NATIONAL INSTITUTE OF ELECTRONICS AND INFORMATION TECHNOLOGY

DEEMED TO BE UNIVERSITY UNDER DISTINCT CATEGORY

(AN AUTONOMOUS INSTITUTION UNDER MINISTRY OF ELECTRONICS & IT, GOVT. OF INDIA)

Curriculum and Syllabi for

M.Tech - Electronics Design and Technology

Main Campus at Ropar and Constituent Units at Aizawl, Agartala, Aurangabad, Calicut, Gorakhpur, Imphal, Itanagar, Kekri, Kohima, Patna and Srinagar

Vision

To prepare fresh electronics engineers and up skill working professionals into proficient Electronics Product Design Engineers by enhancing their expertise in both hardware and software aspects of electronic design. The curriculum covers industry-relevant topics, aligning with the current needs of electronics product development sectors. Graduates of this program will be well-equipped to contribute directly to the industry or pursue applied research in this rapidly evolving field.

Mission

To empower and transform aspiring and working electronics engineers into skilled Electronics Product Design professionals by delivering industry-relevant education in hardware and software design. The program aims to bridge the gap between academic knowledge and industrial needs, fostering innovation, hands-on expertise, and readiness for careers in electronics product development and applied research.

Program Education Objectives

PEO1: Professional Excellence
To equip fresh graduates and working professionals with the knowledge and skills required to become proficient Electronics Product Designers.
PEO2: Industry Readiness
To prepare students for careers in industries involved in electronics product development through industry-relevant training.
PEO3: Research and Innovation
To build a strong foundation for students to pursue application-oriented research and innovation in the domain of electronic systems.

Program Outcomes

Upon successful completion of the program, the learners will be able to:
PO1: Apply Engineering Knowledge
Demonstrate comprehensive understanding of hardware and software aspects of electronic systems design.
PO2: Industry-Relevant Skills
Integrate practical and theoretical knowledge to solve real-world problems in electronics product development.
PO3: Design Proficiency
Design, develop, and test electronic products using current technologies and tools.
PO4: Lifelong Learning and Research Orientation
Exhibit readiness to engage in lifelong learning and pursue research or higher studies in advanced areas of electronic systems.
PO5: Problem Solving and Innovation
Apply analytical skills to design innovative solutions for electronics system applications in industrial contexts.

Program Specific Outcomes

PSO1: Electronics Product Design Competence
To equip students with the skills to conceptualize, prototype, and develop end-to-end electronic products, incorporating both hardware and software design aspects.
PSO2: Industry-Aligned Skillset
To provide hands-on experience in tools and practices widely used in the electronics industry, enabling immediate productivity in professional roles.
PSO3: Research and Application Development
To foster the capability to engage in application-oriented research and to develop innovative solutions in embedded systems and electronics domains.

CATEGORY WISE CREDIT DISTRIBUTION

Category	Category Code	Credits
Open Elective Courses	OE	8
Audit Courses (without grade or credit)	AU	0
Skill / Employment Enhancement Courses (Project/Internship/Seminar/Dissertation/Research)	EE	36
Professional Elective Courses	PE	16
Program Core Courses	PC	20
Total Credits (Common track)		**80**

EVALUATION AND ASSESSMENT

1. Performance of a student in a semester shall be evaluated through continuous Class assessment, Tutorial/Lab assessment, Mid-Semester Examination (MSE) and End-Semester Examination (ESE). Both the MSE and ESE shall be the University examination and will be conducted as notified by the Controller of Examinations (CoE) of the University.

2. The marks for continuous assessment shall be awarded at the end of the semester.

3. The continuous assessment shall be based on assignments, tutorials, paper presentation / quizzes/ viva-voce / flipped classes, lab work / projects / fieldwork and attendance.

4. The MSE/ESE shall be comprising of written papers.

5. The overall assessment of the students will be done as per the following scheme:

S. No.	Assessment Type	Marks
1.	Mid Semester Examination	20
2.	End Semester Examination	40
3.	Continuous Assessment - Practical /Lab/ Tutorial	25
4.	Continuous Assessment - Theory	15

For more details, please refer the applicable ordinance for this programme.

CURRICULUM

Semester 1						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-MTEEDT-101	DASH250002	Advanced Engineering Mathematics	3	1	0	4
PC-MTEEDT-102	DOEE250185	System Design Using Embedded Processors	3	0	2	4
PC-MTEEDT-103	DOEE250005	Advanced Digital System Design	3	0	2	4
PE-MTEEDT-104	Elective	Program Elective				4
EE-MTEEDT-105	DASH250095	Research Methodology and IPR	3	1	0	4
AU-MTEEDT-106	Elective	Audit Elective				0

Semester 2						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-MTEEDT-201	DOAI250002	Advanced Machine Learning	3	0	2	4
PC-MTEEDT-202	DOEE250175	Reconfigurable Computing	3	0	2	4
PE-MTEEDT-203	Elective	Program Elective				4
PE-MTEEDT-204	Elective	Program Elective				4
OE-MTEEDT-205	Elective	Open Elective				4
AU-MTEEDT-206	Elective	Audit Elective				0

Semester 3						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PE-MTEEDT-301	Elective	Program Elective				4
OE-MTEEDT-302	Elective	Open Elective				4
EE-MTEEDT-303	EE	Dissertation Phase I				14

Semester 4						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
EE-MTEEDT-401	EE	Dissertation Phase II				18

Elective Courses

Elective Courses for PE Category						
Course Code	**Course Title**	**L**	**T**	**P**	**Cr**	**For Sem./Code**
DCSE250020	Cloud Computing	3	0	2	4	104
DOEE250090	Embedded Programming	3	0	2	4	104
DOEE250088	Embedded OS and RTOS	3	0	2	4	203
DOEE250115	Hardware Design Verification	3	0	2	4	203
DOEE250121	Internet of Things	3	0	2	4	204
DOEE250207	Wireless Technologies	3	1	0	4	204

Elective Courses for OE Category
Any Scheduled Course as per the Course Schedule of the respective Constituent Unit, and meeting the credit requirements can be chosen, subject to availability of seats.

