

**Bansilal Ramnath Agarwal Charitable Trust’s**

**Vishwakarma Institute of Technology**

*(An Autonomous Institute affiliated to Savitribai Phule Pune University)*

**Structure and Syllabus of**

**B.Tech.**

**Electronics and Telecommunication Engineering**

**Effective from Academic Year 2025-26**

#### Prepared by: - Board of Studies in Electronics and Telecommunication Approved by: - Academic Board, Vishwakarma Institute of Technology, Pune

**Chairman – BOS Chairman – Academic Board**

**Institute Vision**

“To be a globally acclaimed institute in technical education and research for holistic socio- economic development.”

**Institute Mission**

* To ensure that 100% students are employable and employed in Industry, Higher Studies, become Entrepreneurs, Civil / Defense Services / Govt. Jobs and other areas like Sports and Theatre.
* To strengthen Academic Practices in terms of Curriculum, Pedagogy, Assessment and Faculty Competence.
* Promote Research Culture among Students and Faculty through Projects and Consultancy.
* To make students Socially Responsible Citizen.

**Department Vision**

“To be a centre of academic excellence in Electronics, Telecommunication and related domains through continuous learning and innovation.”

**Department Mission**

* + To provide state of art education in Electronics and Telecommunication Engineering to meet current and future needs of society, industry, and academia.
	+ To strengthen collaborations with industries and institutes of repute to foster research culture among faculty members and students.
	+ To promote ethically conscious engineers demonstrating sustainable entrepreneurship and professional maturity in a social context.

**Program Educational Objectives (PEOs)**

Graduates of the program will

1. Have a comprehensive knowledge of Electronics engineering fundamentals to face the challenges of real-life complex problems.
2. Be professionals imbibed with a spirit of leadership, ethical behavior, and societal commitment.
3. Be compliant to constantly evolving technology through lifelong learning.

**Program Specific Objectives (PSOs)**

E&TC Graduates will have the ability to:

1. Design, develop and analyze complex Electronic Systems for communication, Signal Processing, Embedded Systems, and VLSI applications.
2. Identify and apply domain-specific hardware and software tools to solve real-world problems in Electronics and Communication.

**Program Outcomes (POs)**

Engineering Graduate will be able to

|  |
| --- |
| 1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems. |
| 2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences. |
| 3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and the cultural, societal,andenvironmental considerations. |

|  |
| --- |
| 4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions. |
| 5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations. |
| 6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice. |
| 7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development. |
| 8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. |
| 9. **Individual and teamwork:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings. |
| 10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions. |
| 11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments. |
| 12. **Life-long learning:** Recognize the need for, and have the preparation and ability toengage in independent and life-long learning in the broadest context of technological change. |

**Contents**

[**Second Year Structure, Semester 1, AY 2025-26** 7](#_Toc205460008)

[**Structure of Honours in VLSI and Embedded system design** 8](#_Toc205460009)

[**Structure of Honours in Robotics and Automation** 9](#_Toc205460010)

[**Structure of Honors in Deep Learning and Generative Artificial Intelligence** 9](#_Toc205460011)

[**Structure of Double Minor in Embedded Technology** 10](#_Toc205460012)

[**Structure of Double Minor in Intelligent and Autonomous Robotic Systems** 10](#_Toc205460014)

[**ET2301: Data Structures** 12](#_Toc205460016)

[**ET2302: Digital Electronics** 15](#_Toc205460017)

[**ET2303: Solid State Devices and Circuits** 17](#_Toc205460018)

[**ET2304: Signals and Systems** 20](#_Toc205460019)

[**ET2292: Engineering Design and Innovations-III** 23](#_Toc205460020)

[**ET2245: Design Thinking-3** 26](#_Toc205460021)

[**SH2001: Reasoning And Aptitude Development** 28](#_Toc205460022)

[**ETM001: IOT for Smart Applications** 30](#_Toc205460023)

[**ETM002: Microcontroller and Applications** 33](#_Toc205460024)

[**ETH001:Computer Organization and Architecture** 37](#_Toc205460025)

[**ETH002: Microcontroller Applications** 40](#_Toc205460026)

[**ETH003:Analog Mixed Signal Design** 42](#_Toc205460027)

[**ETH004:Digital Integrated Circuit Design** 44](#_Toc205460028)

[**ETH005: Real-Time Embedded Systems** 46](#_Toc205460029)

[**ETH007: Introduction to Robotics** 48](#_Toc205460030)

[**ETH008: Robotic Mechanisms and Control** 52](#_Toc205460031)

[**ETH009: Advanced Robotics and Computer Vision** 55](#_Toc205460032)

[**ETH010: IoT and ML in Robotics** 59](#_Toc205460033)

[**ETH011: Special Topics in Robotics** 63](#_Toc205460034)

[**ETH012: Capstone Robotics Project** 66](#_Toc205460035)

[**ETH013: Data Science** 68](#_Toc205460036)

[**ETH014: Deep Learning for Visual Recognition** 70](#_Toc205460037)

[**ETH015: Deep Neural Networks for Natural Language Processing** 72](#_Toc205460038)

[**ETH016: Large Language Models (LLMs) and Generative AI** 75](#_Toc205460039)

[**ETH017: Edge AI and Cloud-based Systems for Real-Time Deep Learning** 78](#_Toc205460040)

[**ETH018:** **Capstone Project in AI Systems** 82](#_Toc205460041)

[**ETD001: Fundamentals of Digital Electronics** 83](#_Toc205460042)

[**ETD002: Introduction to Microcontrollers and Embedded C** 85](#_Toc205460043)

[**ETD003: Embedded System Interfacing and Applications** 88](#_Toc205460044)

[**ETD004: Embedded System Development with Arduino** 90](#_Toc205460045)

[**ETD005: Advanced Embedded Systems with Raspberry Pi** 92](#_Toc205460046)

[**ETD007: Fundamental of Robotics** 94](#_Toc205460047)

[**ETD008: Control Systems and Embedded Programming for Intelligent Robotics** 97](#_Toc205460048)

[**ETD009: Advanced Robotics: Dynamics, Perception, Artificial Intelligence, and Path Planning** 100](#_Toc205460049)

[**ETD010: Intelligent Robotic Systems: IoT, Machine Learning, Mobile Robotics, and ROS Frameworks** 103](#_Toc205460050)

[**ETD011: Advanced Robotics: Humanoid Design, Nano-Scale Control, AI Integration, and Ethical Governance** 107](#_Toc205460051)

[**ETD012: Capstone Robotics Project** 110](#_Toc205460052)

# **Second Year Structure, Semester 1, AY 2025-26**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr. No. | Course Code | SY (Sem1) | Teaching Scheme(Hrs/Week) | Credits |
|  |  |  | Theory | Lab | Tut |  |
| CORE | ET2301 | DATA STRUCTURES | 3 | 2 | 0 | 4 |
| CORE | ET2302 | DIGITAL ELECTRONICS | 2 | 2 | 0 | 3 |
| CORE | ET2303: | SOLID STATE DEVICES AND CIRCUITS | 2 | 2 | 0 | 3 |
| CORE | ET2304 | SIGNALS AND SYSTEMS | 2 | 0 | 0 | 2 |
| Multidisciplinary  |   | [MULTIDISCIPLINARY MINOR](file:///E%3A%5CAcademic%5CNEP%5CNEP%5CNEP%5CSyllabus%20ver%202%20SY%20sem1%5CMDM%20poolwise%281%29.xlsx) | 2 | 0 | 1 | 3 |
| HS | HS2002 | [FROM CAMPUS TO CORPORATE - 1](file:///E%3A%5CAcademic%5CNEP%5CNEP%5CNEP%5CSyllabus%20ver%202%20SY%20sem1%5CCampus) | 2 | 0 | 0 | 2 |
| CORE | HS2001 | REASONING AND APTITUDE DEVELOPMENT - 3 | 1 | 0 | 0 | 1 |
| CORE | ET2292 | :ENGINEERING DESIGN AND INNOVATION - III |  0 |  0 |  0 | 2 |
| CORE | ET2245 | DESIGN THINKING - 3 | 0 | 0 | 1 | 1 |
|  |   |  |  |  |  | 21 |

## Title: Course Structure-B25, Module: III

|  |  |  |  |
| --- | --- | --- | --- |
|  | - |  | **FF No. 653** |
| **Branch:** E&TC | **Year:** S.Y. | **A.Y.:** 2025-26 | **Module:** V, Sem1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Subject Head** | **Course Code** | **Course Name** | **Theory** | **Lab** | **Tut** | **Credit** | **Theory(W)ESE** | **Theory(O)** | **CVV(oral)** | **Course Project(CP)** | **Lab** | **Home Assignment** | **MSE** | **Test1(w)** | **Test1(online)** | **Test2(W)** | **Test2(Online)** | **ESE** | **Pract+CVV (Oral )** |
| S1 | ET2301 | Data Structures | 3 | 2 | 0 | 4 |   |   |   | 30 | 10 |   |   |   |   |   |   |   | 40+20 |
| S2 | ET2302 | Digital Electronics | 2 | 2 | 0 | 3 | 40 |   | 20 | 30 | 10 |   |   |   |   |   |   |   |  |
| S3 | ET2303: | Solid State Devices And Circuits | 2 | 2 | 0 | 3 | 40 |   | 20 | 30 | 10 |   |   |   |   |   |   |   |  |
| S4 | ET2304 | Signals and Systems | 2 | 0 | 0 | 2 |   |   | 30 |   |   |   |   | 35 |   | 35 |   |   |  |
| S5 | ETM001 | IoT for Smart Applications | 2 | 0 | 1 | 3 |  |  |  |  |  | 30 |   |   | 35 |   | 35 |   |  |
| S6 | ETM002 | Microcontroller and Applications |
|  | HS2002 | [From Campus To Corporate - 1](file:///E%3A%5CAcademic%5CNEP%5CNEP%5CNEP%5CSyllabus%20ver%202%20SY%20sem1%5CCampus) | 2 | 0 | 0 | 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | HS2001 | Reasoning And Aptitude Development - 3 | 1 | 0 | 0 | 1 |  |  |  |  |  |  |  |  |  |  |  | **100** |  |
|  | ET2292 | Engineering Design And Innovation - III |  0 |  0 |  0 | 2 |  |  |  |  |  |  | 30 |  |  |  |  | **70** |  |
|  | ET2245 | Design Thinking - 3 | 0 | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  | **100** |  |
|  |  |  |  |  |  | 21 |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Course Type | Course Code | Course Name | Theory | Lab | Tutorial | Credits |
| 1 | Core | ET2301 | Data Structures | 3 | 2 | 0 | 4 |
| 2 | Core | ET2302 | Digital Electronics | 2 | 2 | 0 | 3 |
| 3 | Core | ET2303 | Solid State Devices and Circuits | 2 | 2 | 0 | 3 |
| 4 | Core | ET2304 | Signals and Systems | 2 | 0 | 0 | 2 |
| 5 | MDM | ETM001 | IoT for Smart Applications | 2 | 0 | 1 | 3 |
| 6 | MDM | ETM002 | Microcontroller and Applications | 2 | 0 | 1 | 3 |

# **Structure of Honours in VLSI and Embedded system design**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Course Type | Course Code | Course Name | Theory | Lab | Tutorial | Credits |
| 1 | **Honours** | ETH001 | Computer Organization and architecture | 2 | 2 | 0 | 3 |
| 2 | **Honours** | ETH002 | Microcontroller and applications | 2 | 2 | 0 | 3 |
| 3 | **Honours** | ETH003 | Analog Mixed Signal Design | 2 | 2 | 0 | 3 |
| 4 | **Honours** | ETH004 | Digital Integrated Circuit Design | 2 | 2 | 0 | 3 |
| 5 | **Honours** | ETH005 | Real time embedded system | 2 | 2 | 1 | 4 |
| 6 | **Honours** | ETH006 | Capstone Project  |  |  |  | 4 |

# **Structure of Honours in Robotics and Automation**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Course Type | Course Code | Course Name | Theory | Lab | Tutorial | Credits |
| 1 | **Honours** | ETH007 | Introduction to Robotics | 2 | 2 | 0 | 3 |
| 2 | **Honours** | ETH008 | Robotic Mechanisms and Control | 2 | 2 | 0 | 3 |
| 3 | **Honours** | ETH009 | Advanced Robotics and Computer Vision | 2 | 2 | 0 | 3 |
| 4 | **Honours** | ETH010 | IoT and ML in Robotics | 2 | 2 | 0 | 3 |
| 5 | **Honours** | ETH011 | Special Topics in Robotics | 2 | 2 | 1 | 4 |
| 6 | **Honours** | ETH012 | Capstone Project in Robotics and Automation  |  |  |  | 4 |

# **Structure of Honors in Deep Learning and Generative Artificial Intelligence**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Course Type | Course Code | Course Name | Theory | Lab | Tutorial | Credits |
| 1 | **Honours** | ETH013 | Data Science | 2 | 2 | 0 | 3 |
| 2 | **Honours** | ETH014 | Deep Learning for Visual Recognition | 2 | 2 | 0 | 3 |
| 3 | **Honours** | ETH015 | Deep Neural Networks for Natural Language Processing | 2 | 2 | 0 | 3 |
| 4 | **Honours** | ETH016 | Large Language Models (LLMs) and Generative AI | 2 | 2 | 0 | 3 |
| 5 | **Honours** | ETH017 | Edge AI and Cloud-based Systems for Real-Time Deep Learning | 2 | 2 | 1 | 4 |
| 6 | **Honours** | ETH018 | Capstone Project in AI Systems |  |  |  | 4 |

# **Structure of Double Minor in Embedded Technology**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Course Type | Course Code | Course Name | Theory | Lab | Tutorial | Credits |
| 1 | Double Minor | ETD001 | Fundamentals of Digital Electronics | 2 | 2 | 0 | 3 |
| 2 | Double Minor | ETD002 | Introduction to Microcontrollers and Embedded C | 2 | 2 | 0 | 3 |
| 3 | Double Minor | ETD003 | Embedded System Interfacing and Applications | 2 | 2 | 0 | 3 |
| 4 | Double Minor | ETD004 | Embedded System Development with Arduino | 2 | 2 | 0 | 3 |
| 5 | Double Minor | ETD005 | Advanced Embedded Systems with Raspberry Pi | 2 | 2 | 1 | 4 |
| 6 | Double Minor | ETD006 | Capstone Project in Robotics and Automation  |  |  |  | 4 |

# **Structure of Double Minor in Intelligent and Autonomous Robotic Systems**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr. No | Course Type | Course Code | Course Name | Theory | Lab | Tutorial | Credits |
| 1 | Double Minor | ETD007 | Introduction to Robotics | 2 | 2 | 0 | 3 |
| 2 | Double Minor | ETD008 | Robotic Mechanisms and Control | 2 | 2 | 0 | 3 |
| 3 | Double Minor | ETD009 | Advanced Robotics and Computer Vision | 2 | 2 | 0 | 3 |
| 4 | Double Minor | ETD010 | IoT and ML in Robotics | 2 | 2 | 0 | 3 |
| 5 | Double Minor | ETD011 | Advanced Robotics: Humanoid Design, Nano-Scale Control, AI Integration, and Ethical Governance | 2 | 2 | 1 | 4 |
| 6 | Double Minor | ETD012 | Capstone Project in Robotics and Automation  |  |  |  | 4 |

 **Template FF No. : 654**

# **ET2301: Data Structures**

**Credits: 4**

 **Teaching Scheme:**

**Theory: 03 Hours / Week**

 **Lab: 02 Hours / Week**

**Section 1: Topics/Contents**

**Introduction to Object-Oriented Programming (OOP):** Fundamental concepts of OOP, including classes, objects, inheritance, polymorphism, and encapsulation. **Basic Concepts of Arrays:** Definition, representation, and operations such as arithmetic traversal, insertion and deletion.

