M.Tech. Programme Curriculum

(Effective from academic year 2016-17)

International Institute of Information Technology
Bangalore – 560100
January 2016

Table of Contents

1 Overall M.Tech. Programme Structure

Tables 1 and 2 provide a summary of the credit distribution in the M.Tech. programme.

Table 1: Overview of the curriculum

Preparatory Term (3 weeks)	**0 credits** • Introductory Programming (C and Java) • (PASS / FAIL mandatory courses)
Semester 1 (15 weeks)	**16 credits** **For M.Tech (IT)** • core courses Three Courses (8 credits) mandatory • Balance 8 credits to be acquired by choosing courses from a list **For M.Tech (ESD)** • 16 credits must be chosen from 5 courses worth 20 credits •
Semester 2 (15 weeks)	**16 credits** • 4 electives **0 credits** • Technical Communication for those found deficient in a test conducted in Semester 1 • Pass /Fail
Semester 3 (15 weeks)	**16 credits** • 4 electives
Semester 4 (26 weeks)	**16 credits** • Internship / Thesis
Total	**64 credits**

Table 2: Credit Distribution

Proposed	Credits	%
IT Core Course Credits	16	25 %
Elective Course Credits	32	50 %
Internship / Thesis Credits	16	25 %
Total credits requirement for M.Tech.	**64**	

2 Areas of specialization

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

- Computer Science and Engineering (CSE)

- Data Science(DS)
- Networking & Communication and Signal-processing (NCS)

The M.Tech. (ESD) curriculum has two areas of specialization:

- System on Chip (SoC)
- Embedded Systems

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. The preparatory term has two courses in programming (covering C and Java). The two courses will not carry any credit. However, they are mandatory courses with a PASS/FAIL grade. These Programming courses will be taught with emphasis on hands-on programming exercises and projects.

4 IT Core courses

Core courses are those that all the students must take in the prep and first semester. The complete list of core IT courses is provided in Table 1 below.

Table 3: List of core courses for M.Tech. (IT)

Course	Credits	Semester
Introductory Programming 1	0	Prep term
Introductory Programming 2	0	Prep term
Data Structures and Algorithms	4	Semester 1
Operating Systems	2	Semester 1
Database Systems	2	Semester 1
Choice List (students have to choose courses from the list below totaling at least 8 credits and at most 12 credits)	8	Semester 1
Computer Networking & Communication	*4*	*Semester 1*
Software Engineering theory and Practice	*4*	*Semester 1*
IT Project and Product Management	*4*	*Semester 1*
Discrete Mathematics	*2*	*Semester 1*
Probability and Statistics	*2*	*Semester 1*
Linear Algebra	*2*	*Semester 1*

Table 4: List of core courses for M.Tech. (ESD)

Course	**Credits**	**Semester**
C Programming	0	Prep term
Principles of Electronics	0	Prep term
Choice List (students have to choose courses from the list below totaling at least 16 credits	16	Semester 1
Mathematics for ESD	*4*	*Semester 1*
Analysis and Design of Digital ICs	*4*	*Semester 1*
Analog CMOS VLSI Design	*4*	*Semester 1*
Principles of Embedded System Design (2 credits) + Operating systems (2 credits)	*4*	*Semester 1*
Data Structures and Algorithms	*4*	*Semester 1*

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 with the exception of elective courses from the Information Technology and Society area of specialization, which will be offered as open electives.

Each student is required to take at least **five** electives from his/her area of specialization. The remaining three courses can be chosen either from the area of specialization or from open electives. For example, for a student from the Computer Science and Engineering area of specialization, all electives that are listed under Networking Communication and Signal-processing area of specialization (and not cross-listed under Computer Science and Engineering) will be considered as open electives.

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 core course from the Choice List as a pre requisite.

6 Project Electives /Supervised Reading

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/Internship

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

For the students pursuing Internship:

- Internships to be considered as six months (not less than five months) of supervised project work carried out at industry or academic institutions.
- The internship 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 internship completion the internship 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.