Elective Courses for AU (Audit) Category					
Course Code	**Course Title**	**L**	**T**	**P**	**Cr**
DASH250023	Constitution of India	2	1	0	0
DASH250027	Disaster Management	3	0	0	0
DASH250034	Energy and Environmental Engineering	3	0	0	0
DASH250047	English for Research Paper Writing	2	0	0	0
DASH250054	Essence of Indian Knowledge and Tradition	2	0	0	0
DASH250085	Pedagogy Studies	2	0	0	0
DASH250086	Personality Development	2	0	2	0
DASH250097	Sanskrit for Technical Knowledge	3	0	0	0
DASH250103	Stress Management by Yoga	3	0	0	0
DASH250106	Value Education	2	0	0	0
DASH250117	Innovative Thinking and Entrepreneurship Skills	1	0	0	0
DASH250118	Intellectual Property Rights	2	0	0	0

The choice of all type of Electives available shall be as per the Course Schedule of the respective Constituent Unit.

Guidelines for Massive Open Online Courses (MOOC) / approved Online Courses

1. Relevant Swayam, MOOC, NPTEL, NIELIT NSQF and any other online courses approved by UGC/AICTE shall also be allowed as Program Electives(PE) & Open Electives(OE).

2. To choose an approved online course as per above as Program Elective, the student shall identify an approved Online Course and submit a request to their department or designated authority in the NDU.

3. The Dean (Academics) or Chairman, Board of Studies, shall approve the online credit course for credit transfer on the recommendation of the Head of the Department, keeping in view academic requirements of the programme and alignment of Course Outcomes with the Programme Objectives.

4. On receipt of approval, the student may enroll and complete the course.

5. A student can choose any approved online course as Open Elective, subject to minimum credit requirements.

6. The student has to submit certificate issued by the institution offering the MOOCs course, along with the number of credits and grades, to get credits transferred into his/her marks certificate issued by NDU. The transfer of credits shall be as per rules and procedure stipulated by AICTE / UGC.

Detailed Syllabus

Course Code	Course Name	L	T	P	Cr
DASH250002	Advanced Engineering Mathematics	3	1	0	4

Course Outcomes:

CO1 : Model real-world uncertainty using probability theory and manipulate random variables in applied contexts.
CO2 : Analyze convergence behavior of sequences of random variables and apply statistical limit theorems in AI/ML.
CO3 : Solve abstract algebraic and vector space problems relevant to machine learning algorithmic design.
CO4 : Perform linear transformations, change of basis, and analyze solvability of linear systems in data-centric applications.
CO5 : Apply numerical methods such as interpolation, numerical differentiation, and integration in data approximation and machine learning tasks.

Module 1:

Probability Theory and Random Variables

Axiomatic definition of probability, Bayes' theorem with applications, Random variables: discrete and continuous, Cumulative distribution function (CDF), Probability Mass Function (PMF), Probability Density Function (PDF), Conditional and joint distributions, independence of random variables.

Module 2:

Expectation, Moment Generating Functions, and Limit Theorems

Expectation and fundamental properties, Moment generating functions (MGF) and characteristic functions (CF), Conditional expectation and covariance matrix, Uncorrelated random variables and joint Gaussian PDFs, Markov inequality, Chebyshev inequalities, Central Limit Theorem (CLT), weak and strong laws of large numbers, Modes of convergence of random variables.
Random Process, Markov Process and Markov Chain.

Module 3:

Algebraic Structures and Vector Spaces

Groups, rings, and fields (basic definitions and examples), Vector spaces and subspaces over a field, Linear independence, span, basis, and dimension, Direct sum of subspaces. System of Equations, Solving system of linear equations using Gaussian elimination.

Module 4:

Linear Transformations

Linear transformations and their matrix representation, Four fundamental subspaces: null space, column space, row space, and left null space, Rank, Rank-Nullity Theorem, Solving systems of linear equations: existence and uniqueness, Transpose of a linear transformation. Orthogonality & Orthonormality, Orthogonalization, Eigen Value, Eigen Vector, Diagonalization of vectors.

Module 5:

Interpolation, Numerical Differentiation & Integration

Lagrange's interpolation, Newton's divided difference interpolation, Newton's forward and backward difference interpolation, Approximation of derivatives using interpolation polynomials, Numerical integration using: Trapezoidal rule, Simpson's 1/3 rule.

Labs / Practicals:

N/A

References:

[1] S. M. Ross, A First Course in Probability, 10th ed., Boston, MA, USA: Pearson, 2018.
[2] D. P. Bertsekas and J. N. Tsitsiklis, Introduction to Probability, 2nd ed., Belmont, MA, USA: Athena Scientific, 2008.
[3] M. H. DeGroot and M. J. Schervish, Probability and Statistics, 4th ed., Boston, MA, USA: Pearson, 2012.
[4] G. Strang, Linear Algebra and Its Applications, 5th ed., Boston, MA, USA: Cengage Learning, 2016.
[5] S. Axler, Linear Algebra Done Right, 3rd ed., Cham, Switzerland: Springer, 2015.
[6] K. E. Atkinson and W. Han, Elementary Numerical Analysis: Numerical Methods, 3rd ed., Hoboken, NJ, USA: Wiley, 2003.
[7] J. L. Gross and J. Yellen, Handbook of Discrete and Combinatorial Mathematics, 2nd ed., Boca Raton, FL, USA: CRC Press, 2017.

Course Code	Course Name	L	T	P	Cr
DOEE250185	System Design Using Embedded Processors	3	0	2	4

Course Outcomes:

CO 1 Comprehend Embedded Processor and its software
CO 2 Design an Embedded system with ARM microcontrollers
CO 3 Design an Embedded system using processors, memory I/O devices and communication network within realistic constraints
CO 4 Comprehend ARM Cortex M4 microcontrollers
CO 5 Comprehend the peripheral programming of ARM cortex M4 Microcontrollers
CO 6 Comprehend advanced Embedded Controllers, Features and case studies
CO 7 Perform ARM Cortex M4 Microcontroller configuration for various applications using peripheral devices
CO 8 Demonstrate an embedded system on ARM Cortex M4 Microcontroller development board

Module 1:

Introduction to Embedded Systems
Overview of embedded system architecture, Development and debugging Tools for Embedded Systems, Overview ARM Architecture - Architecture Versions, Instruction Set Development,Thumb-2 and Instruction Set Architecture, AMBA, AXI bus overview. Overview of the ARM Cortex-Mx Processor Architectures.

Module 2:

ARM Cortex M4 Microcontroller System
ARM Cortex M4 Core, Interconnect Matrix in ARM Cortex M4 Microcontroller, System configuration Controller, NVIC, External Interrupt Controllers, DMA, Reset and Clock Control, Clock Recovery System, Power Control.

Module 3:

ARM Cortex M4 Microcontroller Peripheral Overview
Introduction to ARM Cortex-M4 Programming, overview of CMSIS, GPIO, ADC, DAC, Communication & Peripherals - USART, UART, I2C, SPI, USB, CAN
Watchdogs and Timers, PWM.