**Sorting Techniques:** Bubble Sort, Insertion Sort, Quick Sort, and Heap Sort with time and space complexity analysis.

**Searching Techniques:** Linear Search and Binary Search with time complexity analysis.

**Linked Lists:** Dynamic memory allocation, Singly Linked Lists, Doubly Linked Lists, Circular Linked Lists.

**Stack:** Representation and implementation using Arrays and Linked Lists. Applications include expression conversion and evaluation.

**Queues:** Representation and implementation using Arrays and Linked Lists. Types of Queues.

**Applications of Stack, Queue, and Linked List:**

**Stack:** Balancing parentheses in an expression, reversing a string, and evaluating postfix expressions.

**Queue:** Implementing a ticketing system, printing job scheduling, and simulating a queue at a service center.

**Linked List:** Managing a to-do list, implementing a simple phone directory, and designing a basic memory allocator.

**Section2:** **Topics/Contents**

**Trees:** This section covers the basic terminology and representation of trees using arrays and linked lists. It includes tree traversals, both recursive and non-recursive methods, and operations on binary trees and binary search trees (BST). Applications of trees are explored through the construction and analysis of Huffman Trees. Advanced trees such as AVL trees, Red-Black trees, B-trees, and B+ trees are introduced along with their analysis.

**Graphs:** Graphs are discussed in terms of terminology and representation using adjacency matrices and adjacency lists. Graph traversals, including BFS and DFS, are covered, along with applications like connected graphs, bipartite graphs, and cycle detection. Algorithms for finding minimum spanning trees, such as Prim's and Kruskal’s, and shortest path algorithms are explained. Union-Find operations are also included, and the applications of graphs are illustrated through the analysis of the Traveling Salesman Problem.

**Hashing:** This topic introduces hashing techniques, hash tables, and hash functions. It covers collision handling and resolution techniques, including dynamic hashing. Applications of hashing are discussed with examples such as password encryption and integrity checks.

**List of Practicals:**

1. Assignment based on Object-Oriented Programming (OOP).
2. Assignment based on Basic Concepts of Arrays.
3. Assignment based on Sorting and Searching.
4. Assignment based on Stack Application (Expression conversion, etc.).
5. Assignment based on Queue Application (Job scheduling, resource allocation, etc.).
6. Assignment based on Linked List.
7. Assignment based on Advanced Linked List (Circular or Doubly Linked List).
8. Assignment based on Trees (binary trees, traversals, Huffman tree).
9. Assignment based on Advanced Trees (AVL, Red-Black trees).
10. Assignment based on Graphs (representation, BFS, DFS, cycle detection, bipartite check).
11. Assignment based on Minimum Spanning Tree and Shortest Path Algorithms (Prim’s, Kruskal’s, Dijkstra’s).
12. Assignment based on Union-Find operations (disjoint sets, cycle detection).
13. Assignment based on Hashing (hash tables, collision handling, dynamic hashing, password encryption).

**List of Project areas:**

1. Simple Contact Manager using Arrays and Linked Lists
2. Basic Calendar App with Event Reminders
3. Maze Solver using BFS and DFS Traversals
4. Auto-complete Feature using Trie Data Structure
5. Simple Spell Checker using Hashing Techniques
6. Library or Inventory Management System
7. Job Scheduling Simulator using Queue and Priority Queue
8. Shortest Path Finder in a City Map using Dijkstra’s Algorithm
9. Student Record Storage and Search System using Hash Tables
10. Classic Snake Game using Stack and Queue for Movement Handling

**Text Books:**

1. Yashavant Kanetkar, Data Structures through C++, BPB Publication, 2nd Edition

2. Data structures, Algorithms and Applications in C++, 2nd Edition, Sartaj Sahni, Universities

Press.

**Reference Books:**

1. Yedidyah Langsam, Moshe J Augenstein, Aaron M Tenenbaum – Data structures using C and C++ - PHI Publications (2nd Edition).
2. Ellis Horowitz, Sartaj Sahni- Fundamentals of Data Structures – Computer Science Press.

**Course Outcomes:**

The student will be able to –

1. Understand and apply fundamental Object-Oriented Programming concepts, including classes, inheritance, and polymorphism.
2. Implement and analyze basic data structures such as arrays, linked lists, stacks, and queues.
3. Implement and analyze efficient sorting and searching algorithms with complexity evaluation.
4. Perform operations and traversals on trees and apply them to practical problem solving.
5. Represent graphs, execute traversal algorithms, and solve problems using graph algorithms.
6. Implement hashing techniques and collision resolution methods for secure and efficient data storage.

 **FF No. : 654**

# **ET2302: Digital Electronics**

**Credits: 03 Teaching Scheme:**

**Theory: 02 Hours / Week**

 **Lab: 02 Hours / Week**

**Prerequisites:** Digital Logic Design and Testing (Course Code: ET1017)

**Section 1:**

**Unit-I: Combinational Logic Design:** Logic simplification using K-map (up to 4 variables), Quine-McCluskey minimization technique, multiplexers and demultiplexers as function generators (Few examples on each topic).

**Unit-II: Sequential Logic Design:** Latches and Flip-flops, Conversion of flip-flops, Shift Registers, Types of shift registers, Counters: Asynchronous Counters, Synchronous Counters, Up/Down Synchronous Counters, Design of Synchronous Counters, modulo-N counters.

**Unit-III:** **FSM:** State transition diagram, State table, Moore and Mealy Machines, Design examples of FSM.

**Section 2:**

**Unit-IV: Digital Logic Families:** Characteristics of Digital ICs: Speed of Operation, Power Dissipation, Figure of Merit, Fan in, Fan out, Current and Voltage Parameters, Noise Immunity, Types of logic families: TTL, Operation of TTL NAND gate, CMOS, Operation of CMOS Inverter, NAND and NOR gates, Introduction to Low-Voltage CMOS (LVCMOS) and Bi-CMOS logic families.

**Unit-V: Programmable Logic Devices:** Simple Programmable Logic Devices (SPLDs): PROM, PAL and PLA. CPLDs and FPGAs, Introduction to HDL.

**Unit-VI: A/D and D/A Converters:** Basics of analog to digital conversion: Sampling, Holding, Quantizing and Encoding. Digital-to-Analog Converters (DACs): Specifications, Binary-Weighted DAC, R/2R ladder DAC. Analog-to-Digital Converters (ADCs): Specifications, Flash ADC, Dual-Slope ADC and Successive-Approximation ADC.

**List of Practical:**

1. Design and implement combinational logic circuit using multiplexer & demultiplexer (Trainer kit or EDA tool)
2. Design and implement BCD (Decade) counter (Trainer kit or EDA tool)
3. Design and implement 4-bit up-down ripple counter (Trainer kit or EDA tool)
4. Design and implement Mod-N counters (Trainer kit or EDA tool)
5. Design and implement 4-bit bidirectional shift register (Trainer kit or EDA tool)
6. Design and implement pulse train generator (Trainer kit or EDA tool)
7. Desing and simulation of 2-input TTL NAND gate using IRSIM.
8. Design and simulation of 2-input CMOS NAND gate using IRSIM.
9. Design N-bit Full Adder using VHDL and implement it on FPGA (Xilinx Vivado)
10. Design N-bit counter using VHDL and implement it on FPGA (Xilinx Vivado)
11. Design and implement Digital-to-Analog converter (Trainer kit or EDA tool)
12. Design and implement Analog-to-Digital converter (Trainer kit or EDA tool)

**List of Project areas:**

1. Mini projects using digital ICs (Software simulation and hardware implementation)
2. Design of moderate to complex digital building blocks such as different types of adders (excluding ripple adder), Adder cum Subtractor, different types of multipliers, ALU, simple controllers using FSMs, Barrel shifter, basic controller/processor architectures using VHDL and its implementation on FPGAs.

**Text Books:**

1. *Thomas L. Floyd, “Digital Fundamentals,” 11th Edition, Pearson Education*
2. *R.P. Jain, “Modern Digital Electronics,” 4th Edition, Tata McGraw Hill*
3. *Douglas L. Perry, “VHDL: Programming by Example,” 4th Edition, McGraw-Hill*

**Reference Books:**

1. *M. Morris Mano, “Digital Design,”4th Edition, Pearson Education*
2. *Donald P. Leach, Albert Paul Malvino and Goutam Saha, “Digital Principles and Applications,” 8th Edition, McGraw Hill*
3. *A. Anand Kumar, “Fundamentals of Digital Circuits,’’ 4th Edition, PHI*
4. *J. Bhasker, “VHDL Primer,” 3rd Edition, Person*

**Course Outcomes:**

The student will be able to

1. Design and implement combinational logic circuits.

2. Design and implement sequential logic circuits.

3. Design finite state machines.

4. Describe digital circuit design using different logic families.

5. Describe various programmable logic devices and its applications using HDL.

6. Compare various ADC and DAC circuits with respect to schematic, working, advantages and disadvantages.

 **FF No. : 654**

# **ET2303: Solid State Devices and Circuits**

**Credits: 3 Teaching Scheme:**

**Theory: 02 Hours / Week**

 **Lab: 02 Hours / Week**

**Section 1:**

UNIT 1: **Semiconductor Fundamentals**:

Energy band and Bond Models in solids: conductors, semiconductors, insulators, Types of semiconductors, Intrinsic and extrinsic semiconductors, PN junction formation, energy band diagrams, Transition and diffusion capacitance. JFET: construction, operation, characteristics, Pinch of voltage, Comparison of FET with BJT.

UNIT 2: **MOSFET:**

Introduction to MOSFET, Types of MOSFET: Enhancement and Depletion type MOSFET, EMOSFET: Threshold voltage, operation regions, V-I characteristics: Transfer characteristics and output characteristics, MOSFET as a switch, non-ideal effects in MOSFET viz. Finite output resistance, channel length modulation, body effect, subthreshold conduction, breakdown effects and temperature effects. Numericals on characteristic equations of MOSFET. Introduction to CMOS technology: nMOS, pMOS, CMOS, HBTs FinFETs, TFETs, etc.

 UNIT 3: **MOSFET DC analysis**:

DC Load Line, operating point (Q point), Types of biasing: Voltage divider bias and drain feedback bias, MOSFET Configurations: Common source circuit, Common Drain circuit and common Gate circuit.

**Section2:**

UNIT 4: **MOSFET AC analysis:**

MOSFET Low frequency and high frequency small signal equivalent circuits. Introduction to CS, CG, and CD amplifiers. AC analysis of a common-source amplifier with and without a bypass capacitor, Numerical analysis based on AC analysis of a common-source amplifier.

UNIT 5: **Feedback Amplifiers**

Concept of Feedback, Feedback topologies: Voltage series, Voltage Shunt, Current series, Current shunt. Effect of feedback on performance parameters of amplifiers. Applications of feedback amplifiers.

UNIT 6: **Oscillators**

Concept of positive feedback, Barkhausen criterion, Types of oscillators, RC phase shift oscillator, Hartley and Colpitts oscillators.

Introduction to Power Amplifiers.

**List of Practical:**

1. Study of VI characteristics of JFET.
2. Study of VI characteristics of MOSFET.
3. Build and test MOSFET as a switch to drive the LED and calculate the Q point.
4. Build a voltage divider biasing circuit for the MOSFET. Calculate the operating point parameter theoretically and practically.
5. Design and build single stage CS amplifier using MOSFET with bypass capacitor. Calculate Av, Ri, Ro theoretically.
6. Simulate an LC oscillator using MOSFET.
7. Design and build single stage Common Source self-biasing circuit for MOSFET. Calculate operating point parameter theoretically and practically.
8. Simulate Voltage-Series feedback amplifier and calculate Rif, Rof, Avf and Bandwidth.
9. Simulate Current-Series feedback amplifier and calculate Rif, Rof, Avf and Bandwidth.
10. Simulate transient, AC, and DC response of MOSFET single stage CD amplifier.
11. Simulate transient, AC, and DC response of MOSFET single stage CG amplifier.
12. Simulate RC oscillator using MOSFET.

**List of Project areas:**

1. JFET based Applications
2. MOSFET based Applications
3. Power amplifier circuits
4. Oscillators

**Text Books: *(As per IEEE format)***

1. Floyd, “Electronic Devices and Circuits”, Pearson Education.
2. Donald Neamen, “Electronic Circuits Analysis and Design”, 3rdEdition, TMH*tion*
3. David A. Bell, “Electronic Devices and Circuits”, 5th Edition, Oxford press.

**Reference Books: *(As per IEEE format)***

1. Millman, Halkias, “Integrated Electronics- Analog and Digital Circuits and Systems”, 2nd TMH.
2. Boylstad, Nashlesky, “Electronic Devices and Circuits Theory”, 9th Edition, PHI, 2006.

**Course Outcomes:**

The student will be able to –

1. Understand the Semiconductor Fundamentals.
2. Understand construction, working, and characteristics of MOSFET.
3. Perform the DC analysis of MOSFET.
4. Perform the AC analysis of MOSFET.
5. Analyse MOSFET based amplifiers.
6. Differentiate between MOSFET based oscillators.

 **FF No. : 654**

# **ET2304: Signals and Systems**

**Credits: 02 Teaching Scheme: 02 Hours / Week**

**Section 1:**

**Unit I: Introduction to Signals and Systems** 6L

Definition of signals and systems, communication and control systems as examples, Classification of signals: Continuous time and discrete time, even, odd, periodic and non periodic, deterministic and non deterministic, energy and power.

Operations on signals: Amplitude scaling, addition, multiplication, differentiation, integration (accumulator for DT), time scaling, time shifting and folding, precedence rule.

Elementary signals: exponential, sine, step, impulse and its properties, ramp, rectangular, triangular, signum, sinc.

Systems: Definition, Classification: linear and nonlinear, time variant and invariant, causal and non-causal, static and dynamic, stable and unstable, invertible.

**Unit II: System Analysis 5L**

System modeling: Input output relation, Definition of impulse response, convolution integral, convolution sum, computation of convolution integral using graphical method for unit step to unit step, unit step to exponential, exponential to exponential and unit step to rectangular, rectangular to rectangular only. Computation of convolution sum. Properties of convolution, system interconnection, system properties in terms of impulse response

**Unit III: System Analysis in Frequency Domain using Fourier Transform 5L**

Definition and necessity of CT Fourier series and Fourier transforms. CT Fourier series, CT Fourier transform and its properties, problem solving using properties, amplitude spectrum, phase spectrum of the signal and system. Interplay between time and frequency domain using sinc and rectangular signals. Limitations of FT and need of LT.

**Unit IV: System Analysis in Frequency Domain using Laplace Transform 5L**

Definition and its properties, ROC and pole zero concept. Application of Laplace transforms to the LTI system analysis. Inversion using duality, numerical based on properties. Signal analysis using LT.

**Unit V: Correlation and Spectral Density 4L**

Definition of Correlation and Spectral Density, correlogram, analogy between correlation, convolution, conceptual basis, auto-correlation, cross correlation, energy/power spectral density, properties of correlation and spectral density, inter relation between correlation and spectral density.