For students pursuing thesis, 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 core courses in the curriculum. Each subsection below contains details regarding the various core courses. Elective courses topics will be given by the respective faculty member(s) and it will be processed through the Senate, before addition to the semester's elective.

Introductory Programming

Students come for M.Tech. from many different branches of Engineering. They all have varying levels of programming knowledge. Good programming skills are recognized as being a minimum pre-requisite for virtually all the courses (both core and elective). The goal of the preparatory term is to give a fast-track introduction to programming. The following table highlights some of the details of the course:

Course Name	Introductory Programming I & II
Term	Preparatory Term
Course Credits	0
Duration	3 weeks

Introductory Programming I C Programming (Morning)		Introductory Programming I Java Programming (Afternoon)	
Session duration	3 hours per day	**Session duration**	3 hours per day
Sessions per week	5	**Sessions per week**	5
Total duration	45 hours (3 weeks)	**Total duration**	45 hours (3 weeks)

Course Objectives: At the end of the course, the students should have knowledge and competencies in the following areas:

- C Programming Language
  - Knowledge on the proper usage of all C programming constructs.
  - Ability to compile, debug and run multi-file C programs in a Linux environment.
- Java Programming Language

- Knowledge on the proper usage of core Java.
- Ability to compile, debug and run Java programs in a Linux environment.

The sessions for the modules are theme-based within the backdrop of a single case study. Students will make individual incremental progress on implementing that case study in each session while the necessary background knowledge needed for that will be handled during the lecture session. While the use of case study is mandatory for teaching this course, the instructor is free to design the case study as per his / her interests. The only requirement is that the case study should be designed for incremental development so that the student can be guided to complete it by following the module-wise schedule of the course.

The course is divided into multiple **modules**. Each module is comprised of **lecture session**(s) and **lab session**(s). A session typically has a one hour lecture followed by a 2 hour lab every day. As mentioned earlier, the sessions for the modules are going to be theme-based within the backdrop of a single Case Study. Considering that there are 15 sessions available and 10 modules, instructor may choose devote more time on certain important modules as he/she deems fit.

C Programming:

The C Programming labs will be based on Linux programming environment. In addition to learning the C programming language, the students are expected to gain good user-level experience and skills of the Unix operating system. The following topics will be covered during the course. The topics are grouped into modules and could be used as a guideline during lectures.

- Preliminaries: Introduction to Unix, Introduction to case study
- Data types and expressions: Variables and data types, scope and lifetime of variables, type casting and data type conversion, expression evaluation
- Control flow: if statement, if-else statement, switch-case statement, for loop, while loop, do-while loop
- Functions: User-defined functions, parameters and return values, global variables, static variables, multi-file programming, introduction to built-in libraries (math.h, string.h, etc.).
- Recursion: Recursion for divide-and-conquer
- Arrays: 1-d array, 2-d array and n-d array
- Pointers: Pointers and addresses, pointers and function arguments, pointers and arrays, address arithmetic, character pointers and functions.
- More on pointers: Pointer arrays, pointers to pointers, pointers to functions.
- Structures: Basics of structures, structures and functions, arrays of

structures.
- Advanced structures and unions: Pointers to structures, self-referential structures, unions, bit-fields
- File I/O: Text I/O sequential access, binary I/O sequential access, binary I/O random access

Java Programming:

The following topics will be covered during the course.

