming the FPGA development boards: Design, optimize and simulate a digital application
Tools:
icarus verilog or Xilinx ISE– to simulate HDL
Xilinx Vivado development Suite – to program the Xilinx FPGA
Xilinx Basys 3 Artix-7 FPGA Board

Text Book / References

1. Palnitkar, Samir, *Verilog HDL, Second Edition*, Prentice Hall.
2. Verilog HDL: A Guide to Digital Design and Synthesis, By Samir Palnitkar
3. Verilog® HDL Quick Reference Guide – by Sutherland
4. Vivado Design Suite Tutorial, [suggested readings - handbook]
5. Vivado Design Suite User Guide [suggested readings - handbook]