Module 4:

Memory, Safety and Security in ARM Cortex Microcontroller
Flash, Quad SPI Interface, Flexible Memory controller, CRC, Random Number Generator, memory protections, Advanced Encryption Standard HW Accelerator (AES), Safety support.

Module 5:

Advanced Embedded Controllers, Features and case studies
Programming for Power-Efficient Computing - High Level and low level Techniques, Cortex M7, M23 and M33 Controllers and Features, Overview of mbed platform, Embedded Systems case studies - Consumer, Medical, Automotive.

Labs / Practicals:

Embedded C Programming on ARM Cortex M4 Microcontroller with CMSIS/HAL libraries, Embedded C Programming - Peripheral Programming - GPIO, ADC, DAC, USART, Timers and PWM, Design of a real-time data acquisition & control system using the ARM Cortex M4 Microcontroller

References:

1. Yiu J. The Definitive Guide to ARM Cortex M3 and Cortex M4 Processors, 3rd Edition, Elsevier
2. Andrew N Sloss, Dominic Symes, Chris Wright, “ARM System Developer's Guide - Designing and Optimizing System Software”, 2006, Elsevier

3. Steve Furber, “ARM System-on-Chip Architecture”, 2nd Edition, Pearson Education
4. Raj Kamal, “Microcontroller - Architecture Programming Interfacing and System Design” 1st Edition, Pearson Education
5. P.S Manoharan, P.S. Kannan, “Microcontroller based System Design”, 1st Edition, Scitech Publications
6. Cortex-M4 Technical Reference Manual (TRM)
7. STMicroelectronics Nucleo-G4xx development board user manual

Course Code	Course Name	L	T	P	Cr
DOEE250005	Advanced Digital System Design	3	0	2	4

Course Outcomes:

CO1: Understand and apply number systems, including two's complement and IEEE 754 floating-point formats, for performing basic and advanced arithmetic operations.

CO2: Design and analyze combinational logic circuits using functional blocks such as encoders, decoders, multiplexers, adders, and comparators, considering timing and hazard analysis.

CO3: Develop and optimize sequential logic systems using latches, flip-flops, and finite state machines (Mealy and Moore), with timing analysis and practical design examples.

CO4: Construct and implement digital subsystems including ALUs, shifters, pattern detectors, and arbiters, demonstrating knowledge of modular digital design.

CO5: Analyze and apply digital logic testing concepts such as fault modeling, fault simulation, test generation, and Design for Testability (DFT) and BIST strategies.

CO6: Simulate and implement combinational and sequential digital circuits using Verilog HDL, including multiplexers, counters, code converters, and flip-flops.

CO7: Design and test integrated digital components such as ALUs and counters in Verilog HDL, using simulation tools to verify functionality and timing behavior.

Module 1:

PROCESSOR ARITHMETIC
Two's Complement Number System – Arithmetic Operations Floating Point Number system – IEEE 754 format & POSIT, Basic binary codes.

Module 2:

COMBINATIONAL LOGIC DESIGN
Functional blocks – Decoders, Encoders, Three-state devices, Multiplexers, Parity circuits, Comparators, Adders, substractor, carry look- ahead adder, timing analysis. Combinational multiplier structures, Timing hazards.

Module 3:

SEQUENTIAL LOGIC DESIGN
Latches and Flip-Flops, Sequential logic circuits – timing analysis (Set up and hold times) Synchronizers and met stability, State machines – Mealy & Moore machines, Analysis, FSM design using D Flip-Flops, FSM optimization and partitioning; FSM Design examples: Vending machine, Traffic light controller, Washing machine.

Module 4:

DIGITAL SUBSYSTEMS
ALU, 4-bit combinational multiplier, Barrel shifter, Simple fixed point to floating point encoder, Dual Priority encoder, Cascading comparators .Pattern (sequence) detector, Programmable Up-down counter, Round robin arbiter with 3 requesters, Process Controller, FIFO.

Module 5:

DIGITAL LOGIC TESTING
Introduction to digital logic testing, Fault modelling, fault collapsing, fault simulation, test generation ,Introduction to Design For Testability(DFT),DFT and Built-In-Self-Test(BIST).

Labs / Practicals:

Hands-on Sessions Featuring Simulations of Combinational and Sequential Circuits with Verilog HDL.

- Logic Gates
- Multiplexers/Demultiplexers: 2-to-1, 4-to-1,
- Encoders/Decoders: priority and nonpriority encoders, decoders.
- Adders/Subtractors: Building full adders, ripple carry adders, etc.
- Comparators: Creating magnitude comparators.
- ALUs
- Parity Generator & Checker
- Code Converters
- latch an flip-flop
- counters

References:

Text Books

1. Digital Design by Frank,John Wiley and Sons Publishers.
2. Digital Computer Arithmetic Datapath Design Using Verilog HDL by James E. Stine, Springer
3. Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits by M. Bushnell ,Vishwani Agrawal, Springer.

Reference Books

1. Digital Design by M. Morris R. Mano and Michael D. Ciletti.,Pearson Education.
2. Gustafson and Yonemoto. 2017. Beating Floating Point at its Own Game: Posit Arithmetic. Supercomputing Frontiers and Innovations: an International Journal, Volume 4I, ssue 2, June 2017 , pp 71–86, https://doi.org/10.14529/jsfi170206.
3. Digital Design Principles and Practices by John F. Wakerly, Pearson Education.
4. An Introduction to Logic Circuit Testing by Parag K. Lala, Morgan & Claypool Publishers.
5. Digital Systems Testing and Testable Design by Melvin A. Breuer , Arthur D. Friedman, MironAbramovici, Wiley-IEEE Press

Course Code	Course Name	L	T	P	Cr
DASH250095	Research Methodology and IPR	3	1	0	4

Course Outcomes:

CO1: Understand the principles and process of research methodology. (Level 2 – Understand)
CO2: Apply appropriate research design and data collection methods in computing projects. (Level 3 – Apply)
CO3: Analyze and interpret qualitative and quantitative data using statistical tools. (Level 4 – Analyze)
CO4: Understand various forms of intellectual property and apply processes for protection and patent filing. (Level 2 – Understand)
CO5: Develop ethically sound, well-documented research or project reports conforming to academic and industrial standards. (Level 3 – Apply)

Module 1:

Introduction to Research and Research Ethics (10 Hours)
Definition and objectives of research, types of research – fundamental, applied, exploratory, empirical, significance of research in computer applications, research ethics and integrity, plagiarism, citation styles (APA, IEEE), tools for plagiarism check.