Unit VI: **Unit 6: Sampling Theorem and its Applications 3L**

Sampling process, Nyquist criteria, ADC Blocks, Sampling theorem in time and frequency domain, Aliasing effect, Applications of sampling

**Text Books:**

1. Ramesh Babu, “Signals and Systems”, 4th Ed., Scitech Publications

2. A. Nagoor Kani, “Signals and Systems”, Tata McGraw Hill

**Text Books:**

1. Ramesh Babu, “Signals and Systems”, 4th Ed., Scitech Publications

2. A. Nagoor Kani, “Signals and Systems”, Tata McGraw Hill

**Reference Books:**

1. Simon Haykin, “Signals and Systems”, John Wiley and Sons.

2. B. P. Lathi, “Signal Processing and Linear Systems”.

3. Oppenheim, Willsky and Hamid, “Signals and Systems”, Prentice Hall.

**List of Projects:**

1. ECG Signal Analysis for Arrythmia
2. Musical Instrument Identification.
3. Speech Signal preprocessing and Analysis
4. Analyze the overlapping and Aliasing effects in a Discrete time Signal
5. Extraction of Time and frequency domain features of a discrete time signal
6. Playing of Melodious music
7. Analysis of various Biomedical signals like ECG, EEG, EMG
8. Brain-Controlled Robotics via EEG
9. Gravitational Wave Detection
10. DNA Sequence Compression
11. Neuromorphic Signal Processing
12. Automotive & Transportation
13. Earthquake Early Warning System
14. Flood Prediction via River Sound Analysis
15. Steganography Detection in Images
16. Steganography Detection in Images
17. Motor Fault Detection via Vibration Analysis
18. Lung Sound Classification (COVID-19 Screening)

FF No.: 654

# **ET2292: Engineering Design and Innovations-III**

**Course Prerequisites:**

Basic Electronics, Physics, Engineering Mathematics, Statistics, Programming Languages

**Course Objectives**:

1. To develop critical thinking and problem-solving ability by exploring and proposing solutions to realistic/social problems.
2. To Evaluate alternative approaches, and justify the use of selected tools and methods,
3. To emphasize learning activities those are long-term, inter-disciplinary and student centric.
4. To engage students in rich and authentic learning experiences.
5. To provide every student the opportunity to get involved either individually or as a group so as to develop team skills and learn professionalism.

**Credits: 6 Teaching Scheme:** Lab 2 Hours/Week

Course Relevance:

Project Centric Learning (PCL) is a powerful tool for students to work in areas of their choice and strengths. Students can solve socially relevant problems using various technologies from relevant disciplines. The various socially relevant domains can be like Health care, Agriculture, Defense, Education, Smart City, Smart Energy and Swaccha Bharat Abhiyan. Students can be evaluated for higher order skills of Blooms taxonomy like ‘analyze, design and apply’. This course is capable of imparting hands on experience and self-learning to the students which will help them throughout their career. This is a step ahead in line with national policy of Atmanirbhar Bharat.

|  |
| --- |
| **Preamble** - **The content and process mentioned below is the guideline document for the faculties and students to start with**. It is not to limit the flexibility of faculty and students; rather they are free to explore their creativity beyond the guideline mentioned herewith. This course is designed to encourage and ensure application of technology for solving real world problems using an interdisciplinary approach.Students need to plan their work in following steps:1. Formation of project group comprising of 4-5 students. Multidisciplinary groups are allowed
2. A supervisor/mentor teacher assigned to individual groups.
3. Carrying out literature survey
4. Finalization of problem statement
5. Planning the project execution
6. Execution of project and testing
7. Writing a report
8. Publication in the form of research paper/patent/copyright as found suitable by supervisor/mentor

**Teacher’s Role in PCL:*** 1. Teacher is not the source of solutions rather he will they act as the facilitator and mentor.
	2. To utilize the principles of problems solving, critical thinking and metacognitive skills of the students.
	3. To aware the group about time management.
	4. Commitment to devote the time to solve student’s technical problems and interested in helping students to empower them better.

**Student's Role in PCL:*** + 1. Students must have ability to initiate the task/idea they should not be mere imitators.
		2. They must learn to think.
		3. Students working in PCL must be responsible for their own learning.
		4. Students must quickly learn how to manage their own learning, Instead of passively receiving instruction.
		5. Students in PCL are actively constructing their knowledge and understanding of the situation in groups.
		6. Students in PCL are expected to work in groups.
		7. They must develop interpersonal and group process skills, such as effective listening or coping creatively with conflicts.
 |

|  |
| --- |
| **Core Technology domains identified for E&TC Engg are as below. However, this list can be extended as per the need of project and multidisciplinary approach**1. VLSI Design
2. Embedded Systems
3. Signal Processing
4. Communication
5. Machine learning
 |
| **Assessment Scheme:**Mid Semester Examination - 30 Marks End Semester Examination - 70 Marks |
| **MOOCs Links and additional reading material:**[www.nptelvideos.in](http://www.nptelvideos.in/) https://worldwide.espacenet.com/ |
| **Course Outcomes:**1. Review the literature to formulate problem statement to solve real world problems.
2. Apply knowledge of technology and modern tools to design solution considering sustainability and environmental issues.
3. Manage project ethically as team member/ lead.
4. Demonstrate effectively technical report/ research paper/ prototype/patent.
 |
| **CO PO Map** |
|  | **CO** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** |
|  | 1 | 3 | 3 | 2 | 3 | 1 | 0 | 0 | 3 | 3 | 3 | 1 | 2 | 2 | 2 |
|  | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 1 | 2 | 3 | 3 |
|  | 3 | 1 | 1 | 1 | 1 | 1 | 2 | 0 | 3 | 3 | 3 | 3 | 2 | 1 | 1 |
|  | 4 | 3 | 1 | 1 | 1 | 0 | 0 | 0 | 3 | 1 | 3 | 3 | 2 | 1 | 2 |
| 1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)**CO attainment levels:** CO1: - Level 3 CO2: - Level 4 CO3: - Level 3 CO4: - Level 4 |

FF 654

# **ET2245: Design Thinking-3**

Credits: 1 Teaching Scheme Tut: 1

Hour/Week

|  |
| --- |
| **Course Objectives:**To provide ecosystem for students and faculty for paper publication and patent filing |
| **Contents:**Structure of The paper Journal List (Top 50 Journals) Selection of the journalUse of various online journal selection tools Plagiarism checkingImproving contents of the paper Patent draftingPatent search Filing of patentWriting answers to reviewer questions Modification in manuscriptChecking of publication draft |
| **Suggest an assessment Scheme:**Publication of paper or patent |
| **Course Outcomes:**On completion of the course, learner will be able to– CO1: Understand the importance of doing ResearchCO2: Interpret and distinguish different fundamental terms related to ResearchCO3: Apply the methodology of doing research and mode of its publication CO4: Write a Research Paper based on project workCO5: Understand Intellectual property rights CO6: Use the concepts of Ethics in ResearchCO7: Understand the Entrepreneurship and Business Planning**CO- PO Mapping** |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | PSO1 | PSO2 |
|  | CO1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 2 |
|  | CO2 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
|  | CO3 | 2 | 2 | 3 | 3 | 2 | 2 | 1 | 2 | 2 | 3 | 0 | 1 | 2 | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CO4 | 3 | 3 | 3 | 3 | 3 | 2 | 1 | 2 | 2 | 3 | 1 | 1 | 0 | 0 |  |
|  | CO5 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |  |
|  | CO6 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 3 | 2 | 3 | 3 | 1 | 0 | 1 |  |
|  | CO7 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 |  |
| CO Attainment levels |  |
| CO1 | CO2 | CO3 | CO4 | CO5 | CO6 | CO7 |  |
| 2 | 2 | 3 | 5 | 2 | 3 | 2 |
|  |

FF 654

# **SH2001: Reasoning And Aptitude Development**

**Unit 1: English Language**

Familiarity with English Language, Ability to understand written text, spoken word and effective communication through written documents; Coverage of vocabulary to cope up with general and specific terminology, syntax and sentence structure, prevention of incorrect use leading to distortion in communication; synonyms, antonyms and contextual vocabulary, Grammar – Error identification, sentence improvement and construction, Reading Comprehension

**Unit 2: Logical Ability**

Objective interpretation of things, ability to perceive and interpret trends to make generalizations; ability to analyze assumptions behind an argument or statement; Deductive reasoning: Assessment of ability to synthesize information and derive conclusions - Coding deduction logic, Data Sufficiency, Directional Sense, Logical word sequence, Objective reasoning, Selection and decision tables, puzzles; Inductive reasoning: Assessment of ability to learn by example, imitation or by trial – Analogy pattern recognition, Classification pattern recognition, Coding pattern recognition, Number series pattern recognition; Abductive reasoning: Critical thinking ability of seeing through logical weak links or loopholes in an argument or a group of statements; Critical reasoning: assessment of ability to think through and analyze logical arguments, assessment of ability to use logical constructs to offer reasoning in unfamiliar situations; Information Gathering and synthesis: Ability of locating information, information ordering, rule based selection and data interpretation, order and classify data, interpret graphs, charts, tables and make rule based deductions. Application of these approaches for using visual, numerical and textual data from single or multiple sources

**Unit 3: Quantitative Ability**

Basic numbers – decimals and fractions, factorization, divisibility: HCF, LCM, Odd, even, prime and rational numbers. Application of algebra to real world, direct and inverse proportion, common applications – Speed-time -distance, Profit-loss, percentage, age relations, mixtures, other miscellaneous quantitative combination, exponentials and logarithms, permutations and combinations, probability. Spatial reasoning: Inductive – Missing portions, Sequence and series; Deductive analysis.

**Reference Books –**

1: "English Grammar in Use" by Raymond Murphy, Cambridge University Press. 2: "Word Power Made Easy" by Norman Lewis, Goyal Publishers & Distributors. 3: "Objective General English" by S.P. Bakshi, Arihant Publications.

4: "English for Competitive Examinations" by K. Sinha, S. Chand Publishing. 5: "Essential English Grammar" by Philip Gucker, Wiley.

6: "English Idioms and Phrasal Verbs" by M.A. Yadav, Vikas Publishing House.

7: "The Oxford English Grammar" by Sidney Greenbaum, Oxford University Press.

8: "A Modern Approach to Verbal & Non-Verbal Reasoning" by R.S. Aggarwal, S. Chand Publishing, ISBN: 978-8121903409.

9: "Logical Reasoning and Data Interpretation for the CAT" by Nishit K. Sinha, Pearson India, ISBN: 978-8131709117.

10: "Logical Reasoning and Data Interpretation for the CAT" by Arun Sharma, McGraw Hill Education, ISBN: 978-0070709642.

11: "A New Approach to Reasoning Verbal and Non-Verbal" by B.S. Sijwali & Indu Sijwali, Arihant Publications, ISBN: 978-9311124692.

12: "Quantitative Aptitude for Competitive Examinations" by R.S. Aggarwal, S. Chand Publishing, ISBN: 978-8121900637.

13: "How to Prepare for Quantitative Aptitude for the CAT" by Arun Sharma, McGraw Hill Education, ISBN: 978-0070709642.

14: "The Pearson Guide to Quantitative Aptitude for Competitive Examination" by Pearson, Pearson India, ISBN: 978-8131709117.

15: "Quantitative Aptitude for Competitive Examinations" by Abhijit Guha, Tata McGraw Hill Education, ISBN: 978-0070666653.

16: "Data Interpretation & Data Sufficiency" by R.S. Aggarwal, S. Chand Publishing ISBN: 978- 8121903515.

17: "Quantitative Aptitude for Competitive Examinations" by S. Chand, S. Chand Publishing, ISBN: 978-8121903423.

**Course Outcomes –**

Upon completion of the course, the student will be able to –

1. Improve the reading, writing and verbal skills, and enhance comprehension and articulation abilities
2. Develop logical reasoning abilities, enabling them to make sound decisions in problem-solving scenarios
3. Develop mathematical aptitude as well as data interpretation abilities and use them in test cases and real world problems
4. Learn to apply approaches for optimum time-management, prioritization maximizing the accuracy
5. Learn data interpretation, apply mathematical skills to draw accurate conclusions
6. Apply their knowledge of English, reasoning and quantitative skills for planning, critical thinking and real world problems

FF 654

# **ETM001: IOT for Smart Applications**

Total Credits: 3 Teaching Scheme:

Theory: 2 Hours / Week ;

Tutorial: 1 Hours / Week

**SECTION I:**

**UNIT I: Introduction to IoT ( 4)**

Introduction, Definitions Characteristics of IoT, IoT Architectures, Physical and Logical Design of IoT, Enabling Technologies in IoT, IoT frameworks, Applications of IOT in various domains.

**UNIT II:** **IOT Design Methodology and Platform.** (4)

IoT Design Methodology Steps and IoT System Design Cycle.

Hardware Platforms - Arduino, Raspberry Pi, NodeMCU, ESP32. Sensors, actuators selection criteria for specific applications and interfacing basics.

**UNIT III:** **IOT Protocols and Standards (6)**

RFID, IEEE 802.15.4, Zigbee, NFC, Z-Wave, BLE, Bacnet, Modbus. IPv6, 6LowPAN, MQTT. Authorization and Access Control in IOT.

**SECTION II:**

**UNIT IV: Wireless Sensor Networks for IOT (4)**

Types of Wireless Sensors, Examples and Working, Wireless Sensor Networks: History and Context, of the node, Connecting nodes, Networking Nodes, WSN and IoT. Types of Network and network topologies for IOT.

**UNIT V: Cloud Integration and Data Visualization (5)**

Cloud platforms for IoT: Thing Speak, Blynk, Firebase. Real-time data logging, charting, and alerts. Security and privacy fundamentals: Security Challenges in IoT Systems, Authentication, Authorization, and Access Control.

**UNIT VI: Domain-Specific Smart Systems: Case Studies (5)**

Smart agriculture, Smart health, Smart cities, Smart transportation, Industrial IoT. System integration and real-world challenges.