- Module 1: What is Java, JVM, WORA, Java programming environment, Eclipse
- Java Basic Syntax: Variables, scope and lifetime of variables, arithmetic expressions, control structures
- OOP with Java: Class and object, methods, instance variables, constructors, data hiding and encapsulation, exception handling
- Class Relationships: Inheritance and polymorphism, method overriding and overloading, associations (1-1, 1-N, M:N), composition, interfaces and abstract classes
- The Java library: Text and numbers, collections, input / output, utilities
- GUI with JFC/Swing: Swing with Netbeans IDE, Swing components, data transfer
- Serialization and file I/O: Serialization, writing serialized objects to file, de-serializing objects from file
- Multi-threaded programming: Concurrency concepts, concurrency in Java (processes, threads, etc.), Java memory model (atomicity, "synchronized", "volatile"), Java threads, deadlocks
- Network programming: Internet addressing, internet streams, sockets, clients, servers, remote method invocation

Mathematics

Course Objectives: Three courses, of two credits each, are offered in the first semester. They provide an introduction to Mathematical Concepts which are used widely in designing engineering systems.

Discrete Mathematics:

- Basic logic: Propositional logic: logical connectives, truth tables, normal forms (conjunctive and disjunctive), predicate logic, modus ponens and modus tollens.
- Proof techniques: Notions of implication, converse, inverse, contra-positive, negation, and contradiction, the structure of formal proofs, direct proofs, proof by counter example, proof by contraposition, proof by contradiction, mathematical induction, strong induction.
- Set Theory: Definition of set, relations, equivalence relations and equivalence classes, cardinality and countability.
- Combinatorics: Pigeonhole principle, inclusion-exclusion principle, generating functions, recurrence relations.
- Groups, rings and fields, with application.

Probability Theory and Statistics:

- Sample space, axioms of Probability theory, random variables, distribution function and density function, mean, variance, characteristic function and central limit theorem.
- Law of large numbers, introduction to Stochastic processes with examples.
- Statistical estimation, parametric distribution estimation, non-parametric distribution estimation, optimal detector design and hypothesis testing, Chebyshev and Chernobounds.

Linear Algebra and Matrix Theory:

- Vector spaces, subspaces and bases, norms, inner product spaces, Gram Schmidt orthogonization, linear transformations.
- Matrices, Eigenvalues and Eigen vectors, LU and QR factorization, trace and determinant, quadratic forms and canonical forms, singular value decomposition, least squares problem and Moore Penrose inverse.

SE Theory and Practices

Knowledge of software engineering principles is critical for any IT professional. Students can imbibe and internalize these principles only by applying in a systematic and structured manner. The Software Engineering Practices course is designed with a greater emphasis on hands-on practices of well-known principles. The course is divided two components:

1. Lecture (about 25 hours) - January - April
2. Project (about 6 months duration) - January - June

While the lecture components will cover all essential concepts and principles, the project component will provide an opportunity for the students to actually put the principles into practice. The project component will also help in filling the void created by the absence of projects in the OS, Databases, and Data Structures and Algorithm courses as per the revised curriculum.
Assessments will be done based on about 70% weight given to the project and about 30% weight given to the lecture component, thus emphasizing the importance of practicing what is being taught.
The Software Engineering Practices course is intended to be offered as a fifth course in the second semester because the value of the course is fully realized only when the project component happens in parallel. The following table highlights some of the details of the course:

Course Name	Software Engineering Theory and Practices
Course Credits	4
Lectures hours per week	2
Total number of lecture hours (per semester)	25
Project Duration	6 Months

Course objectives: At the end of the course, students should have knowledge and competencies in the following areas:

- Practical application of project management practices
- Awareness of practices for developing programs with emphasis on quality
- Defining project tasks with guidance from well-defined process models
- Effective management of source code
- Familiarity with basic terminology associated with process models and quality models.

Lecture component (Theory part): The lecture component will be conducted and completed with-in the second regular semester. The following modules are recommended to be covered as a part of the lectures to be taught in class:

- Process Models (3 hours): Waterfall model, spiral model, V model, iterative models, agile methods (Scrum, XP etc.)
- Project management principles (10 hours): Planning, estimation, monitoring, control, reporting
- Testing principles (6hours): Black box testing, white box testing, non-functional testing, testing metrics
- Configuration management (3 hours): Version control, project space and version space
- Software Quality (3 hours): Quality models (CMMi, Six Sigma, ISO), formal reviews, quality metrics (product quality and process quality)

Project Component: The course includes a mandatory project of "reasonable" complexity. The project is intended to be developed and delivered over a period of 6 months in a group of 4-6 students.