Module 2:

Research Design and Data Collection (10 Hours)
Research problem identification and formulation, hypothesis generation, literature review techniques, types of research design: experimental, descriptive, analytical, sampling techniques, primary and secondary data sources, survey design, case study design.

Module 3:

Data Analysis and Interpretation (10 Hours)
Quantitative and qualitative data analysis, measures of central tendency and dispersion, t-test, ANOVA, correlation and regression, chi-square test, software tools: Excel, R, SPSS (demo-level), data visualization and result interpretation.

Module 4:

Technical Writing and Project Report Preparation (10 Hours)
Structure of research papers and reports, abstract, introduction, literature review, methodology, results, and conclusions, referencing tools (Mendeley/Zotero), IEEE/ACM/SCI journal guidelines, thesis and dissertation formatting, proposal writing basics.

Module 5:

Intellectual Property Rights and Patent Process (10 Hours)
Introduction to IPR: patents, copyrights, trademarks, industrial design, Indian and international patent systems (WIPO, TRIPS), criteria for patentability, patent filing process in India (IPINDIA portal), open-source licenses, patent drafting basics, IPR in software and AI innovations.

Labs / Practicals:

1. Identify a research problem and draft a problem statement.
2. Design a survey/questionnaire and analyze sample responses.
3. Use plagiarism checking and reference management tools.
4. Perform statistical analysis (e.g., t-test or correlation) using Excel/R.

5. Prepare a mini technical report with citations and referencing.
6. Explore Indian patent portal and simulate a patent search or draft.

References:

1. C.R. Kothari & Gaurav Garg – Research Methodology: Methods and Techniques, New Age International.
2. Ranjit Kumar – Research Methodology – A Step-by-Step Guide, Sage Publications India.
3. P. Narayan – Intellectual Property Law, Eastern Book Company.
4. Neeraj Pandey & Khushdeep Dharni – Intellectual Property Rights, PHI Learning.
5. Yogesh Kumar Singh – Fundamentals of Research Methodology and Statistics, New Age International.
6. Bharat Bhushan – Research Methodology in Computer Science, Himalaya Publishing.
7. WIPO and IPIndia Manuals – Guidelines for Filing Patents in India.
8. NPTEL MOOC – Research Methodology for Science and Engineering (IITs)

Course Code	Course Name	L	T	P	Cr
DOAI250002	Advanced Machine Learning	3	0	2	4

Course Outcomes:

CO1: Understand the foundational concepts of machine learning, explore various learning paradigms and tasks, and apply essential data preprocessing and evaluation techniques.
CO2: Apply linear and logistic regression, regularization methods, and classification algorithms including SVM, Decision Trees, and Naïve Bayes for predictive modeling.
CO3: Analyze data using clustering algorithms and perform effective feature selection and dimensionality reduction to enhance machine learning model performance.
CO4: Gain knowledge of probabilistic graphical models and implement ensemble techniques to improve model robustness and accuracy.
CO5: Develop and apply deep learning models including CNNs, RNNs, LSTMs, and Transformers for advanced applications in vision, sequence processing, and natural language understanding.

Module 1:

Overview of machine learning: supervised, semi-supervised, unsupervised learning, reinforcement learning.
Types of Machine Learning Tasks: Classification, Regression, and Clustering.
Cross Validation.
Classifier and Regressor Performance Evaluation Measures, ROC Curves.
Cost functions: Definition and Types.
Bias-Variance trade off.
Data Preprocessing: Standardization, Normalization, Label Encoding, One-hot Encoding.

Module 2:

Linear Regression, Least Square Gradients.
Regularization: Ridge and Lasso Regression.
Classification Methods: K Nearest Neighbor, Logistic Regression.
Sigmoid function and differentiation.
Support Vector Machine (SVM): Optimal Separating Hyperplane, Kernel Trick, Kernel Functions.
Bayes Rule, Naïve Bayes Model.
Decision Tree – ID3.

Module 3:

Clustering Methods: K-means Clustering, Hierarchical Clustering Techniques, Density-Based Clustering
Feature Selection Techniques: Entropy, Correlation Coefficient, Chi-square Test, Forward and Backward Selection.
Dimensionality Reduction: PCA, LDA, t-SNE.
Maximum Likelihood Estimation Technique.

Module 4:

Basics of Graphical Models: Bayesian Networks, Hidden Markov Model.
Ensemble Methods: Boosting, Bagging, Random Forest, XGBoost

Module 5:

Neural Networks – Concept of Perceptron and Artificial Neuron.
Weight Initialization Techniques.
Feed Forward Neural Network, Back Propagation Algorithm.
Convolutional Neural Networks (CNNs) for image classification and object detection.
Recurrent Neural Networks (RNNs),Long Short-Term Memory (LSTM) networks Autoencoders for sequential data processing.

Transformer Models (BERT, GPT) for NLP applications.

Labs / Practicals:

1. Implement supervised, unsupervised, and semi-supervised learning on sample datasets and identify the type of machine learning task.
2. Apply cross-validation and evaluate classifier performance using accuracy, precision, recall, F1-score, and ROC-AUC metrics.
3. Perform data preprocessing using standardization, normalization, label encoding, and one-hot encoding and observe its effect on model accuracy.
4. Implement linear, Ridge, and Lasso regression models on a dataset and compare their performance using MSE.
5. Apply K-Nearest Neighbor, Logistic Regression, and SVM classifiers on a dataset and visualize decision boundaries.
6. Train and evaluate a Decision Tree using the ID3 algorithm and a Naïve Bayes classifier on a classification dataset.
7. Implement and compare K-means, hierarchical clustering, and DBSCAN on a dataset using visualization and silhouette scores.
8. Perform feature selection using entropy, correlation coefficient, and chi-square test, and evaluate model accuracy with selected features.
9. Apply PCA, LDA, and t-SNE for dimensionality reduction and visualize data in 2D space.
10. Build a Hidden Markov Model to predict sequences and evaluate performance using real or synthetic sequential data.
11. Train and compare Random Forest, Bagging, AdaBoost, and XGBoost models on a classification problem.
12. Implement a feedforward neural network with backpropagation to classify handwritten digits using the MNIST dataset.
13. Build a Convolutional Neural Network (CNN) for image classification using CIFAR-10 or MNIST and evaluate performance.
14. Train an LSTM model for time-series prediction and use a pre-trained BERT/GPT model for sentiment analysis in NLP.