**List of Tutorials**

1. Smart Agriculture System
2. Weather Reporting System
3. Home Automation System
4. Air Pollution Monitoring System
5. Smart Parking System
6. Smart Traffic Management System
7. Smart Cradle System
8. Smart Gas Leakage Detector Bot
9. Streetlight Monitoring System
10. Smart Anti-Theft System
11. Liquid Level Monitoring System
12. Night Patrol Robot
13. Health Monitoring System
14. Smart Irrigation System
15. Flood Detection System
16. Mining Worker Safety Helmet
17. Smart Energy Grid

**Suggested Sensors for project to respective domain:**

**Domain Suggested Sensors**

Civil Ultrasonic, MQ135, Soil Moisture, Vibration

Mechanical Load Cell, Thermocouple, Accelerometer

Electrical Voltage/Current Sensors, LDR, Relay

Biomedical Pulse Sensor, MLX90614, MAX30100

Agriculture Soil Moisture, DHT11, UV Sensor, pH Sensor

Transport GPS, IR, Hall Sensor, Accelerometer

**List of Project areas:**

1. Smart Home & Building Automation.
2. Smart Agriculture.
3. Smart Healthcare & Biomedical
4. Smart Transportation & Mobility
5. Industrial IoT & Predictive Maintenance
6. Civil and Infrastructure Monitoring
7. Electrical & Energy Systems
8. Environmental Monitoring

**Course Outcomes**

The student will be able to –

1. CO1: Demonstrate fundamental concepts of Internet of Things (L3 – Apply)
2. CO2: Recognize IoT Design Methodology Steps (L2-Understand)
3. CO3: Apply basic Protocols in IoT (L3 – Apply)
4. CO4: Analyze fundamentals of networking (L4-Analyze)
5. CO5: Apply of cloud platforms to visualize real-time sensor data (L3 – Apply)
6. CO6: Provide IoT solutions practically with the help of case study (L6 – Create)

**Books and E-Resources**

 **For Reference Print Book -**

1. R. Kamal, Internet of Things: Architecture and Design Principles, 1st ed., McGraw Hill Education, 2021.
2. Olivier Hersent, D. Boswarthick, and O. Elloumi, The Internet of Things: Key Applications and Protocols, 2nd ed., Wiley, 2021.
3. Pethuru Raj and Anupama C. Raman, “The Internet of Things: Enabling Technologies, Platforms, and Use Cases”, CRC Press

**Text Books:**

1. Arsheep Bahga and Vijay Madisetti, Internet of Things: A Hands-On Approach, 1st ed., Universities Press, 2014.
2. Adrian Mcewen and Hakim Cassimally, Designing the Internet of Things, 1st ed., Wiley, 2013.
3. Daniel Lion, Introduction to Internet of Things (IoT), 1st ed., Independently Published, 2023.
4. S. Verma, R. Verma, O. Farhaoui, and J. Lyu, Eds., Emerging Real-World Applications of Internet of Things, 1st ed., CRC Press, 2024.

**For MOOCs and other learning Resources**

1. <https://www.coursera.org/specializations/iot>, **An Introduction to Programming the Internet of Things (IOT) Specialization.** Create Your Own Internet of Things (IoT) Device. Design and create a simple IoT device in just six courses. Instructor: [Ian Harris](https://www.coursera.org/instructor/ianharris)
2. <https://www.coursera.org/learn/raspberry-pi-interface> Interfacing with the Raspberry Pi, This course is part of An Introduction to Programming the Internet of Things (IOT) Specialization. Instructor: [Ian Harris](https://www.coursera.org/instructor/ianharris)

FF 654

# **ETM002: Microcontroller and Applications**

Total Credits: 3 Teaching Scheme:

 Theory: 2 Hours / Week

 Tutorial: 1 Hours / Week

**SECTION I:**

**UNIT 1: Introduction to Microcontrollers and Architecture (5)**

Microprocessor & Microcontroller comparison, Harvard & Von Neumann architecture, RISC & CISC comparison, Evolution of microcontrollers, Microcontroller selection criteria for particular application, MCS-51 architecture, family devices & its derivatives. Pin configuration,

**UNIT 2: Internal modules of 8051 microcontroller (4)**

Internal Port architecture, memory organization, external memory interfacing. Timers and its modes. Interrupt structure, Serial communication and its modes.

**UNIT 3: 8051 Instruction Set, Programming and development tools (5)**

Addressing modes, 8051 Instruction set, Programming environment: Study of software development tool chain (IDE), debugging tools, Programs: Assembly language programs.

**SECTION II:**

**UNIT 4: 8051 Microcontroller based Real World Interfacing and programming I (5)**

Interfacing peripheral devices using GPIO: LEDs 7 segment LED, generating various delays using timer, counter, switches, relay, stepper motor, LCD interfacing, keyboard interfacing, (Programming in C).

**UNIT 5: 8051 Microcontroller based real world interfacing and programming II (4)**

Basics of serial communication protocol: Synchronous and asynchronous communication, RS232, RS485, SPI, I2C. Interfacing of devices using protocols: Interfacing of peripherals using UART, interfacing RTC DS1307 using I2C protocol, Programs in C.

**UNIT 6: AVR RISC Microcontroller and programming (5)**

Overview of AVR family, AVR Microcontroller architecture, ROM space and other hardware modules, **interfacing peripheral devices with AVR:** DC motor control using PWM programming, ADC and temperature sensor LM35 interfacing, **Application areas:** home automation, smart health, smart agriculture. Design of simple real-life applications using microcontroller platforms

**List of Tutorials :**

1. Setting up the Microcontroller Development Environment (IDE and Toolchains)
2. Simple programs to explore 8051 IDE (Addition, subtraction, multiplication etc)
3. Writing and uploading program (Blinking an LED)
4. Interfacing of LED’s, switches, buzzer, relay with 8051 Microcontroller.
5. Interfacing of 16x2 LCD in 8 bit/4-bit mode with 8051 Microcontroller and display message on it.
6. Interface 4x4 matrix keyboard with 8051 Microcontroller. Display value of pressed switch on LCD.
7. Interface Computer with 8051 Microcontroller using UART communication.
8. Interface stepper Motor with 8051Microcontroller and write program to rotate it in clockwise and anticlockwise direction using different drives (Full step drive, Half step drive and wave drive).
9. Interfacing of ADC PCF8591 with 8051 Microcontroller using IIC protocol read the analog voltage from ADC and display its equivalent digital value on LCD.
10. Writing and uploading AVR Program (LED Blink)
11. AVR based Temperature indicator using sensor LM35
12. Servo Motor interfacing with AVR ATMega32 Microcontroller
13. DC Motor interfacing with AVR ATMega32 Microcontroller

**Text Books: *(As per IEEE format)***

1. *Mazidi Muhammad Ali; Mazidi Janice Gillispie; McKinlay Rolin D, “The 8051 Microcontroller and Embedded Systems Using Assembly and C”, 2nd Edition, Dorling Kindersley.*
2. *The AVR Microcontroller and Embedded Systems Using Assembly and C, By Muhammad Ali Mazidi, Sarmad Naimi and Sepehr Naimi, Pearson Education.*

**Reference Books: *(As per IEEE format)***

1. *Dhananjay Gadre, Programming and Customizing the AVR Microcontroller, McGraw Hill edu.*
2. *Richard Barnett, Sarah Cox, Larry O'Cull, “Embedded C Programming and the AVR Microcontrollers”, 2nd edition Thomson publication.*
3. *Ayala Kenneth J, Gadre Dhananjay V, “8051 Microcontroller and Embedded Systems”, Cengage Learning.*

**Datasheets (from websites):**

1. *ATMEL 8051/52 data sheet 2. AVR ATmega32 data sheet*

**Online Resources:**

1. [*https://archive.nptel.ac.in/courses/108/105/108105102/*](https://archive.nptel.ac.in/courses/108/105/108105102/) *(From Lecture 23 onwards)*
2. [*https://nptel.ac.in/courses/117104072*](https://nptel.ac.in/courses/117104072)
3. [*https://www.coursera.org/learn/microcontroller-and-industrial-applications*](https://www.coursera.org/learn/microcontroller-and-industrial-applications)

**Course Outcomes:**

The student will be able to –

1. Explain the architecture of 8051 CISC and RISC microcontroller. (Level 2 Understand)
2. Understand the internal peripheral modules in 8051 microcontroller. (Level 2 Understand)
3. Write and debug assembly language programs using appropriate instruction sets, addressing modes, and debugging tools. (Level 3 Apply)
4. Interface peripheral devices with the 8051 microcontroller. (Level 3 Apply)
5. Demonstrate bus standards used in industrial environment. (Level 3 Apply)
6. Develop system using different microcontroller based for embedded applications. (Level 6 Create)

 **FF No. : 654**

# **ETH001:Computer Organization and Architecture**

**Credits: 3** **Teaching Scheme Theory**: 2 Hours/Week

 **Lab:** 2 Hours/Week

**Section 1**

**Unit 1: Computer Evolution & Arithmetic**

Organization & Architecture, Structure & Function, Brief History of Computers, Integer Representation: Fixed point & Signed numbers. Integer Arithmetic: 2‘s Complement arithmetic, multiplication, Booth‘s Algorithm, Division Restoring Algorithm, Non-Restoring algorithm.

**Unit 2: Processor Design**

CPU Architecture, Register Organization, Instruction types, Types of operands, Types of operation, Instruction formats, Addressing modes and address translation. Instruction cycles. RISC Processors: RISC- Features, CISC Features, Comparison of RISC & CISC Processors. Case study of Processor: Von Neumann Architecture.

**Unit 3: Control Unit**

Fundamental Concepts: Single Bus CPU organization, register transfers, performing an arithmetic/ logic operation, fetching a word from memory, storing a word in memory, Execution of a complete instruction. Micro-operations, Types of Control Unit, Hardwired Control, Micro-programmed Control: Microinstructions.

**Section 2**

**Unit 4: Memory Organization**

Need, Hierarchical memory system, Characteristics, Size, Access time, Read Cycle time and address space. Main Memory Organization: ROM, RAM, EPROM, E2PROM, DRAM, Cache memory Organization, Cache Mapping techniques: Direct, Set Associative, Fully Associative.

**Unit 5: Pipelining (5 hours)**

Data hazards: operand forwarding, handling data hazards in software, side effects. Instruction hazards: unconditional branches, conditional branches, and branch prediction. Performance considerations: effect of instruction hazards, number of pipeline stages.

**Unit 6: I/O Organization**

Input/output systems, I/O Transfer Techniques: Program-controlled, Interrupt-Driven, DMA controlled synchronous, asynchronous, working mechanisms of peripherals: keyboard, video displays, touch screen panel, printers

**List of Practicals**

* + - 1. Study of 8086 Architecture and Execution of sample programs.
1. Write 8086 ALP to access marks of 5 subjects stored in array and find overall percentage and display grade according to it.
2. Write 8086 ALP to pe Data bytes in a block stored in one array transfer to another array. Use debugger to show execution of program.
3. Write 8086 ALP to find and count zeros, positive number and negative number from the array of signed number stored in memory and display magnitude of negative numbers.
4. Write 8086 ALP to convert 4-digit HEX number into equivalent 5-digit BCD number.
5. Write 8086 ALP to convert 5-digit BCD number into equivalent 4-digit HEX number.
6. Write 8086 ALP for following operations on the string entered by the user. a. String length b. Reverse of the String c. Palindrome
7. Write 8086 ALP for following operations on the string entered by the user (Use Extern Far Procedure).
8. Concatenation of two strings b. Find number of words, lines.
9. Find number of occurrences of substring in the given string.
10. Write 8086 ALP to initialize in graphics mode and display following object on screen.
11. Write 8086 ALP to encrypt and decrypt the given message.
12. Write 8086 ALP to perform following operations on file a. Open File b. Write data in the file. c. Delete data in the file. d. Close the file.

**List of Course Projects:**

* + - 1. Combinational and Sequential circuits
			2. Memory Management
			3. Graphics Mode
			4. IOT based projects.
			5. IoT based atmospheric CO2 administration.
			6. IoT based flood risk predictor.
			7. Simulate modern traffic control system.
			8. Online Parallel Examination.

**Text Books:**

1. "Computer Organization" – *Carl Hamacher, Zvonko Vranesic, Safwat Zaky*, Publisher: McGraw Hill
2. “Computer Organization and Architecture: Designing for Performance “by William Stalling, **Publisher** Pearson Education / Prentice Hall
3. **Microprocessor Architecture, Programming and Applications with the 8085/8086"** – Ramesh Gaonkar

**Reference books**

1. The 8086 Microprocessor: Programming & Interfacing the PC" – Kenneth J. Ayala
2. Computer Architecture: A Quantitative Approach***"*** – John L. Hennessy, David A. Patterson, Publisher: Morgan Kaufmann

**Course outcome**

1. Explain computer evolution and perform integer arithmetic operations.
2. Analyze CPU design and compare RISC and CISC architectures.
3. Describe control unit operations and distinguish control types.
4. Explain memory hierarchy and compare memory and cache types.
5. Evaluate pipelining and apply hazard-handling techniques.
6. Explain I/O systems and demonstrate I/O transfer method

# **ETH002: Microcontroller Applications**

**Credits: 3** Teaching Scheme: 2 Hours / Week

 Lab: 2 Hours / Week

**Section I**

**Unit 1: Microcontroller Fundamentals**

Microcontroller Vs Microprocessor, Brief History of 8-bit Microcontrollers and Family Overview, Features of 8051, RISC Vs CISC and Harvard Architecture Vs Von Neumann Architecture, Architecture of 8051

**Unit 2: Assembly Language Programming**

Instruction Set, Addressing Modes , Arithmetic and Logical Instructions, Branch Instructions and Looping, I/O Port Programming and Bit Manipulation, Call Instruction and Stack, Time Delay and Instruction Pipeline, introduction to embedded C Programming

**Unit 3: GPIO Programming**

GPIO overview and functionality, Read/write operations on digital pins, Practical examples: LED blinking, Switch-controlled LED blinking, LCD Interfacing,

**Section-II**

**Unit 4: On-Chip Peripheral Programming**

 On-chip Timers & Their programming in C, Interrupts  & Their programming in C, Types of ADC and On-chip ADC Programming, Types of DAC and Interfacing of DAC 0808 to 8051

**Unit 5: Communication Protocol**

RS 232, UART, SPI Protocol, I2C protocol programming

**Unit 6: Sensors and Actuators Interfacing**

Sensor Interfacing and Signal Conditioning, Relay and Optoisolator Interfacing, Stepper Motor Interfacing, PWM Programming and DC Motor Control

**List of Practicals:**

1. Write an ALP code to flash LEDs connected to PORT

2. Write an ALP to display 0 to 9 numbers on 7-Segment Display using Timer

3. Write a C program to display message on 16 X 2 LCD

4. Write a C program to interface 4 X 4 matrix keypad

5. Write a C program to use on chip ADC

6. Write a C program to read/write on chip EEPROM.

7. Write a C program to control DC motor using PWM

8. Write a C program for serial communication using on chip UART

9. Write a C program to implement I2C protocol for RTC

10.Write a C program to read switch and control AC Lamp load using relay.

**List of Project areas:**

1. Industrial Automation

2. Home automation

3. Robotics

4. Medical Electronics

5. Communication

6. Automotive Electronics

Examples: - Design of Electronic Lock, Design of Temperature Controller

**Text Books:-**

1. Mohammad AliMazidi, Sarmad Naimi&SepehrNaimi; The 8051 Microcontroller and Embedded Systems using Assembly and C; 1stEdition; Pearson Education India.

2. Dhananjay Gadre; Programming and Customizing the AVR Microcontroller; 1 st Edition; McGraw Hill.

**Reference Books:**

8051 8-Bit Microcontrolle Datasheet complete

**Course Outcomes:**

The student will be able to –

Compare microcontrollers and microprocessors

Explain 8051 Microcontroller Architecture.