Students have to choose a project in one of the following areas:

- Systems software (operating systems and other system-level software development)
- Information Systems (databases, web, OOAD, supply chain, etc.)
- Computing Methodologies[1] (graphics, visualization, image processing, artificial intelligence, machine learning, etc.)
- Networking and Communication (covers mobile, sensor networks, wireless communication, etc.)
- Embedded Systems

Every project necessarily needs a faculty member as a supervisor and mentor. Faculty members can announce and mentor projects one of the following two ways:

- Linking the on-going elective projects with the SE project provided it belongs to one of the above areas.
- Offering independent projects provided it belongs to one of the above areas.

The final evaluation of the project will be done at the end of the six months duration. The evaluation of project component will be based on:

[1]This name comes from ACM classification system (see http://cran.r-project.org/web/classifications/ACM.html)

- Product deliverables (as defined and evaluated by the project mentor).
- SE practices (as defined and evaluated by the course instructors).

In effect, if 100 marks are allocated for the overall course, following is the recommended break up of evaluation:

- Lecture component – 30 marks (exam-based evaluation)
- Project – SE practices – 20 marks (documentation-based evaluation)
- Project – Product deliverables – 50 marks (demo and testing-based evaluation)

Database Systems

The following table highlights some of the details of the database systems course:

Course Name	Database Systems
Course Credits	2
Lectures hours per week	3
Total number of lecture hours (per semester)	24

Course objectives: At the end of the course, the students should have knowledge and competencies in the following areas:

- Understand the principles of conceptual modeling
- Design databases
- Principles of database programming
- Knowledge of DBMS components
- Other data management technologies (e.g., data exchange, in-memory, etc.)

Course contents:

- Information systems (2 hours): Basic concepts (models, schema, data, information, knowledge), elements of information systems, overview of database systems.
- Conceptual modeling (3 hours): Introduction to conceptual modeling, entity relationship models, UML class diagrams.
- Relational databases (11 hours): Relational data model, database design concepts, DB design via OR mapping, relational algebra, SQL tutorial, functional dependencies, overview of normal forms (till BCNF)

- DBMS (7 hours): Components of a DBMS, storage structures – primary, clustering, secondary, multi-level, query processing – overview, query transformation, query evaluation, transaction processing – overview, ACID properties, concurrency control – schedules, serializability, deadlocks.

Application development (3 hours): Database programming (SQL, embedded SQL, JDBC etc.), overview of 2-tier, 3-tier and n-tier architectures

IT Project and Product Management

Course Name	IT Project and Product Management
Course Credits	4
Lectures hours per week	2
Total number of lecture hours (per semester)	45

COURSE OVERVIEW AND OBJECTIVES

Good project and product management has become necessity for large IT service firms. The first part of the course covers the life cycles of software project management including processes, tools and methods, organizational, financial and market analysis. A detailed exposure is given to the students in different knowledge areas of project management including, risk management, human resource management, estimation procedures, scope and requirements management and financial management of projects. Since most of the IT projects span geographical locations, culture, time zones and common protocols for distributed projects will also be discussed. Agile and lean methodologies in project management literature will also be discussed. The second part of the course deals with product management. The specific case of market analysis and the product-market fit will be discussed complemented by cases in the knowledge area. Processes for large scale product management will also be discussed. Most of the product management firms struggle with governance structure, either in terms of co-existing with services part of the business or stand alone. These and regulatory issues relating to IT products in areas such as Intellectual Property, data privacy and security, regulatory arbitrage will also be illustrated with associated cases. Finally, the ethical issues firms face in product and project management will also be discussed.