References:

1. Ethem Alpaydın, Introduction to Machine Learning , MIT Press, 2014.
2. Kevin Murphy, Machine Learning: A Probabilistic Perspective (MLAPP), MIT Press,2012.
3. Han, Jiawei, and Micheline Kamber. Data Mining: Concepts and Techniques. San Francisco: Morgan Kaufmann Publishers.
4. Christopher M. Bishop, “Pattern Recognition and Machine Learning”, Springer, 2006.

Course Code	Course Name	L	T	P	Cr
DOEE250175	Reconfigurable Computing	3	0	2	4

Course Outcomes:

CO 1: Able to select suitable hardware for an application
CO 2: Able to understand the design methodology of micro-processor system on Chip (SoC) buses, memory peripherals on FPGA
CO 3: Build reconfigurable system using FPGAs
CO 4: Able to evaluate hardware accelerator and achieve acceleration factor for a specific application.
CO 5: Perform partial reconfiguration for various applications using peripheral devices.
CO 6: Demonstrate an embedded system on FPGA using IP blocks.

Module 1:

Introduction to Reconfigurable Computing
Reconfigurable Architectures: Classification of Reconfigurable Architectures. FPGA Technology and Architectures, LUT devices and Mapping, Placement and Partitioning. Programming Technology: HDL Based Programming and High level Synthesis using C, Partial Reconfiguration. Intellectual Property Based Design: Soft core, Firm core and Hard Core, Software tools.

Module 2:

System on chip (SoC) system in FPGA devices
Embedded computer organization and methodology of System on chip (SoC) system in FPGA devices. Design challenges and Differences GPP, DSP, ASIC and FPGA based System on Chip platforms. Application profiling and partitioning, FPGAs vs. Multi-core processor architectures. Xilinx Zynq 7000 family programmable SoC (system on chip) in particular - hybrid device with ARM + FPGA architecture.

Module 3:

Overview of AXI Bus protocol
Design of Master and Slave Bus protocols based IPs.Design Metrics, General purpose peripherals (interrupt, timer, clock, DMA etc.) and special purpose peripherals Serial Transmission protocols & Standards, and advanced high speed buses. Debugging methodologies.

Module 4:

Emulating SoC Architectures on FPGAs
Emphasis on different embedded processors and multiprocessor and architectures. Coprocessor creation, hardware design for System-On-a-Chip. Memory and peripheral interfacing. System level design Tradeoffs, Power, Energy, Performance and Area.

Module 5:

High level synthesis and system modeling
Overview of high level synthesis (HLS). Exploration of HLS tools, System Modeling. Models of Computation and System Specification Languages, High Level Computation/Behavioral Synthesis. Application case study like FFT, JPEG.

Labs / Practicals:

Hands-on Sessions on FPGA development board.

1. To design and implement various combinational logic circuits on an FPGA development board

• Basic logic gates, 2x1 Mux, 4x1 Mux, 3 bit Comparators, Half adder, 3 bit Full adder, Ripple Carry adder using Structural Modelling
• Code converters – 4 bit BCD, Gray Code, Gray to Excess-3 code
• Encoders and Decoders

2. To design and implement various sequential logic circuits on an FPGA development board
• D Flip-flop with Reset
• Up, Down, Up-down Counter
• Ring & Johnson Counter
• Traffic Light Controller implementation with FSM
• IP CORE - Binary Counter
• IPCORE - Adder/Subtractor

3. System on chip (SoC) implementation on FPGA Devices : Zynq 7000
• GPIO configuration and control of LEDs on Zynq Board
• Configuration and data transmission with UART on Zynq Board
4. System Modelling with High Level Synthesis
• Matrix multiplication using the concept of HLS. Tool used : Vivado HLS
• FFT process using concept of HLS. Tool used : Vivado HLS

5. Demonstration of debugging methodologies
• Capture and display internal signal waveforms in real-time on the FPGA using Integrated Logic Analyzer & Virtual Input Output IP Cores

References:

Text Books
1. R. Sass and A. G. Schmidt. Embedded Systems Design with Platform FPGAs Principles and Practices. Elsevier Inc, USA, 2010.
2. The Zynq Book: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 All Programmable SoC, Strathclyde Academic Media , UK ,2014.

Reference Books

1. S. Hauck and A. DeHon, Reconfigurable Computing: The Theory and Practice of FPGA-Based Computing, Morgan Kaufmann, 2008.
2. Cardoso, João M. P.; Hübner, Michael (Eds.), Reconfigurable Computing: From FPGAs to Hardware/Software Codesign, Springer, 2011.

Course Code	Course Name	L	T	P	Cr
DCSE250020	Cloud Computing	3	0	2	4

Course Outcomes:

CO1: Explain the evolution, architecture, and key characteristics of cloud computing, including service models (IaaS, PaaS, SaaS) and deployment types (public, private, hybrid).

CO2: Describe various types and implementation levels of virtualization, including CPU, memory, storage, and network virtualization in cloud environments.

CO3: Analyze layered cloud architecture and address design challenges related to inter-cloud resource management and platform deployment.

CO4: Apply distributed and parallel programming paradigms such as MapReduce and Hadoop for cloud-based application development and understand various cloud platforms like AWS, OpenStack, and Google App Engine.

CO5: Evaluate cloud security challenges and solutions, including data and application security, VM security, and risk management strategies in cloud environments.

Module 1:

Introduction to Cloud Computing

Evolution of Cloud Computing, System Models for Distributed and Cloud Computing, NIST Cloud Computing Reference Architecture, Features of Cloud Computing, Cloud Services – IaaS, PaaS, SaaS, Cloud service Providers – Public , Private and Hybrid Clouds.

Module 2:

Introduction to component virtualization

Basics of Virtualization, Types of Virtualization, Implementation Levels of Virtualization, Virtualization of CPU, Memory, I/O Devices, Desktop Virtualization, Server Virtualization, Storage Virtualization, Network Virtualization.

Module 3:

Architectural Design of Compute and Storage Clouds

Layered Cloud Architecture Development, Design Challenges, Inter Cloud Resource Management, Resource Provisioning and Platform Deployment, Global Exchange of Cloud Resources.

Module 4:

Parallel and Distributed Programming Paradigms

Map Reduce, Twister and Iterative MapReduce, Hadoop Library from Apache, Mapping Applications, Programming Support, Google App Engine, Amazon AWS, Cloud Software Environments - Eucalyptus, Open Nebula, OpenStack.

Module 5:

Security Overview

Cloud Security Challenges, Software-as-a-Service Security, Security Governance, Risk Management, Security Monitoring, Security Architecture Design, Data Security, Application Security, Virtual Machine Security.