2. Develop an Assembly Language Program & Embedded C Program for 8051 Microcontroller.

3. Make use of internal peripherals of 8051Microcontroller.

4. Develop an interface for Sensors and Actuators

 **FF No. : 654**

# **ETH003:Analog Mixed Signal Design**

**Credits: 3 Teaching Scheme:**

**Theory: 2 Hours / Week**

**Laboratory: 2 Hours/ Week**

**Section 1:**

**MOSFET Amplifiers** - MOSFET Small signal Model, Basic MOS Transistor Amplifiers, two stage amplifiers, Cascode – Folded and Telescopic, DC and AC behaviour

**CMOS Analog Circuits** - Current and Voltage references, Current mirrors, Bandgap references, Sampling circuits, switched capacitor filter

**Data Converters and Layout** - Performance Metrics of Data Converters, Sigma-Delta and Pipelined Architectures, Layout of Analog Circuits, Custom Analog Circuit Design Flow on the EDA tools

**Section2:**

**Introduction to RF -** Introduction to RF and Wireless Technology, Basic Concepts in RF Circuits, MOSFET - RF perspective, Transmission media and Reflections

**Passive Components** - Resistor, Inductor, Capacitor, Series and Parallel RLC networks, Q Factors, S parameters

**Noise and RF Circuits -** Noise, Low Noise Amplifier (LNA), Voltage Controlled Oscillator (VCO), Phase-Locked Loop (PLL)

**List of Practicals:**

1. Extraction of parameters like threshold voltage, transconductance from I-V characteristics of MOSFET
2. Design and Simulation of basic current mirror
3. Simulation of an Amplifier to find frequency response, gain, bandwidth, and phase margin
4. Design and simulation of S/H circuit to analyze droop and aperture error.
5. Design and simulation of ADC to find INL, DNL.
6. Simulation of basic loop filter in PLL
7. Simulation of a bandgap reference circuit.
8. Simulation of Cascode current mirror for higher output resistance.
9. Layout of a simple analog circuit.
10. Parasitic extraction and post-layout simulation.

**List of Project areas:**

1. Design and simulation of Flash, SAR, Pipeline, or Sigma-Delta ADCs with performance metrics like SNDR, ENOB, INL/DNL
2. Comparator Design - Low-offset, low-power comparators for ADCs or mixed-signal circuits
3. Sample and Hold Circuit Design
4. Phase-Locked Loop (PLL) - Design including PFD, CP, VCO, and loop filter for clock generation.
5. Implementing Clock Data Recovery (CDR) circuits for high-speed serial interfaces.
6. Sigma-Delta Modulator - Design of first-order or higher-order modulators for high-resolution ADCs.
7. Pipelined Data Converters Design
8. Switched-Capacitor (SC) Circuits - Design of SC integrators, filters, or ADC front-ends.

**Text Books:**

1. Behzad Razavi, “Design of Analog CMOS Integrated Circuits”, 2nd Edition, McGraw Hill.

2. Phillip E. Allen and Douglas R. Holberg, “CMOS Analog Circuit Design”, Third Edition, Oxford University Press.

**Reference Books:**

1. [Franco Maloberti](https://link.springer.com/book/10.1007/978-0-387-32486-9#author-0-0), “Data Converters”, Springer, 2002

2. [Paul R. Gray,](https://www.wiley.com/en-hk/search?filters%5bauthor%5d=Paul%20R.%20Gray&pq=++) [Paul J. Hurst,](https://www.wiley.com/en-hk/search?filters%5bauthor%5d=Paul%20J.%20Hurst&pq=++) [Stephen H. Lewis,](https://www.wiley.com/en-hk/search?filters%5bauthor%5d=Stephen%20H.%20Lewis&pq=++) [Robert G. Meyer](https://www.wiley.com/en-hk/search?filters%5bauthor%5d=Robert%20G.%20Meyer&pq=++), “Analysis and Design of Analog Integrated Circuits, 6th Edition, Wiley, 2017.

3. [Behzad Razavi](https://www.amazon.in/Behzad-Razavi/e/B000APU0HI/ref%3Ddp_byline_cont_book_1), “RF Microelectronics”, Third Edition, Prentice Hall.

**Course Outcomes:**

The student will be able to –

1. Draw the Schematic of the circuit given
2. Find out performance parameters of Amplifier like gain, bandwidth
3. Realize a specific value resistance using switch capacitors
4. Calculate S-parameters of given amplifier
5. Carry out the layout of a design
6. Extract the parasitics in the design

# **ETH004:Digital Integrated Circuit Design**

**Credits: 3 Teaching Scheme:**

**Theory: 2 Hours / Week**

**Laboratory: 2 Hours/ Week**

**Section 1:**

[**Overview of VLSI Design Flow**](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/overview-of-vlsi-design-flow/7A748E95133A2F6D6F1C66EE8C8E845B) - [Introduction to Integrated Circuits](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/introduction-to-integrated-circuits/EAA7D27B0582E5C522C89EE2110D420E), [RTL to GDS Implementation Flow](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/rtl-to-gds-implementation-flow/C90155374C511C59A466C71F731D1F46), [Verification Techniques](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/verification-techniques/6C1159E6A9B9B76091E36F80E74E7DAA), [Testing Techniques](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/testing-techniques/BEB255D403BBF28FABC76E273FBA19E3), [Post-GDS Processes](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/post-gds-processes/771EC491AB5A05D538FF2E8110A03F6D)

[**Logic Design**](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/logic-design/7F3389A728291ABCF340585938763882) - [Modeling Hardware Using](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/modeling-hardware-using-verilog/DF9FB3E42AE147F7D2F08E9CF0275590) HDL, [Simulation-based Verification](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/simulation-based-verification/958D6EE4B65030765BD184077A06EB42), [RTL Synthesis](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/rtl-synthesis/F73F27DD0E2E70AB07CD6345D2D4D8F1), [Formal Verification](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/formal-verification/A1D5B47AD0ED119E6890E26F5A62FB96), [Logic Optimization](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/logic-optimization/3E85FBBA474BDD5B61932F08F0AAB89C)

**Design Optimization -** [Technology Library](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/technology-library/A9F4D2B5A1AC4071568E963EEA921FEE), [Static Timing Analysis](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/static-timing-analysis/8D015682E80B10D052A7AF16F083C46D), [Constraints](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/constraints/F199B2062EE193E84AE602B6E6912B06), [Technology Mapping](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/technology-mapping/624F8C739EF4EE821F1C66EEE4390CF8), [Timing-driven Optimizations](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/timing-driven-optimizations/5DA9925766079C86A02F774364893FC1), [Power Analysis](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/power-analysis/8C4C2CD01B7C4AEE0EBCF44CFEEBD2B5), [Power-driven Optimizations](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/power-driven-optimizations/920E0EFA80D391107FCFCF58C37A0E64)

**Section2:**

[**Design for Testability (DFT)**](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/design-for-testability-dft/99A4A4427E8C6F5968ECC44CEE21FE5B) - [Basics of DFT](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/basics-of-dft/76A38AE741164C0BFFCBB096AC1DE5C0), [Scan Design](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/scan-design/4FD9DED0863D2AC2994AE1D991107AD4), [Built-in Self-test](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/built-in-self-test/7D8FCEF80AF785ED2E52634F2965FD9D)

[**Physical Design**](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/physical-design/3844AA5D2F4EC5EBEC11D95CAD25D54C) - [Basic Concepts for Physical Design](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/basic-concepts-for-physical-design/15519C4AE6F049D602BF153190F4CAC7), [Chip Planning](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/chip-planning/4631B2320EDF38247D7FE3EFD83CAF57), [Placement](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/placement/2B4F3B40D541471ADAC069BBD38837CE), [Clock Tree Synthesis](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/clock-tree-synthesis/38434C8375376E5A1F170926F2CD4E33)

**Physical Verification -** [Routing](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/routing/F5113FF035E6B08655B363CFFAE0EE5F), [Physical Verification and Signoff](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/physical-verification-and-signoff/F43E4CE9B0D8D2EB6ADF9CB32DFA781E), [Post-silicon Validation](https://www.cambridge.org/highereducation/books/introduction-to-vlsi-design-flow/93E6832E63FE6B795181D6D67B552333/post-silicon-validation/567B33F310111B6940613D7CEA1F88F2)

**List of Practicals:**

Practicals using simple digital design (counter, shifter)

1. RTL coding and functional verification
2. Perform logic synthesis
3. Generation of netlist and constraint file
4. Analyse synthesis report – area, power, timing
5. Running equivalency checks at different stages of the flow
6. Perform placement of standard cells
7. Carry out place-and-route on the synthesized netlist
8. Generate clock tree for synthesized netlist
9. Writing out a GDSII file
10. Design for test

**List of Project areas:**

1. RISC-V Core Design
2. Floorplanning & Placement of a RISC-V Core
3. AXI4-Lite Bus Protocol Implementation
4. Clock Tree Synthesis and Analysis
5. FIFO/ALU/Multiplier RTL
6. Power Grid Design for a Macroblock
7. Design of Low-Power FIR Filter
8. IR Drop Analysis for a Block-Level Design
9. DFT Insertion and ATPG for a Core
10. DRC/LVS Checks on a Custom Layout
11. FPGA Prototyping of an SoC
12. Routing Optimization for Congested Block

**Text Books:**

1. Brock J. LaMeres, “Introduction to Logic Circuits & Logic Design with Verilog”, Springer, 2024.

2. Sneh Saurabh, “Introduction to VLSI Design Flow”, Cambridge University Press, 2023.

**Reference Books:**

1. Lavagno, I. L. Markov, G. Martin, and L. K. Scheffer (Editors), “Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology”, CRC Press, 2016

2. [Himanshu Bhatnagar](https://link.springer.com/book/10.1007/b117024#author-0-0), Advanced ASIC Chip Synthesis, Springer, 2002.

**Course Outcomes:**

The student will be able to –

1. Code a design in HDL to the provided design specification
2. Compile, elaborate and simulate the design
3. Synthesize the coded design
4. Carry out the Floorplan of a design
5. Run placement, optimization, clock tree synthesis on the design
6. Write out a GDSII

# **ETH005: Real-Time Embedded Systems**

**Credits: 3** Teaching Scheme: 2 Hours / Week

Lab: 2 Hours / Week

**Section I**

**Unit 1: Introduction to Real-Time Systems & RTOS Basics**

Definition and characteristics of real-time systems, Types: Hard, Firm, and Soft real-time systems, RTOS vs General Purpose OS, Components of RTOS, Task management: States, creation, deletion, context switching

**Unit 2: RTOS Scheduling and Inter-Task Communication**

Task scheduling algorithms: Preemptive, Round-robin, Rate Monotonic, EDF, Priority inversion and priority inheritance, Semaphores, Mutexes, Message queues, Mailboxes, Event flags, Case study using FreeRTOS

**Unit 3: RTOS in Embedded Applications**

Memory management in RTOS, Real-time debugging and trace tools, Case study: Real-time data acquisition system, Overview of RTOS ports (FreeRTOS, uC/OS-II/III) on ARM Cortex-M, Writing RTOS-based embedded applications

**Section II:**

**Unit 4: Introduction to Embedded Linux**

Overview of Linux architecture, Difference between Desktop Linux and Embedded Linux, Kernel space vs User space, Toolchains and cross-compilation, Embedded Linux development environment (Buildroot, Yocto basics)

**Unit 5: Booting Process and File Systems**

Bootloader (U-Boot) and booting process, Root filesystem types (initramfs, ext3/4, NFS), Filesystem creation and mounting, Startup scripts (init, systemd), Kernel configuration and compilation

**Unit 6: Application Development and Peripheral Interface**

GPIO, UART, I2C, SPI interfacing on Embedded Linux, Writing and compiling user-space applications, Shell scripting for automation, Introduction to device drivers, Case study: Embedded Linux on Raspberry Pi / BeagleBone Black

**List of Experiments**

1. Task Creation & Scheduling using FreeRTOS – Create multiple tasks and demonstrate cooperative & preemptive scheduling.
2. Semaphore & Mutex in FreeRTOS – Use semaphore to synchronize two tasks.
3. Message Queue Communication – Implement inter-task communication using queues.
4. RTOS-Based Real-Time Application – Develop a temperature monitoring system with RTOS.
5. Linux Toolchain Setup & Cross-Compilation – Compile a basic C program using a cross-toolchain.
6. Booting Embedded Linux on ARM Board – Load Linux using U-Boot and verify boot process.
7. Shell Scripting on Embedded Linux – Write scripts to automate basic system functions.
8. Peripheral Interfacing (GPIO/UART/I2C) – Interface and control GPIO or UART using Linux user-space programs.
9. File System Creation and Mounting – Create a simple ext4 file system and mount it.
10. Mini Project – Develop a simple RTOS or Embedded Linux based application (e.g., home automation switch, data logger, etc.)

**Text Books:**

1. Raj Kamal, Embedded Systems: Architecture, Programming and Design, McGraw Hill Education
2. Jean J. Labrosse, MicroC/OS-II: The Real-Time Kernel, CMP Books
3. Chris Simmonds, Mastering Embedded Linux Programming, Packt Publishing

**Reference Books:**

1. Jack Ganssle, The Art of Designing Embedded Systems, Newnes
2. Douglas Schmidt, Real-Time Systems: Design Principles for Distributed Embedded Applications, Springer
3. Shibu K. V., Introduction to Embedded Systems, McGraw Hill
4. Yaghmour Karim, Building Embedded Linux Systems, O'Reilly Media

**Course Outcomes:**

The student will be able to:

1. CO1: Understand the fundamental concepts and requirements of Real-Time Operating Systems (RTOS).
2. CO2: Develop applications using RTOS concepts such as tasks, scheduling, semaphores, and message queues.
3. CO3: Analyze and implement real-time systems using RTOS for time-critical applications.
4. CO4: Understand the structure and components of Embedded Linux and its booting process.
5. CO5: Develop embedded applications using Linux with tools such as shell scripts and cross-compilation.
6. CO6: Interface peripherals and manage processes in Embedded Linux environments.

 **FF No. : 654**

# **ETH007: Introduction to Robotics**

**Total Credits: 3 Teaching Scheme:**

**Theory: 2 Hours / Week ;**

 **Laboratory: 2 Hours / Week;**

**Syllabus**

**Theory**

**Unit 1: Basics of Robotics (5 Hours)**

Definition and importance of robotics. Short history and key developments in robotics. Different types of robots like fixed, mobile, and humanoid. Main parts of a robot such as links, joints, and end-effectors. Concept of degrees of freedom in robots. Simple classification of robots by use like industrial or service. Basic components like sensors, actuators, and controllers. Advantages and common limitations of robots. Introduction to laws of robotics. Basic safety rules and ethical points in robot use.

**Unit 2: Robot Kinematics and Motion (5 Hours)**

Difference between Cartesian space and joint space in robot movement. Basics of forward kinematics for position calculation. Simple methods for solving inverse kinematics. Use of transformation matrices for position and orientation. Introduction to Denavit-Hartenberg (DH) parameters for robot modeling. Concept of kinematic chains and link connections. Workspace analysis for robot reachability. Overview of kinematic simulation tools.

**Unit 3: Sensors in Robotics (5 Hours)**

Purpose and types of sensors used in robots. Internal sensors like encoders, accelerometers, and gyroscopes for position and motion detection. External sensors like ultrasonic and infrared for environment sensing. Distance and proximity measurement techniques. Introduction to force and tactile sensors for object handling. Basics of environmental perception for robots. Signal conditioning methods for clean data. Simple concept of sensor fusion for combining data. Ways to interface sensors with microcontrollers. Basic real-time data reading methods.

**Unit 4: Actuators and Motion Control (5 Hours)**

Types of electrical actuators such as DC, servo, and stepper motors. Working of pneumatic and hydraulic actuators. Important features like torque, speed, and efficiency. Basic power transmission using gears, belts, and linkages. Introduction to motor driver circuits. Different control signals like PWM and digital control. Basic torque and speed control techniques. Simple feedback systems for better control. Measuring actuator performance like speed and stability. Selection of actuators for different robot tasks.