COURSE CONTENTS

- Project management life cycle, knowledge areas

- Estimation methods and processes
- Human resource planning and management
- Financial planning and management
- Risk analysis
- Project Planning, and Execution
- Project Monitoring & Control: metrics and measurement
- Project Management Tools and methodologies: SCRUM, Agile, lean
- Cases on successful project management processes
- Program management
- Product management life cycle: product-market fit
- Market and competition analysis
- User requirement analysis
- Governance structure for product and project management
- Regulatory issues in IT products
- Intellectual Property laws and guidelines
- Regulatory arbitrage
- Data security and privacy
- Ethics and Social responsibility in product and project management

Operating Systems

The following table highlights some of the aspects of the Operating Systems course:

Course Name	Operating Systems
Course Credits	2
Lectures hours per week	3
Total number of lecture hours (per semester)	24

The following topics will be covered course in the first regular semester:

- General ideas about operating systems
- The evolution of operating systems, types of operating systems
- System calls; user vs. super- user
- Processes and threads, process scheduling and management
- IPC (Inter Process Communication) and the dining philosophers problem
- Race conditions and mutual exclusion, scheduling
- Memory, virtual memory and memory management
- Paging vs. segmentation
- Page replacement algorithms
- Distributed systems, briefly
- Message-passing vs. shared memory
- Logical clocks and the ordering of events, impossibility results and proofs
- Security issues
- Threats, encryption, symmetric vs. asymmetric, attacksfrom within the system and from outside, protection mechanisms
- A selection of advanced and related topics including introduction to RTOS, Mobile OS etc.

There will be no project component in this course.

Networking and Communication

Course Name	Networking and Communication
Course Credits	4
Lectures hours per week	2
Total number of lecture hours (per semester)	45

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.

Data Structures and Algorithms

Objectives: Students passing this course are expected to have basic familiarity with the presented topics.

This document does not mandate a specific teaching style, text book or load distribution across the topics. A suggested ideal time distribution is presented across each topic. Suggested textbook is Cormen et al. Introduction to Algorithms (popularly called CLR earlier – and now CLRS book).

Total credits: 4
Class hours (per week): 3

- Algorithmic analysis (1 week): Asymptotic notations for algorithms, recurrence tree methods, complexity classes
- Abstract Data Structures(5 weeks): Properties of: Arrays, stacks, queues, linked lists, trees, binary trees, heaps, DAGs, balanced trees, hash tables, graphs, regular graphs
- Algorithmic paradigms: Divide and conquer, Dynamic Programming, greedy algorithms (4 weeks): General method of divide and conquer, example divide and conquer algorithms: merge sort, quick sort, Strassen's matrix multiplication, binary search, general method of dynamic programming, relaxation techniques, knapsack problems.
- Search and Traversal (4 weeks): Searching in Binary Trees, Graph traversals: DFS and BFS, backtracking methods, branch and bound techniques with examples from graph algorithms, spanning tree algorithms, Algorithms based on graph cuts.
- Randomized Algorithms (1 week): Las Vegas and Monte Carlo paradigms, some example randomized algorithms.

C Programming

Students come for M.Tech. from Electronics, Electrical and Instrumentation Engineering background. They all have varying levels of programming knowledge. Good programming skills are recognized as being a minimum pre-requisite for virtually all the courses (both core and elective). The goal of the preparatory term is to give a fast-track introduction to programming in C. The following table highlights some of the details of the course:

Course Name	C Programming
Term	Preparatory Term
Course Credits	0
Duration	3 weeks
Session duration	3 hours per day
Sessions per week	5
Total duration	45 hours (3 weeks)

Course Objectives: At the end of the course, the students should have knowledge and competencies in the following areas:

- Preliminaries: Introduction to Unix, Introduction to case study
- Data types and expressions: Variables and data types, scope and lifetime of variables, type casting and data type conversion, expression evaluation
- Control flow: if statement, if-else statement, switch-case statement, for loop, while loop, do-while loop
- Functions: User-defined functions, parameters and return values, global variables, static variables, multi-file programming, introduction to built-in libraries (math.h, string.h, etc.).
- Recursion: Recursion for divide-and-conquer
- Arrays: 1-d array, 2-d array and n-d array
- Pointers: Pointers and addresses, pointers and function arguments, pointers and arrays, address arithmetic, character pointers and functions
- More on pointers: Pointer arrays, pointers to pointers, pointers to functions
- Structures: Basics of structures, structures and functions, arrays of

structures
- Advanced structures and unions: Pointers to structures, self-referential structures, unions, bit-fields
- File I/O: Text I/O sequential access, binary I/O sequential access, binary I/O random access

The course is divided into multiple **modules**. Each module is comprised of **lecture session**(s) and **lab session**(s). A session typically has a one hour lecture followed by a 2 hour lab every day.

Principles in Electronics

The goal of basic electronics preparatory course is to revise and clarify some of the basic concepts in electronics. This will help them to get more confidence in designing circuits and logics in regular semesters. Good electronic designing and trouble shooting skills are required throughout the MTech curriculum. The following table highlights some of the details of the course:

Course Name	Principles in Electronics
Term	Preparatory Term
Course Credits	0
Duration	3 weeks
Session duration	3 hours per day
Sessions per week	5
Total duration	45 hours (3 weeks)

Course Objectives: At the end of the course, the students should have knowledge and competencies in the following areas:

Design of RLC filter circuits, Rectifier circuits; Ebers-Moll model applied to basic transistor circuits, BJTs and MOSFETS amplifier circuits; FET switches. Feedback and operational amplifiers and use of Opamps as amplifiers, peak detector, differentiators, integrator circuits, Schmitt Trigger and comparators. Active filters and Oscillators, TTL and CMOS, Digital gates using MOSFETs, Decoders, Multiplexers, Latch, Flip-Flops, Counters, Registers, Memories, Mealy and Moore machines, Finite State Machine, state equivalence and machine minimization; Algorithmic state machines, Analog/Digital Conversions, and Introduction to different microcontroller boards such as Arduino, Raspberry Pi, Galileo Development boards, Verilog example codes.

The course is divided into multiple **modules**. Each module is comprised of **lecture session**(s) and **lab session**(s). A session typically has a one hour lecture followed by a 2 hour lab every day.

Mathematics for Electronic Systems Design

This course will cover aspects of mathematics relevant to the design and analysis of Embedded Systems, and Semiconductor devices. We will cover aspects of discrete mathematics relevant for the analysis of switching circuits such as Boolean Algebra, logics and predicate calculus needed for the analysis of real-time systems, graph theory that is relevant in the analysis of digital circuits and development of EDA tools, probability and statistics needed for reliability analysis, Monte-Carlo simulation, linear algebra and differential equations needed for circuit simulation, understanding CMOS technology and semiconductor physics. Computational geometry will also be included that is needed for robotics.

Electronic Systems Design Practices

Knowledge of engineering principles is critical for any IT professional. Students can imbibe and internalize these principles only by applying in a systematic and structured manner. The Electronic Systems Design Engineering Practices course is designed with a greater emphasis on hands-on practices of well-known principles. The course is divided two components:

3. Lecture (about 25 hours) - January - April
4. Project (about 6 months duration) - January - June

While the lecture components will cover all essential concepts and principles, the project component will provide an opportunity for the students to actually put the principles into practice.

Assessments will be done based on about 70% weight given to the project and about 30% weight given to the lecture component, thus emphasizing the importance of practicing what is being taught. The Electronic Systems Design Engineering Practices course is intended to be offered as a fifth course in the second semester because the value of the course is fully realized only when the project component happens in parallel. The following table highlights some of the details of the course:

Course Name	Electronic Systems Design Engineering Practices
Course Credits	4
Lectures hours per week	2
Total number of lecture hours (per semester)	25
Project Duration	6 Months

Course objectives: At the end of the course, students should have knowledge and competencies in the following areas:

- Practical application of project management practices
- Awareness of practices for developing programs with emphasis on quality
- Defining project tasks with guidance from well-defined process models
- Effective management of source code and design
- Familiarity with basic terminology associated with process models and quality models.