Labs / Practicals:

1. Exploring Cloud Deployment and Service Models
2. Simulating Public, Private, and Hybrid Clouds
3. Virtualization of CPU and Memory
4. Network and Storage Virtualization
5. Deployment of a Layered Cloud Architecture
6. Resource Provisioning and Scaling
7. Deploying a Web Application
8. Simulating Security Attacks and Defenses in a Cloud VM
9. Implementing Access Control and Encryption on Cloud Storage

References:

[1] R. Buyya, C. Vecchiola, and T. Selvi, Mastering Cloud Computing: Foundations and Applications Programming. New Delhi, India: McGraw-Hill Education, 2013.
[2] A. T. Velte, T. J. Velte, and R. Elsenpeter, Cloud Computing: A Practical Approach. New Delhi, India: McGraw-Hill Education, 2010.
[3] K. Hwang, G. C. Fox, and J. J. Dongarra, Distributed and Cloud Computing: From Parallel Processing to the Internet of Things. Burlington, MA, USA: Morgan Kaufmann, 2012.
[4] T. Erl, R. Puttini, and Z. Mahmood, Cloud Computing: Concepts, Technology & Architecture. Boston, MA, USA: Pearson Education, 2013.
[5] G. Reese, Cloud Application Architectures: Building Applications and Infrastructure in the Cloud. Sebastopol, CA, USA: O'Reilly Media, 2009.

Course Code	Course Name	L	T	P	Cr
DOEE250090	Embedded Programming	3	0	2	4

Course Outcomes:

CO1: Understand fundamental concepts of programming, algorithms, flowcharts, and the basics of computation for effective problem-solving and logical thinking.

CO2: Apply C programming constructs including data types, operators, control statements, functions, arrays, and pointers to develop modular programs.

CO3: Develop and manipulate advanced C structures such as unions, linked lists, and other data structures, with applications in embedded system contexts.

CO4: Demonstrate object-oriented programming concepts using C++, including class hierarchies, templates, and exception handling for robust code development.

CO5: Analyze and apply embedded programming standards such as MISRA C/C++, use compiler directives, and perform code optimization and profiling for reliable and efficient embedded software development.

CO6: To understand Programming & algorithms for problem solving

CO7: To understand the Programming Concepts

CO8:Build Application using High level languages C/C++ Programming

Module 1:

Introduction to Programming & algorithms for problem solving

The Basic Model of Computation, Algorithms, Flow-charts, Programming Languages, Compilation, Linking and Loading, Testing and Debugging, Algorithms for Problem Solving: Decimal Base to Binary Base conversion, Reversing digits of an integer, GCD (Greatest Common Division), LCM etc. , Use case diagrams and UML.

Module 2:

C Programming Basics

GNU Tools: gcc, gdb, gprof, Makefiles, Basic data types, operations, and flow control (decision-making statements), Flow control (loops), typecasting, and computer logic, Switch-case, arrays, and the basics of strings, pointers, functions, storage class.

Module 3:

Advanced C programming for Embedded Systems

Structure and union, Linear and nonlinear data structures, Linked List

Module 4:

Object Oriented Programming concepts with C++
Overview of C++, , Fundamentals of the object-oriented approach, Class hierarchy, Advanced class concepts, Templates, Accessing data and dealing with exceptions.

Module 5:

Embedded Programming Coding standards & Concepts
Coding Standards For Compliance , ANSI C, C89, C95, C99, C11, MISRA C & C++, Profiling & Optimisation

techniques, Pragma , floating-point exceptions rounding multi-precision libraries (GMP, MPFR, MPIR), IEEE754, Static and dynamic Code analysis.

Labs / Practicals:

Develop algorithms/ Flow diagrams for any one from the following
1.Decimal Base to Binary Base conversion
2.Reversing digits of an integer
3. GCD Greatest Common Division)
4. LCM

Familiarisation of UML diagrams With the help of open source UML tools
1. Draw UML diagrams
2. Export diagrams
3. Share diagrams using Eclipse
4. Create new, custom UML elements
5. Build sequence and activity diagrams from plain text

C Programming using Linux platforms
1. Familiarisation of Linux Commands
2. Example usage of gcc compiler with sample C programs
3. C programming covering data types, operators, loops, strings

C Programming using Linux platforms
1. Examples using Pointers using C
2. Examples using functions, pass by value, pass by reference methods.
3. Examples covering the structure and union

Advanced C Programming for Embedded Systems (Linear & Nonlinear data structure implementation using C Language)
1. Write a c program to implement queue and its operations
2. Write a C Program to implement stack and its operation
3. Write a C program to implement singly linked list and its operations
4. Write a C program to implement circular linked lists and its operations.

Class hierarchy C++ programming
1. Write program to understand the class concepts in C++
2. Write C++ program to cover inheritance concepts in C++
3. Write C++ program to cover overloading concepts in C++.

Advanced class concepts using C++ programming
1. Write C++ program to cover virtual functions in C++.
2. Write C++ program to cover Templates, Accessing data and dealing with exceptions.

References:

Text Books
1. C Programming language, Kernighan, Brian W, Ritchie, Dennis M, Prentice Hall PTR
2. “Embedded C”,Michael J. Pont, Addison Wesley Reference

Reference Books
1. The Complete Reference C++, Herbert Schildt, TMH
2. GNU C++ For Linux, Tom Swan , PrenticeHall India
3. Daniel W. Lewis, “Fundamentals of embedded software where C and assembly meet”, Pearson Education, 2002.

Course Code	Course Name	L	T	P	Cr
DOEE250088	Embedded OS and RTOS	3	0	2	4

Course Outcomes:

CO 1 Attain comprehensive understanding of Embedded OS (Linux) fundamentals
CO 2 Attain comprehensive understanding of Embedded Processor and its software
CO 3 Attain comprehensive understanding of Embedded Linux Drivers
CO 4 Attain comprehensive understanding of basics of real-time concepts
CO 5 Attain comprehensive understanding of incorporating RTOS in an embedded system
CO 6 Attain comprehensive understanding of applying the knowledge for developing practical applications of modern real-time systems
CO 7 Get knowledge in Embedded OS (Linux) fundamentals
CO 8 Understand Embedded Linux Internal programming
CO 9 Comprehend Embedded Linux Drivers
CO 10 Learn about the basics of real-time OS Programming concepts

Module 1:

Introduction to Embedded Linux

Overview of Linux OS, Directory structures, basic Linux shell commands, Overview of Systems Calls, Classification of system Calls, Inter Process Communication, Multithreading and Thread Management

Module 2:

Embedded Linux Driver Development

Linux Kernel Module Programming, Character device driver development, USB device driver development, Block, Network and PCI device driver development

Module 3:

Building and Customisation of Linux for Embedded systems

Linux booting procedure, Bootloader, hypervisor, opensbi, qemu, Linux build tools - Buildroot and Yocto

Module 4:

Overview of Real Time OS

Basics of RTOS: Real-time concepts, Hard Real time and Soft Real-time, Differences between General Purpose OS & RTOS, Basic architecture of an RTOS, Scheduling Systems, RTOS Issues – Selecting a Real Time Operating System

Module 5:

RTOS for Embedded Applications

FreeRTOS, Thread creation & Management, Inter thread Communication, Mutual Exclusion, MbedOS, Other real time OS (VxWorks, Azure RTOS, SAFERTOS etc.)