**Unit 5: Introduction to Robotic Software Tools (4 Hours)**

Use of software tools like MATLAB or RoboAnalyzer for design and testing robotic case studies. Simulation Tools: Introduction to robotic simulation environments like Gazebo, Webots, and V-REP. Understanding robot models (URDF), environments, and plugin systems. Running basic simulations and analyzing robot behavior in virtual environments.

**Unit 6: Safety, Communication, and Human Interaction (4 Hours)**

Robot Safety Standards: Introduction to safety protocols, ISO standards, and risk assessment in robotic systems. Understanding emergency stop systems, workspace safety, and fail-safe mechanisms. Application of safety design in industrial and collaborative robots. Human-Robot Interaction (HRI): Fundamentals of HRI design and ergonomics. Ethical considerations and designing socially aware, user-friendly robotic systems

**Laboratory**

**List of Experiments**

1. Introduction to Robot Types and Anatomy
2. Robot Modeling and Degree of Freedom Analysis
3. Robot Workspace Visualization Using Forward Kinematics
4. Solving Inverse Kinematics for 2-DOF Arm
5. Homogeneous Transformations and DH Parameter Assignment
6. Exploring Sensors in Robotics
7. Working with Proprioceptive Sensors
8. Interfacing and Controlling Electrical Actuators
9. Motor Driver Circuits and Feedback Control
10. Trajectory Planning and Simulation of a 3-DOF Arm

**Course Outcomes**

1. CO1: Describe the fundamental concepts, types, and structure of robots along with their historical evolution and ethical considerations.
2. CO2: Apply forward and inverse kinematics techniques to model the motion of robotic arms using transformation matrices and DH parameters.
3. CO3: Analyse the role of various sensors used in robotics by interfacing and interpreting data from proximity, force, and motion sensors.
4. CO4: Demonstrate the working principles of electrical, pneumatic, and hydraulic actuators, and evaluate their performance in robotic applications.
5. CO5: Simulate robot movements and trajectories in virtual environments and validate kinematic models through software-based experiments.
6. CO6: Design and implement basic robotic systems incorporating sensing, actuation, and motion control using appropriate software and hardware tools.

**Books and E-Resources**

 **For Reference Print Book -**

1. Textbooks:
	1. J. Craig, M. Spong; Introduction to Robotics: Mechanics and Control; 4th Ed.; Pearson; 2017; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us
	2. M. Groover, E. Zimmers; Industrial Robotics: Technology, Programming and Applications; 1st Ed.; McGraw-Hill; 2014; Accessed – May 10, 2025; Available: https://www.mheducation.com
	3. B. Siciliano, L. Sciavicco; Modelling and Control of Robot Manipulators; 2nd Ed.; Springer; 2010; DOI: 10.1007/978-1-84996-033-2
	4. R. Siegwart, I. Nourbakhsh; Introduction to Autonomous Mobile Robots; 2nd Ed.; MIT Press; 2011; Accessed – May 10, 2025; Available: https://mitpress.mit.edu
2. References:
	1. P. Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
	2. S. Niku; Introduction to Robotics: Analysis, Control, Applications; 2nd Ed.; Wiley; 2010; https://www.wiley.com
	3. K. Fu, R. Gonzalez; Robotics: Control, Sensing, Vision and Intelligence; McGraw-Hill; 1987
	4. J. Norberto; Robotics: Modelling, Planning and Control; Springer; 2006; DOI: 10.1007/978-1-84628-642-8
3. MOOCs and Other Learning Resources:
4. P. Newman, M. Fallon; Modern Robotics: Mechanics, Planning, and Control; Coursera; https://www.coursera.org/learn/modernrobotics; Accessed – May 10, 2025
5. M. J. Mataric; Control of Mobile Robots; Coursera; https://www.coursera.org/learn/mobile-robot; Accessed – May 10, 2025
6. S. Thrun; Programming a Robotic Car; Udacity; https://www.udacity.com/course/programming-a-robotic-car--cs373; Accessed – May 10, 2025
7. D. Rus, J. R. Walter; Robotics: Fundamentals; edX; https://www.edx.org/course/robotics-fundamentals; Accessed – May 10, 2025

FF 654

# **ETH008: Robotic Mechanisms and Control**

**Total Credits: 3 Teaching Scheme:**

**Theory: 2 Hours / Week ;**

 **Laboratory: 2 Hours / Week;**

**Syllabus**

**Theory**

**Unit 1: Dynamics of Robotics (5 Hours)**

Basic idea of robot motion and forces. Newton-Euler method for calculating robot forces and torques. Lagrangian method for modeling robot energy and movement. Calculation of inertia for robot links. Joint torque calculation for controlling robot motion. Basic force and torque control methods. Using software tools for robot dynamic simulation. Modeling the effect of friction on robot performance. Impact of payload changes on robot movements. Practical examples of robot dynamics in industry.

**Unit 2: Computer Vision for Robotics (5 Hours)**

Introduction to digital image processing for robots. Installing and using OpenCV for vision tasks. Capturing images and removing noise. Detecting edges and corners in images. Thresholding and segmenting images to find objects. Extracting features using SIFT and ORB methods. Recognizing objects using vision data. Calibrating cameras to reduce image distortion. Calculating depth with stereo vision. Controlling robot navigation using camera input.

**Unit 3: Artificial Intelligence for Robot Decision Making (5 Hours)**

Introduction to AI methods used in robotics. Search algorithms like A\* and Dijkstra for finding paths. Designing Finite State Machines (FSM) for robot task control. Using behavior trees for decision flow. Applying rule-based systems for robot actions. Basics of reinforcement learning for adaptive decisions. Using fuzzy logic for handling uncertain conditions. AI techniques for robot vision and perception. AI-based path planning for obstacle avoidance. Examples of autonomous AI-powered robots.

**Unit 4: Robotic Path Planning Methods (5 Hours)**

Understanding robot movement space and possible paths. Using roadmap and graph-based techniques for path planning. Applying artificial potential fields for obstacle navigation. Grid-based methods like Dijkstra and A\* for finding paths.

**Unit 5: Path Optimization and Path Planning Technique (4 Hours)**

Probabilistic Roadmaps (PRM) for motion planning. Using Rapidly-exploring Random Trees (RRT and RRT\*) for fast path generation. Real-time techniques for avoiding obstacles during movement. Methods for optimizing path smoothness and length.

**Unit 6: Advance Robotic Technologies (4 Hours)**

Simulating robot paths in virtual environments. Handling time constraints for real-time robot navigation. Dynamic Navigation,

**Laboratory**

**List of Experiments**

1. Open Loop vs Closed Loop Control Demonstration
2. Implementation and Tuning of a PID Controller
3. Transfer Function and Stability Analysis using MATLAB/Simulink
4. GPIO and Sensor Interfacing with Arduino
5. Motor Speed and Direction Control using PWM and Timers
6. Serial Communication using UART, I2C, and SPI Protocols
7. Programming Basic Motion and Obstacle Avoidance using Arduino
8. Line-Following Robot with Sensor-Based Control Logic
9. Robot Arm Simulation in Gazebo with ROS Integration
10. Creating URDF Files and Simulated Obstacle Detection in RViz

**Course Outcomes**

1. CO1: Analyze control system types, transfer functions, PID control, and system stability in time and frequency domains to optimize robotic performance.
2. CO2: Implement embedded system fundamentals including GPIO, PWM, and communication protocols for robotic hardware control using microcontrollers.
3. CO3: Develop and debug robotic programs using sensor-driven logic, motion control commands, and conditional algorithms on embedded platforms.
4. CO4: Simulate and evaluate robotic systems using Gazebo, ROS, URDF, and RViz to analyze robot motion and sensor integration in virtual environments.
5. CO5: Simulate robot movements and trajectories in virtual environments and validate kinematic models through software-based experiments.
6. CO6: Develop embedded code to control basic robot motion and implement obstacle avoidance using sensors and Construct and program a line-following robot utilizing sensor feedback and control algorithms.

**Books and E-Resources**

 **For Reference Print Book -**

Textbooks:

1. J. Craig, M. Spong; Introduction to Robotics: Mechanics and Control; 4th Ed.; Pearson; 2017; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us
2. M. Groover, E. Zimmers; Industrial Robotics: Technology, Programming and Applications; 1st Ed.; McGraw-Hill; 2014; Accessed – May 10, 2025; Available: https://www.mheducation.com
3. B. Siciliano, L. Sciavicco; Modelling and Control of Robot Manipulators; 2nd Ed.; Springer; 2010; DOI: 10.1007/978-1-84996-033-2
4. R. Siegwart, I. Nourbakhsh; Introduction to Autonomous Mobile Robots; 2nd Ed.; MIT Press; 2011; Accessed – May 10, 2025; Available: https://mitpress.mit.edu
5. References:
6. P. Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
7. S. Niku; Introduction to Robotics: Analysis, Control, Applications; 2nd Ed.; Wiley; 2010; https://www.wiley.com
8. K. Fu, R. Gonzalez; Robotics: Control, Sensing, Vision and Intelligence; McGraw-Hill; 1987
9. J. Norberto; Robotics: Modelling, Planning and Control; Springer; 2006; DOI: 10.1007/978-1-84628-642-8
10. MOOCs and Other Learning Resources:
11. P. Newman, M. Fallon; Modern Robotics: Mechanics, Planning, and Control; Coursera; https://www.coursera.org/learn/modernrobotics; Accessed – May 10, 2025
12. M. J. Mataric; Control of Mobile Robots; Coursera; https://www.coursera.org/learn/mobile-robot; Accessed – May 10, 2025
13. S. Thrun; Programming a Robotic Car; Udacity; https://www.udacity.com/course/programming-a-robotic-car--cs373; Accessed – May 10, 2025
14. D. Rus, J. R. Walter; Robotics: Fundamentals; edX; https://www.edx.org/course/robotics-fundamentals; Accessed – May 10, 2025

FF 654

# **ETH009: Advanced Robotics and Computer Vision**

**Total Credits: 3 Teaching Scheme:**

**Theory: 2 Hours / Week ;**

 **Laboratory: 2 Hours / Week;**

**Syllabus**

**Theory**

**Unit 1: Dynamics of Robotics (5 Hours)**

Basic concepts of robot motion and force interactions. Newton-Euler method for force and torque calculation. Lagrangian approach for energy-based dynamic modeling. Calculation of inertia matrices for robot links. Estimation of joint torques for movement control. Methods for force and torque control. Dynamic simulation of robot systems using software tools. Modeling of friction effects on robot parts. Influence of payload variation on robot performance. Real-world examples of robot dynamic responses.

**Unit 2: Computer Vision for Robotics (5 Hours)**

Basics of digital image processing for robots. Installation and setup of OpenCV for vision applications. Image capturing and filtering to reduce noise. Edge detection and corner point identification in images. Thresholding and image segmentation for object separation. Feature extraction using SIFT and ORB techniques. Object recognition using vision data. Camera calibration to correct image distortions. Depth calculation using stereo vision techniques. Navigation control using camera-based environment perception.

**Unit 3: Artificial Intelligence for Robot Decision Making (5 Hours)**

Introduction to artificial intelligence concepts in robotics. Search algorithms like A\* and Dijkstra for path selection. Design of Finite State Machines (FSM) for task control. Building behavior trees for decision management. Rule-based systems for robot actions. Basics of reinforcement learning for adaptive decision making. Application of fuzzy logic for uncertain environments. AI techniques for robot vision and perception tasks. AI methods for planning paths and avoiding obstacles. Case studies of AI-driven autonomous robots.

**Unit 4: Robotic Path Planning Methods (5 Hours)**

Understanding configuration space for robot movement. Roadmap-based and graph-based methods for path planning. Use of artificial potential fields for navigation control. Grid-based techniques like Dijkstra and A\* for path search. Probabilistic Roadmaps (PRM) for motion planning. Implementation of Rapidly-exploring Random Trees (RRT) and RRT\*. Real-time obstacle avoidance techniques during robot movement. Path length optimization and smoothing strategies. Virtual environment simulation of path planning tasks. Time management techniques for real-time path execution.

**Unit 5: Obstacle Avoidance Techniques (4 Hours)**

Real-time obstacle avoidance techniques during robot movement. Path length optimization and smoothing strategies. Virtual environment simulation of path planning tasks. Time management techniques for real-time path execution.

**Unit 6: Futuristic Robotic Technologies (4 Hours)**

Advance robotics communication. Autonomous delivery robots for logistics and services. Drone-based robotics for aerial applications. Robotic applications in Industry 4.0.

**Syllabus**

**Laboratory**

**List of Experiments**

1. Derive dynamic equations using Newton-Euler and simulate torque for 2-DOF robot
2. Compute Lagrangian dynamics and analyze effect of payload on robotic arm motion
3. Design and simulate force/torque control for robotic joint using MATLAB/Simulink
4. Perform basic image processing and feature extraction using OpenCV
5. Implement edge, corner detection, and segmentation on real-time video
6. Perform camera calibration and depth estimation using stereo vision setup
7. Implement A\* and Dijkstra search algorithms for grid-based robot navigation
8. Design a behavior tree for simple robotic decision-making task
9. Implement Rapidly-Exploring Random Trees (RRT) for obstacle-avoidance path
10. Simulate and compare potential field and probabilistic roadmap planning in ROS/MATLAB

**Course Outcomes**

1. CO1: Analyze and simulate robot dynamics using Newton-Euler and Lagrangian methods for evaluating joint torques, payload, and motion stability.
2. CO2: Apply computer vision techniques such as filtering, feature detection, calibration, and depth estimation for robotic perception and navigation.
3. CO3: Develop and implement AI-based decision-making systems using search algorithms, FSMs, and behavior trees for autonomous robots.
4. CO4: Design and simulate various robotic path planning techniques such as RRT and potential fields under real-time and constraint-based environments.
5. CO5: Integrate theoretical concepts with simulation tools to model, analyze, and control robotic systems effectively across dynamics, vision, and planning domains.
6. CO6: Develop, implement, and validate real-world robotics solutions using AI algorithms, computer vision, and advanced path planning techniques.