Lecture component (Theory part): The lecture component will be conducted and completed with-in the second regular semester. The following modules are recommended to be covered as a part of the lectures to be taught in class:

- Process Models (3 hours): Waterfall model, spiral model, V model, iterative models, agile methods (Scrum, XP etc.)
- Project management principles (10 hours): Planning, estimation, monitoring, control, reporting
- Testing principles (6hours): Black box testing, white box testing, non-functional testing, testing metrics
- Configuration management (3 hours): Version control, project space and version space
- Software Quality (3 hours): Quality models (CMMi, Six Sigma, ISO), formal reviews, quality metrics (product quality and process quality)

Project Component: The course includes a mandatory project of "reasonable" complexity. The project is intended to be developed and delivered over a period of 6 months in a group of 4-6 students.

Students have to choose a project in one of the following areas:

- System on Chip (Analog, Digital, Process Simulation, Verification, Synthesis, etc)

- Embedded Systems (RTOS, Robotics, Integration of Multimodal Sensors, Prototype for various embedded systems applications such as safety, health, etc)

Every project necessarily needs a faculty member as a supervisor and mentor. Faculty members can announce and mentor projects one of the following two ways:

- Linking the on-going elective projects with the Electronic Systems Design project provided it belongs to one of the above areas.
- Offering independent projects provided it belongs to one of the above areas.

The final evaluation of the project will be done at the end of the six months duration. The evaluation of project component will be based on:

- Product deliverables (as defined and evaluated by the project mentor).
- Electronic Systems Design practices (as defined and evaluated by the course instructors).

In effect, if 100 marks are allocated for the overall course, following is the recommended break up of evaluation:

- Lecture component – 30 marks (exam-based evaluation)
- Project – Electronic Systems Design practices – 20 marks (documentation-based evaluation)
- Project – Product deliverables – 50 marks (demo and testing-based evaluation)

Introduction to CMOS Fabrication and Analog CMOS VLSI Design (4 credits) (SBS/MR)

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

Analysis and design of CMOS Digital IC (4credit hours) (MR/SKR)

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, 4th 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.

Operating Systems (2 credit hours) (PGP/SKR)

The following topics will be covered course in the first regular semester:

- System calls; user vs. super- user
- Processes and threads, process scheduling and management
- IPC (Inter Process Communication) and the dining philosophers problem
- Process synchronization using mutex locks, semaphores, monitors.
- Memory, virtual memory and memory management
- Message-passing vs. shared memory
- Kernel modules: changes and compilation

There will be no project component in this course.

References:
1. A. Silberschatz, P. Galvin, G. Gagne, Operating System Concepts, 9th Edition, John Wiley and Sons, 2012.

Principles of Embedded Systems (2 Credits) (PGP) (Approved Elective)

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

Mathematics for Electronic Systems Design (4 Credits) (SN/SKR/PGP/MDS) (New)

Topics : This course will cover aspects of mathematics relevant to the design and analysis of Embedded Systems. We will cover aspects of discrete mathematics relevant for the analysis of switching circuits such as Boolean Algebra, logics and predicate calculus needed for the analysis of real-time systems, graph theory that is relevant in the analysis of digital circuits and development of EDA tools, probability and statistics needed for reliability analysis, Monte-Carlo simulation, linear algebra and differential equations needed for circuit simulation and computational geometry that is needed for robotics.

References:

1. Discrete Mathematics, Kenneth Rosen.
2. Introduction to Linear Algebra, Gilbert Strang.
3. Differential Equations, P. Blanchard, R.L. Devaney and G,R. Hall
4. Computational Geometry, Algorithms and Applications, M.de Berg, O. Cheong, M. van Kreveld, M. Overmars