Labs / Practicals:

Embedded Linux shell commands
Embedded Linux System Call programming
Embedded Linux Interprocess Communication
Development of Multithreaded application in Linux
Performing Thread Synchronization and resource protection (Mutex) based applications in Linux
Develop simple Kernel Module Programming
Develop basic character Device driver in Linux
Familiarization of FreeRTOS, Thread creation, synchronization - Port to FreeRTOS to ARM Cortex M4 development board

References:

1. GNU/LINUX Application Programming, Jones, M Tims
2. Sreekrishnan Venkateswaran Essential Linux Device Drivers, Prentice Hall 2008
3. Embedded/Real Time Systems Concepts, Design and Programming Black Book, Prasad, KVK
4. Software Design for Real-Time Systems: Cooling, J E Proceedings of 17the IEEE Real-Time Systems Symposium December 4-6, 1996 Washington, DC: IEEE Computer Society
5. FreeRTOS Reference Manual

Course Code	Course Name	L	T	P	Cr
DOEE250115	Hardware Design Verification	3	0	2	4

Course Outcomes:

CO1: Acquire knowledge of advanced verification techniques, such as constrained random testing, coverage-driven verification, and assertion-based verification.
CO2: Gain proficiency in using industry-standard verification tools and methodologies, including simulation-based verification using hardware description languages (HDLs) such as Verilog and VHDL.
CO3: Develop skills in creating comprehensive verification plans and testbenches for digital hardware designs, covering functional verification, timing verification, and design constraints.

Module 1:

Overview of hardware design verification:
Design Verification using Verilog HDL-types of test benches, writing test cases. Verification Architecture, Test automation. Random verification. Assertions and Coverage. Verification of a decade counter, memory and FIFO.

Module 2:

Hardware Design Verification Language-System Verilog:
Introduction to C/C++, Object orient programming. Overview of Verification Language-System Verilog. System Verilog test benches. Functional verification.

Module 3:

System Verilog Advanced features:
Coverage driven Verification-System Verilog functional coverage. Constraint Random Verification. Assertion based Verification. IP Verification a case study.

Module 4:

SoC Verification:
Overview of SoCs. SoC Verification architecture. Verification planning and phases of verification. Building SoC Verification environment. Writing test cases and test automation. SoC Verification a case study.

Module 5:

Methodology based Verification:
Introduction to verification methodologies. Universal Verification Methodology (UVM), UVM components. Building UVM test bench/verification environment. UVM based IP verification a case study.

Labs / Practicals:

1. Perform directed Verification of a 1024 x 8 SRAM. Use tasks to obtain good code density.
2. Develop Verification plan and Verify 16x8 FIFO. Perform test automation by scripting. Write FIFO assertions.
3. Design a ones counter with the following behavior.

Ones Counter is a Counter which counts the number of one's coming in serial stream. The Minimum value of the count is "0" and count starts by incriminating one till "15". After "15" the counter rolls back to "0". Reset is also provided to reset the counter value to "0". Reset signal is active negedge. Input is 1 bit port for which the serial stream enters. Out bit is 4 bit port from where the count values can be taken. Reset and clock pins also provided.

a) Develop RTL code for ones counter
b) Develop SV Based Verification plan, which includes coverage and assertion plan.
c) Develop interface to interface DUT with Verification Environment.

d) Develop test cases and perform Verification.
4. Develop SV Based verification plan to verify 128 x 8 fifo.
a) Execute the verification plan and complete the verification.
b) Generate Code Coverage Reports and check the percentage of code coverage.
c) Write SV Assertions to improve verification
5. Perform UVM based Verification of 8-bit ALU
6. Analysis and simulation of a SoC Verification Environment.
7. UVM-Based Verification of a UART (Universal Asynchronous Receiver/Transmitter)
8. Formal Verification using SystemVerilog Assertions (SVA)
9. Interrupt Controller Verification
10. Verification of AXI4-Lite Protocol using UVM

References:

1. Spear, C. and Tumbush, G. SystemVerilog for Verification: A Guide to Learning the Testbench Language Features,3 rd Edition, Springer, 2012.
2. Design Verification with 'e': By Samir Palnitkar.
3. Hardware Design Verification: Simulation and Formal Method-Based Approaches: By William K. Lam, Prentice Hall
4. Writing Testbenches: Functional Verification of HDL Models: By Janick Bergeron, Springer

Course Code	Course Name	L	T	P	Cr
DOEE250121	Internet of Things	3	0	2	4

Course Outcomes:

CO 1 Explain the concept of IoT
CO 2 Networking basics for IoT application development
CO 3 Analyze various protocols for IoT
CO 4 Design a PoC of an IoT system using various hardware platforms
CO 5 Apply data analytics and use cloud offerings related to IoT
CO 6 Analyze applications of IoT in real time scenario

Module 1:

Overview of IoT

Introduction to the Internet of Things, IoT system architecture and standards, Networking Basics - TCP/IP, Networking Basics - IP addressing basics (IPV4 and IPV6), Optimizing IP for IoT: From 6LoWPAN to 6Lo, Routing over Low Power and Lossy Networks.

Module 2:

IoT hardware Platforms

IoT Design Methodology – Embedded computing logic – Microcontroller-System on Chips, Hardware platforms for prototyping IoT node- Arduino, Raspberry Pi, NodeMCU, ESP32, ARM Cortex Microcontrollers, IoT mote hardware platforms, Swadeshi RISC V based solutions, Interfacing sensors and actuators with hardware platforms, Developing IoT applications using Raspberry Pi with Python Programming.

Module 3:

IoT connectivity & Protocols

IoT Access Technologies: WiFi, Zigbee, Zwave, Bluetooth,UWB, sub1GHz, LoRaWAN, Sigfox and NB-IoT, Topology and Security of IEEE 802.15.4, 802.15.4g, 802.15.4e, 1901.2a, 802.11ah and LoRaWAN, IoT application level protocols: MQTT, CoAP, XMPP, HTTP/Rest Services, WebSockets.