**Books and E-Resources**

 **For Reference Print Book -**

**Textbooks:**

1. J. Craig, M. Spong; Introduction to Robotics: Mechanics and Control; 4th Ed.; Pearson; 2017; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us
2. M. Groover, E. Zimmers; Industrial Robotics: Technology, Programming and Applications; 1st Ed.; McGraw-Hill; 2014; Accessed – May 10, 2025; Available: https://www.mheducation.com
3. B. Siciliano, L. Sciavicco; Modelling and Control of Robot Manipulators; 2nd Ed.; Springer; 2010; DOI: 10.1007/978-1-84996-033-2
4. R. Siegwart, I. Nourbakhsh, D. Scaramuzza; Introduction to Autonomous Mobile Robots; 2nd Ed.; MIT Press; 2011; Accessed – May 10, 2025; Available: https://mitpress.mit.edu
5. R. Szeliski; Computer Vision: Algorithms and Applications; 2nd Ed.; Springer; 2022; DOI: 10.1007/978-3-030-34372-9

**References:**

1. P. Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
2. S. Niku; Introduction to Robotics: Analysis, Control, Applications; 2nd Ed.; Wiley; 2010; Accessed – May 10, 2025; Available: https://www.wiley.com
3. K. Fu, R. Gonzalez, C. Lee; Robotics: Control, Sensing, Vision and Intelligence; McGraw-Hill; 1987
4. J. Norberto, B. Siciliano, L. Sciavicco; Robotics: Modelling, Planning and Control; Springer; 2006; DOI: 10.1007/978-1-84628-642-8
5. S. Russell, P. Norvig; Artificial Intelligence: A Modern Approach; 4th Ed.; Pearson; 2020; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us

**MOOCs and Other Learning Resources:**

1. **P. Newman, M. Fallon**; Modern Robotics: Mechanics, Planning, and Control; Coursera; https://www.coursera.org/learn/modernrobotics; Accessed – May 10, 2025
2. **M. J. Mataric**; Control of Mobile Robots; Coursera; https://www.coursera.org/learn/mobile-robot; Accessed – May 10, 2025
3. **S. Thrun**; Programming a Robotic Car; Udacity; https://www.udacity.com/course/programming-a-robotic-car--cs373; Accessed – May 10, 2025
4. **D. Rus, J. R. Walter**; Robotics: Fundamentals; edX; https://www.edx.org/course/robotics-fundamentals; Accessed – May 10, 2025
5. **A. Zisserman, A. Vedaldi**; Introduction to Computer Vision; Coursera; https://www.coursera.org/learn/introduction-computer-vision; Accessed – May 10, 2025

FF 654

# **ETH010: IoT and ML in Robotics**

**Total Credits: 3 Teaching Scheme:**

**Theory: 2 Hours / Week ;**

 **Laboratory: 2 Hours / Week;**

**Syllabus**

**Theory**

**Unit 1: Internet of Things (IoT) in Robotics (5 Hours)**

IoT system architecture used for robots. Types of sensors and actuators in IoT-enabled robots. Working of MQTT protocol for communication between devices. Design of interfaces using Blynk and Node-RED platforms. Methods for remote control and teleoperation of robots over the internet. IoT-based data logging and monitoring systems. Real-time communication between robots and cloud servers. Examples of smart robotics applications using IoT. Security risks, encryption, and data protection in IoT robotics. Case studies on IoT-based mobile robots.

**Unit 2: Machine Learning for Robotics (5 Hours)**

Introduction to machine learning techniques used in robotics. Supervised learning for object classification and decision tasks. Unsupervised learning for clustering and pattern finding. Decision tree and Support Vector Machine (SVM) for robot control. K-Nearest Neighbor (KNN) and simple clustering techniques. Use of regression models for robot behavior prediction. Techniques for selecting and extracting important features from data. Real-time prediction of robot actions based on input data. Preparation and use of training datasets for robotics. Full machine learning workflow from data collection to deployment.

**Unit 3: Mobile and Swarm Robotics (5 Hours)**

Design and working of wheeled and legged mobile robots. Basic kinematic modeling for controlling mobile robots. Simple control strategies for navigation, speed, and obstacle avoidance. Introduction to Simultaneous Localization and Mapping (SLAM). Concept of swarm intelligence for group robot coordination. Algorithms for formation control in robot teams. Leader-follower models for multi-robot tasks. Bio-inspired algorithms for distributed control. Consensus methods for joint decision making. Communication and coordination challenges in large robot groups.

**Unit 4: Robot Operating System (ROS 1) (5 Hours)**

Structure and communication model of Robot Operating System (ROS). Creation and running of ROS nodes for different tasks. Data sharing between nodes using topics and messages. Using ROS services for command-based control. Preparation and execution of launch files for starting robot processes. Writing and managing custom ROS packages for robot software development. Working with the ROS navigation stack for autonomous movement.

**Unit 5: ROS2 and AI Robotics (4 Hours)**

Introduction to ROS 2 with Data Distribution Service (DDS) communication. Methods for migrating applications from ROS 1 to ROS 2. Hardware integration and testing with ROS-controlled robots. Introduction to Artificial Neural Networks (ANN) in robotics. Basics of Deep Learning for object detection and recognition. Concept of Reinforcement Learning for robot decision making.

**Unit 6: Emerging Robotic Technologies (4 Hours)**

Collaborative robots (Cobots) working alongside humans. Soft robotics for flexible and adaptive tasks. Use of cloud robotics for remote intelligence and control. Development of humanoid and social robots. AI-powered autonomous delivery and service robots. Role of 5G and next-gen wireless in robotics communication. Autonomous drone technologies for inspection and delivery. Self-healing robotic systems using smart materials

**Laboratory**

**List of Experiments**

1. Remote Control of Robot Using MQTT & Blynk/Node-RED Interface
2. IoT-Based Sensor Data Logging and Visualization on Cloud
3. Object Classification using Supervised Learning (SVM/Decision Trees)
4. Real-time Obstacle Type Prediction using KNN or Clustering Techniques
5. Training and Deploying a Regression Model for Speed Prediction
6. Mobile Robot Navigation using SLAM and Path Following
7. Swarm Robot Formation and Leader-Follower Behavior
8. Bio-Inspired Control using Particle Swarm Optimization (PSO)
9. ROS 1 Node and Topic Communication with Sensor Integration
10. Migration of ROS1 Package to ROS2 and DDS Communication Test

**Course Outcomes**

1. CO1: Demonstrate the architecture, communication protocols, and secure integration of IoT components for remote robotic control and real-time data acquisition.
2. CO2: Apply supervised and unsupervised learning techniques to enable real-time prediction and decision-making capabilities in robotic systems.
3. CO3: Analyze and implement kinematic modeling, control strategies, and bio-inspired coordination techniques for mobile and multi-robot systems.
4. CO4: Develop and deploy robotic applications using ROS 1 and ROS 2, including node communication, package creation, and hardware integration.
5. CO5: Design and implement IoT and ML-based robotic systems for remote control, prediction, and cloud-based monitoring.
6. CO6: Develop and test robotic simulations and real-time applications using ROS frameworks and swarm behavior models.

**Books and E-Resources**

 **For Reference Print Book -**

**Textbooks:**

1. Rajkumar Buyya, Amir Vahid Dastjerdi; Internet of Things: Principles and Paradigms; 1st Ed.; Morgan Kaufmann; 2016;
2. Tom White; Hadoop: The Definitive Guide – Big Data and Machine Learning; 4th Ed.; O'Reilly Media; 2015;
3. Sebastian Thrun, Peter Norvig; Artificial Intelligence: A Modern Approach; 3rd Ed.; Pearson; 2010;
4. Morgan Quigley, Brian Gerkey, William D. Smart; Programming Robots with ROS; 1st Ed.; O’Reilly Media; 2015;

**References:**

1. Vijay Madisetti, Arshdeep Bahga; Internet of Things: A Hands-on Approach; 1st Ed.; VPT; 2014;
2. Peter Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
3. Russell C. Eberhart, Yuhui Shi, James Kennedy; Swarm Intelligence; 1st Ed.; Morgan Kaufmann; 2001;
4. Jason Brownlee; Machine Learning Mastery With Python; 1st Ed.; Machine Learning Mastery; 2016;

**MOOCs and Other Learning Resources:**

1. **IoT Programming and Big Data Analytics; NPTEL – Prof. Sudip Misra, IIT Kharagpur;https://onlinecourses.nptel.ac.in/noc22\_cs88/preview;**
2. **Machine Learning for Robotics; Coursera – Stanford University (Andrew Ng);**

**https://www.coursera.org/learn/machine-learning;**

1. **Robotics Software Engineering (ROS); Udacity; https://www.udacity.com/course/robotics-software-engineer--nd209;**

FF 654

# **ETH011: Special Topics in Robotics**

**Total Credits: 4 Teaching Scheme:**

**Theory: 2 Hours / Week ;**

 **Laboratory: 2 Hours / Week;**

**Tutorial: 1 hours/Week**

**Syllabus**

**Theory**

Unit 1: **Humanoid Robotics**

Design of humanoid robots, Joint coordination, Walking and balancing, Gait generation, Hand and facial gestures, Speech and vision integration, Human-robot interaction, Emotion recognition, Socially assistive robots, Rehabilitation robotics

Unit 2: **Micro and Nano Robotics**

Introduction to micro/nano robots, MEMS-based devices, Fabrication techniques, Power constraints, Actuation at small scales, Control mechanisms, Navigation in micro-environments, Biomedical applications, Magnetic and optical control, Challenges in nano-robotics

**Unit 3: Image Processing and SLAM**

Simultaneous Localization and Mapping, Visual SLAM algorithms, ORB-SLAM and RTAB-Map, Feature tracking, Map optimization, Sensor fusion for SLAM, Indoor/outdoor navigation, Loop closure detection, ROS SLAM implementation,Real-world case studies

Unit 4: **Robotic Applications in Industry and Healthcare**

Industrial automation robots, Robotic arms in manufacturing, AGVs and AMRs in logistics, Surgical robotics, Telemedicine and tele-robotics, Agricultural robots, Service robots, Mining and space robots, Rescue and disaster robots, Trends in robotic industries

Unit 5: **Robotics Ethics and Standards**

Ethical frameworks, Privacy and surveillance concerns, Military robotics debate, Autonomous weapons issues, Job displacement vs productivity, International standards (ISO, IEEE), Safety regulations, Human-in-the-loop control, Certification and testing, Future policy directions.

Unit 6: **Capstone Preparation**

Finalizing project scope, Literature review skills, Proposal writing, Budgeting and planning, Risk management, Mentor selection, Tools and frameworks, Version control systems, Design documentation, Review and feedback process

**Syllabus**

**Laboratory**

**List of Experiments**

* 1. Gait Generation for a Bipedal Robot using Simulated Environment (e.g., Gazebo/ROS)
	2. Design and Simulation of MEMS-Based Micro Robot with Actuation Control
	3. Real-Time Object Tracking and Facial Gesture Recognition using OpenCV and DLib
	4. Visual SLAM Implementation using ORB-SLAM or RTAB-Map in ROS
	5. Mobile Robot Navigation with Sensor Fusion (IMU + Lidar)
	6. ROS-Based Control of Robotic Arm for Pick-and-Place in Simulated Industry Setting
	7. Simulated Tele-Surgical Robot with Delay Modeling and Camera Feedback
	8. Case Study and Simulation: Search and Rescue Robot Navigation in Post-Disaster Zone
	9. Ethical Decision-Making Scenario Simulation in Autonomous Robots
	10. Capstone Proposal Presentation and Version Control Setup using Git/GitHub

**Course Outcomes**

1. CO1: Students will be able to design and implement IoT architectures for robotic systems using protocols like MQTT and interfaces like Blynk/Node-RED to achieve real-time remote control and cloud communication.
2. CO2: Students will be able to apply supervised and unsupervised machine learning techniques, including classification and regression models, to enable real-time decision-making in robotic systems.
3. CO3: Students will be able to analyze and implement control strategies for mobile robots and multi-agent systems, including SLAM, swarm intelligence, and bio-inspired algorithms for collaborative tasks.
4. CO4: Students will be able to develop and deploy robotic applications using ROS 1 and ROS 2 frameworks, incorporating topics, nodes, services, launch files, and hardware integration for real-world tasks.
5. CO5: Students will be able to simulate, implement, and evaluate robotic systems incorporating humanoid gait generation, SLAM, micro-robot actuation, and human-robot interaction techniques.
6. CO6: Students will be able to develop and present integrated robotic projects using ROS, SLAM, and ethical frameworks, demonstrating competence in real-world applications, version control, and proposal development.

**Books and E-Resources**

 **For Reference Print Book -**

**Textbooks:**

1. B. Siciliano, O. Khatib; Springer Handbook of Robotics; 2nd Ed.; Springer; 2016; DOI: 10.1007/978-3-319-32552-1
2. K. M. Lynch, F. C. Park; Modern Robotics: Mechanics, Planning, and Control; 1st Ed.; Cambridge University Press; 2017;
3. **S. Thrun, W. Burgard, D. Fox;** Probabilistic Robotics; 1st Ed.; MIT Press; 2005;
4. A. Dehon, M. A. Favorov, Y. Bar-Cohen; Micro- and Nano-Robotics: Technologies and Applications; 1st Ed.; Springer; 2010;

**References:**

1. J. Chestnutt, M. Lau, G. Nishiwaki, J. Kuffner; Humanoid Locomotion Planning for Dynamic Environments; Springer Tracts in Advanced Robotics; 2007;DOI: 10.1007/978-3-540-36124-1
2. G. Bekey; Autonomous Robots: From Biological Inspiration to Implementation and Control; 1st Ed.; MIT Press; 2005;
3. P. Corke; Robotics, Vision and Control: Fundamental Algorithms in MATLAB; 2nd Ed.; Springer; 2017;
4. M. J. Mataric; The Robotics Primer; 1st Ed.; MIT Press; 2007;

**MOOCs and Other Learning Resources:**

* 1. **D. Scaramuzza;** Visual SLAM for Autonomous Vehicles; edX (ETH Zurich);
	**URL:** https://www.edx.org/course/visual-perception-for-autonomous-driving
	2. **S. Thrun;** Programming a Robotic Car; Udacity;
	**URL:** https://www.udacity.com/course/programming-a-robotic-car--cs373
	3. **A. Saxena, P. Abbeel;** Artificial Intelligence for Robotics; Coursera;
	**URL:** https://www.coursera.org/learn/ai-for-robotics

FF 654

# **ETH012: Capstone Robotics Project**

**Total Credits: 4 Teaching Scheme:**

**Laboratory: Hours / Week;**

**Objective**

To enable students to integrate multi-disciplinary concepts in robotics, electronics, AI/ML, IoT, and computer vision to design and implement a full-fledged intelligent robotic system that addresses a real-world challenge or research problem.