Module 4:

Data analytics, Cloud and IoT Security

No SQL Databases Vs SQL Databases, Apache web servers, JSON, Open and commercial Cloud solutions for IoT, Python Web Application Frameworks for IoT, IoT data visualisation tools, IoT Security - Need for encryption, standard encryption protocol, lightweight cryptography, Trust models for IoT, ARM Cortex Microcontroller Security, Root Security Services (RSS).

Module 5:

IoT Case studies

Smart Lighting, Smart home, Smart Agriculture, Smart farming, IoT for health care & patient monitoring, Smart and Connected Cities, Building end-to-end smart applications with TinyML.

Labs / Practicals:

Hands-on Sessions on IoT Application Development using IoT Hardware Kits (ESP32/STM32/Rpi etc.)

- IPv4 and IPv6 Implementation
- 6LoWPAN Network Implementation
- Interfacing multiple sensors and actuators with Processor Hardware using different communication protocols and develop an integrated monitoring system.
- Implementation of different IoT connectivity technologies including WiFi, Bluetooth, and BLE for various application scenarios.
- Implementation of MQTT protocol for IoT device communication
- Implementation of CoAP for resource-constrained IoT devices
- Integration of IoT devices with cloud platforms for data analytics

References:

Reference Books

1. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton and Jerome Henry, —IoT Fundamentals: Networking Technologies, Protocols and Use Cases for Internet of Things, Cisco Press, 2017
2. Alessandro Bassi, Martin Bauer, Martin Fiedler, Thorsten Kramp, Rob van Kranenburg, Sebastian Lange, Stefan Meissner, “Enabling things to talk – Designing IoT solutions with the IoT Architecture Reference Model”, Springer Open, 2016
3. VijayMadisetti , ArshdeepBahga, Adrian McEwen (Author), Hakim Cassimally “Internet of Things: A Hands-on-Approach” ArshdeepBahga& Vijay Madisetti, 2014.
4. Gian Marco Iodice, TinyML Cookbook: Combine artificial intelligence and ultralow-power embedded devices to make the world smarter
5. Olivier Hersent, David Boswarthick, Omar Elloumi , —The Internet of Things – Key applications and Protocols, Wiley, 2012

Course Code	Course Name	L	T	P	Cr
DOEE250207	Wireless Technologies	3	1	0	4

Course Outcomes:

CO 1 Understand the different wireless technologies available today
CO 2 An understanding on the functioning of wireless communication systems and evolution of different wireless communication systems and standards
CO 3 An ability to compare recent technologies used for wireless and Mobile communication
CO 4 An ability to explain the architecture, functioning, protocols, capabilities and application of various wireless communication networks
CO 5 An ability to evaluate design challenges, constraints and security issues associated with Ad-hoc wireless networks
CO 6 An ability to explain the Wireless Sensor Networks and Wireless Personal Area Network Technologies

Module 1:

Wireless Communication Fundamentals

RF Basics: Radio Frequency (RF) Fundamentals: Introduction to RF & Wireless Communications Systems, Units of RF measurements, SNR, ISI, Analog& Digital Modulation techniques for Mobile communication, Multiple access techniques, Wireless Antenna basics, OFDM, MIMO.

Module 2:

Cellular & Mobile technologies

The cellular concept - system design issues, Cellular carriers and Frequencies, Channel allocation, Cell coverage, Cell Splitting, Microcells, Picocells, Handoff and outage, Improving coverage and system capacity, Cellular Systems (1G, 2G, 3G, 4G, 5G and beyond 5G), NBIoT, Mobile IP, 6G overview, Software Defined Networking (SDN) , Virtual RAN & Open RAN (VRAN & ORAN).

Module 3:

Mobile Radio Propagation

Reflection, Diffraction. Fading, Multipath Propagation, Channel modeling, Diversity Schemes and Combining Techniques, Wireless Channel Models.

Module 4:

Wireless LAN

Wi-Fi Organizations and Standards: Regulatory Bodies, IEEE, Wi-Fi Alliance, WLAN Connectivity, WLAN QoS& Power-Save, IEEE 802.11 Standards,802.11-2007,802.11a/b/g, 802.11e/h/I,802.11n, 802.11AC, Wi-Fi Hardware & Software: Access Points, WLAN Routers, WLAN Bridges, WLAN Repeaters, WLAN Controllers/Switches, Wireless Topologies, PoE Infrastructure, Wireless signalling, WiFi6, WiFi Security standards.

Module 5:

WSN and WPAN
Wireless Sensor Network (WSN) & Wireless Personal Area Network(WPAN): Introduction to WSN, WSN IEEE standards, WSN Topologies, WSN - Routing protocols, Low Power Lossy networks, RPL, TSCH and 6TiSCH,

Zigbee, Zwave, Thread, Bluetooth 1.0 to 6.0, LoRA&LoRA WAN, WiMaX, 6lowPAN,s igfox.

Labs / Practicals:

N/A.

References:

1. Theodore S. Rappaport, “Wireless Communications: Principles and Practice”, Second Edition, 2002, Pearson Education Asia.
2. David Tse and PramodViswanath, Fundamentals of wireless communications, Cambridge University Press, First Edition, 2012.
3. Henrik Schulz And Christian L¨uders, Theory and Applications of OFDM and CDMA Wideband Wireless Communications, , John Wily & Sons, First Edition, 2005.
4. Bluetooth Revealed; By: Miller, Brent A, Bisdikian, Chatschik; Addison Wesley Longman Pte Ltd., Delhi.
5. Wilson , “Sensor Technology hand book,” Elsevier publications 2005.
6. Andrea Goldsmith, “Wireless Communications,” Cambridge University Press, 2005.
7. Mobile and Personal Communications Services and Systems; 1 st Edition; By: Raj Pandya; PHI, New Delhi.
8. Mobile Communications; By: Schiller, Jochen H; Addison Wesley Longman Pte Ltd., Delhi
9. 3G Networks: Architecture, protocols and procedures based on 3GPP specifications for UMTS WCDMA networks, By Kasera, Sumit, Narang, and Nishit, TATA MGH, New Delhi.
10. Wireless Sensor Networks: information processing by approach, ZHAO, FENG, GUIBAS and LEONIDAS J, ELSEVIER, New Delhi.
11. Holger Karl and Andreas Wiilig, “Protocols and Architectures for Wireless Sensor Networks” John Wiley & Sons Limited 2008.
12. Wireless Communications and Networking, VijayGarg, Elsevier.