Suggested Project Areas (based on prior semester topics)

1. AI in Robotics

* Intelligent robot assistants using NLP and computer vision
* Predictive maintenance bots using ML

2. IoT-Enabled Robotics

* Remote robotic control and monitoring systems for smart homes or industries
* IoT-enabled robotic greenhouse or smart surveillance system

3. Swarm & Convoy Robotics

* Multi-robot coordination for search & rescue missions
* Convoy of autonomous delivery robots

4. Humanoid & Social Robotics

* Assistive humanoid robots for elderly care or education
* Emotion-aware robotic interface using ML

5. Micro & Mobile Robotics

* Medical micro-bot navigation simulation
* Indoor mobile delivery bot with object detection

6. Vision-Based Path Planning and Control

* Drone path planning using deep reinforcement learning
* Obstacle-aware vision navigation robot with SLAM

7. ROS 1/2 Integration Projects

* Full ROS-based control system for warehouse robots
* ROS2 navigation stack-based mapping robot for autonomous ground vehicles

**Syllabus**

Expected Deliverables

1. Project Proposal Document (Problem statement, objectives, methodology, timeline)
2. Literature Survey (10–15 key references)
3. System Design and Implementation (Hardware & software architecture, modules)
4. Mid-Term Demonstration
5. Final Live Demo / Video Submission
6. Project Report (including test results and future scope)
7. Code Repository & Documentation (on GitHub/GitLab or equivalent)

Recommended Tools and Platforms

* Simulation: Gazebo, Webots, RViz, V-REP
* Programming: Python, ROS/ROS2, Arduino IDE, TensorFlow, OpenCV
* Hardware: Raspberry Pi, NVIDIA Jetson, ESP32, Arduino, sensors, actuators, camera modules
* AI/ML: Scikit-learn, Keras, PyTorch
* IoT: MQTT, Blynk, Node-RED, ThingSpeak

**Course Outcomes**

1. Develop intelligent robotic systems by integrating natural language processing and machine learning for real-time interaction and predictive functionality.
2. Design IoT-based robotic platforms for remote control and monitoring in smart home or industrial environments.
3. Implement coordinated multi-robot systems for collaborative tasks such as search and rescue or autonomous delivery.
4. Design emotion-aware humanoid robots capable of interacting with users in assistive or educational settings.
5. Develop autonomous robots with vision-based path planning and SLAM for real-time obstacle-aware navigation.
6. Implement a complete robotic system using ROS1/ROS2 for autonomous control, mapping, and navigation

FF No. : 654

# **ETH013: Data Science**

**Credits: 3 Teaching Scheme:**

 **Theory: 2 Hours / Week**

 **Lab: 2 Hours / Week**

**Section 1:**

**Introduction to Machine Learning & Python Basics**: Types of ML: Supervised, Unsupervised, Reinforcement, Python for ML: NumPy, pandas, matplotlib, Real-world applications and datasets

**Data Preprocessing and Visualization:** Handling missing data, encoding categorical data, Normalization, scaling, outlier detection, Data visualization techniques: histograms, pair plots

**Supervised Learning – Classification:** Logistic Regression, Decision Trees, k-NN, Confusion matrix, precision, recall, F1-score

**Section2:**

**Supervised Learning – Regression:** Linear and Polynomial Regression, Evaluation Metrics: MAE, MSE, R²

**Unsupervised Learning:** Clustering: k-Means, Hierarchical, Dimensionality Reduction: PCA, t-SNE

**Model Evaluation and Cross-Validation:** Bias-Variance Trade-off, k-Fold Cross-validation, Grid Search

**List of Practicals:**

1. Basic Data Analysis using pandas and matplotlib
2. Handling Missing Data and Encoding Categorical Variables
3. Data Normalization and Outlier Detection
4. Data Visualization with seaborn (pairplot, boxplot, heatmap)
5. Classification using k-NN and Confusion Matrix Evaluation
6. Logistic Regression for Binary Classification
7. Decision Trees with sklearn and Visual Interpretation
8. Linear Regression and Performance Metrics (MAE, MSE, R²)
9. Clustering using k-Means and Hierarchical Clustering with dendrograms
10. Model Tuning using Cross-Validation and GridSearchCV

**List of Project areas:**

1. Titanic Survival Prediction
2. Student Performance Predictor
3. Movie Genre Classification from Metadata
4. Customer Segmentation for E-Commerce
5. House Price Prediction
6. Credit Card Default Detection
7. Diabetes Prediction Model
8. Wine Quality Classification
9. Iris Flower Species Classifier
10. Retail Sales Pattern Analysis and Clustering

**Text Books:**

1. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd ed., O'Reilly Media, 2019.
2. J. VanderPlas, Python Data Science Handbook: Essential Tools for Working with Data, 1st ed., O'Reilly Media, 2016.

**Reference Books: *(As per IEEE format)***

1. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, 1st ed., MIT Press, 2016.
2. S. Raschka and V. Mirjalili, Python Machine Learning, 3rd ed., Packt Publishing, 2020.
3. T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, 2nd ed., Springer, 2009.

**Course Outcomes:**

The student will be able to –

1. Understand the basic concepts of Machine Learning and data science workflows using Python.
2. Apply data preprocessing techniques to clean, transform, and visualize data effectively.
3. Implement supervised learning algorithms such as logistic regression, k-NN, and decision trees for classification tasks.
4. Develop regression models and evaluate their performance using appropriate metrics.
5. Apply unsupervised learning methods such as clustering and dimensionality reduction to explore hidden patterns in data.
6. Evaluate and tune machine learning models using cross-validation, bias-variance analysis, and hyperparameter optimization.

# **ETH014: Deep Learning for Visual Recognition**

**Credits: 3 Teaching Scheme:**

 **Theory: 2 Hours / Week**

 **Lab: 2 Hours / Week**

**Section 1:**

**Fundamentals of Deep Learning:** Artificial Neural Networks, Activation functions, Backpropagation and loss functions, Initialization and optimization techniques

**Convolutional Neural Networks (CNNs):** Convolution, Pooling, ReLU, Flatten, Fully connected layers, Architecture: LeNet, VGGNet

**Advanced CNN Architectures:** ResNet, Inception, MobileNet, Transfer learning and pre-trained models

**Section2:**

**Image Classification and Object Detection:** Training pipelines, data augmentation, YOLO basics, SSD introduction

**Model Evaluation and Overfitting:** Evaluation metrics: Accuracy, mAP, ROC, AUC

Overfitting control: Dropout, Regularization, Early stopping

**Frameworks: TensorFlow and PyTorch:** Model definition and training in PyTorch, Transfer learning workflow in TensorFlow, Comparison of TensorFlow and PyTorch

**List of Practicals:**

1. Implement a basic Artificial Neural Network using NumPy from scratch for binary classification.
2. Train and evaluate a CNN model (e.g., LeNet) on the MNIST dataset using TensorFlow or PyTorch.
3. Apply data augmentation techniques (e.g., rotation, flipping, cropping) to improve CNN generalization.
4. Compare performance of VGGNet, ResNet, and MobileNet on CIFAR-10 dataset using transfer learning.
5. Train an image classifier using transfer learning with a pre-trained MobileNet or ResNet model.
6. Implement object detection using YOLOv5 or SSD on a small custom dataset or COCO subset.
7. Evaluate a classification model using metrics such as confusion matrix, accuracy, precision, recall, F1-score, and ROC curve.
8. Demonstrate overfitting control techniques such as dropout and early stopping on a CNN model.
9. Build and train a CNN in PyTorch, visualize training/validation loss and accuracy.
10. Develop a TensorFlow-based transfer learning pipeline, including freezing layers and fine-tuning.

**List of Project areas:**

1. Real-time Sign Language Recognition using CNN and hand gesture datasets.
2. Face Mask Detection System using YOLO or SSD with webcam input.
3. Fruit or Plant Disease Classification using deep CNNs and transfer learning.
4. Facial Emotion Detection System using ResNet and FER2013 dataset.
5. Vehicle Detection and Counting System for traffic monitoring.
6. Animal Species Classification using wildlife image datasets (e.g., iNaturalist).
7. Personalized Fashion Recommendation System using CNNs on clothing images.
8. Document Scanner and OCR Enhancer using deep learning and pre-trained models.
9. Helmet and Safety Gear Detection System for industrial or traffic environments.
10. Visual Quality Inspection System for defect detection in manufacturing.

**Text Books:**

1. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, 1st ed., MIT Press, 2016.
2. F. Chollet, Deep Learning with Python, 2nd ed., Manning Publications, 2021.

**Reference Books:**

1. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed., O'Reilly Media, 2022.
2. D. Foster, Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play, 2nd ed., O'Reilly Media, 2022.
3. V. Dumoulin and A. Visin, A Guide to Convolution Arithmetic for Deep Learning, arXiv preprint arXiv:1603.07285, 2016.

**Course Outcomes:**

The student will be able to –

1. Understand the foundational principles of deep learning including neural network architectures, activation functions, and optimization techniques.
2. Implement and train Convolutional Neural Networks (CNNs) for image classification tasks using modern frameworks such as TensorFlow and PyTorch.
3. Analyse and compare advanced CNN architectures like ResNet, Inception, and MobileNet for visual recognition tasks.
4. Apply transfer learning and pre-trained models to solve real-world computer vision problems.
5. Evaluate deep learning models using appropriate metrics and apply techniques to mitigate overfitting.
6. Design and develop end-to-end deep learning solutions for visual recognition applications such as object detection using YOLO or SSD.

 **FF No. : 654**

# **ETH015: Deep Neural Networks for Natural Language Processing**

**Credits: 3 Teaching Scheme:**

 **Theory: 2 Hours / Week**

 **Lab: 2 Hours / Week**

**Section 1:**

**Introduction to NLP and Deep Learning (4 hours):** Overview of NLP tasks: text classification, NER, translation, summarization, Basic pipeline: tokenization, stemming, lemmatization, stopword removal, Word embeddings: One-hot, Word2Vec (CBOW & Skip-gram), GloVe, Cosine similarity, analogy tasks

**Sequence Modeling with RNNs and LSTMs (5 hours):** RNN architecture and backpropagation through time (BPTT), Limitations of RNNs: vanishing gradient, LSTM and GRU architectures, Implementing text generation and sentiment analysis with LSTM

**Attention Mechanisms and Transformers Basics (5 hours):** Limitations of RNN-based models for long sequences, Self-attention mechanism: scaled dot-product attention, Introduction to the Transformer architecture, Positional encoding and multi-head attention (intuitive overview)

**Section2:**

**Pretrained Language Models (5 hours):** Concept of transfer learning in NLP, Overview of popular models: BERT, GPT, RoBERTa, DistilBERT, Tokenization methods: WordPiece, SentencePiece, Fine-tuning for downstream tasks: text classification, QA, NER

**Applications and Case Studies (5 hours):** Sentiment Analysis, Machine Translation (using Transformer or MarianMT), Text Summarization: extractive vs. abstractive, Named Entity Recognition using Hugging Face models, Zero-shot classification using LLMs

**Ethics, Bias, and Interpretability in NLP (4 hours):** Bias and fairness in NLP models, Toxicity, gender/racial bias in LLMs, Explainability tools for NLP: LIME, SHAP, Integrated Gradients, Best practices for dataset curation and responsible AI in NLP

**List of Practicals:**

1. Implement tokenization, stemming, lemmatization, and stopword removal using NLTK or spaCy.
2. Train Word2Vec (CBOW & Skip-gram) and visualize embeddings using PCA or t-SNE.
3. Build and train an LSTM model for binary text classification (e.g., movie reviews sentiment analysis).
4. Implement a basic NER tagger using a pre-trained spaCy or Hugging Face model.
5. Train an LSTM for text generation (e.g., generate Shakespeare-like text).
6. Code a scaled dot-product attention module in PyTorch or TensorFlow and test on a toy dataset.
7. Fine-tune BERT for sentiment classification using Hugging Face Transformers.
8. Use a pre-trained MarianMT model to translate between English and another language.
9. Implement extractive and abstractive summarization using Hugging Face models like BART or T5.
10. Use LIME or SHAP to interpret predictions from a fine-tuned text classifier.

**List of Project areas:**

1. Fine-tune BERT or RoBERTa to classify news articles as fake or real.
2. Develop a sequence-to-sequence chatbot using Transformers trained on college-related queries.
3. Use XLM-RoBERTa to perform sentiment analysis across multiple languages.
4. Extract entities and skills from resumes and match them to job descriptions using NLP techniques.
5. Build a system to detect toxic or offensive language using fine-tuned LLMs.
6. Use T5 or GPT to generate quiz questions from educational content.
7. Build a zero-shot classifier using Hugging Face’s pipeline for multi-domain categorization.
8. Implement an abstractive summarization engine for real-time news feeds using BART.
9. Deploy a MarianMT-based translation system as a Flask or Streamlit web app.
10. Analyze customer feedback or product reviews to extract themes, emotions, and actionable insights.

**Text Books:**

1. D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed., Pearson, 2023 (draft chapters available online).
2. T. Wolf et al., “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 EMNLP: System Demonstrations*, pp. 38–45, Association for Computational Linguistics, 2020.

**Reference Books:**

1. M. Eisen, Effective NLP with Transformers, 1st ed., O’Reilly Media, 2022.
2. T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent trends in deep learning-based natural language processing,” IEEE Computational Intelligence Magazine, vol. 13, no. 3, pp. 55–75, 2018.

**Course Outcomes:**

The student will be able to –

1. Understand core natural language processing tasks and apply standard text preprocessing techniques.
2. Implement word embeddings and sequence models such as RNNs, LSTMs, and GRUs for text data.
3. Comprehend the architecture of transformers and attention mechanisms for handling long-range dependencies in NLP tasks.
4. Fine-tune and evaluate pretrained language models (BERT, GPT, etc.) for various downstream NLP applications.
5. Analyze and build solutions for real-world NLP problems such as text classification, NER, summarization, and translation.
6. Apply ethical AI principles to identify and mitigate bias and fairness issues in deep NLP systems.

 **FF No. : 654**

# **ETH016: Large Language Models (LLMs) and Generative AI**

**Credits: 3 Teaching Scheme:**

 **Theory: 2 Hours / Week**

 **Lab: 2 Hours / Week**

**Section 1:**

**Introduction to Large Language Models and Generative AI (5 hours):** Evolution of Language Models: From N-grams to Transformers, What are LLMs? Architecture overview (GPT, BERT, T5), Generative AI: Definitions and key concepts, Types of generative models: Autoregressive, Autoencoder, Diffusion models, Overview of datasets, compute, and scale for LLM training

**Transformer Architectures and Scaling Laws (5 hours):** Detailed architecture of GPT-style decoder-only models, Masked language models (BERT) vs. autoregressive models (GPT); Training objectives: MLM, Causal LM, Seq2Seq; Scaling laws: Impact of model size, dataset size, compute ;Tokenization and vocabulary design (Byte Pair Encoding, SentencePiece)

**Training and Fine-tuning Large Language Models (4 hours):** Pretraining strategies and compute infrastructure; Transfer learning and fine-tuning techniques; Prompt engineering and in-context learning; Techniques for efficient training: Mixed precision, model parallelism; Challenges: Catastrophic forgetting, bias, hallucinations

**Section2:**

**Applications of LLMs and Generative AI (5 hours):** Text generation, summarization, translation, and question answering; Code generation with LLMs (e.g., Codex); Conversational AI and chatbots; Multimodal generative AI: Text-to-image, text-to-audio; Case studies of popular GenAI tools and platforms (OpenAI, Anthropic, etc.)

**Responsible Use and Ethical Considerations (3 hours):** Bias and fairness in generative models; Misinformation, disinformation, and hallucination issues; Privacy concerns and data provenance; Legal and societal implications; Responsible AI frameworks and mitigation strategies

**Future Trends and Hands-on Overview (6 hours):** Emerging architectures: Mixture of Experts, Retrieval-Augmented Generation (RAG); Foundation models beyond language (multimodal, robotics); Practical session: Using Hugging Face transformers for text generation; Fine-tuning small LLMs on custom datasets; Deploying LLM-powered applications and APIs

**List of Practicals:**

1. Explore pre-trained language models using Hugging Face Transformers (e.g., BERT, GPT-2)
2. Generate text using OpenAI's GPT (via API or Hugging Face)
3. Implement prompt engineering for tasks like Q&A, summarization
4. Fine-tune a small language model on a custom dataset (e.g., using LoRA or PEFT)
5. Analyze and visualize attention weights in transformer models
6. Compare results of masked (BERT) vs. autoregressive (GPT) language models
7. Perform zero-shot and few-shot learning using LLMs
8. Deploy a text generation model as a REST API
9. Use LLMs for code generation (e.g., with CodeGen or Codex)
10. Implement a basic chatbot using a fine-tuned transformer model

**List of Project areas:**

1. Fine-tuning a LLM for domain-specific QA (e.g., healthcare, legal)
2. Generative AI for automated content creation (e.g., social media captions, blogs)
3. Building a multilingual translation system using a transformer
4. LLM-powered intelligent code assistant for programming help
5. Detecting and mitigating bias in LLM-generated text
6. Retrieval-Augmented Generation (RAG) chatbot for personalized responses
7. Data-to-text generation system for dynamic report generation
8. LLM-based resume screening and job-matching tool
9. Multimodal GenAI: Combining text-to-image and image captioning
10. Exploring hallucination and factual consistency in generative models

**Text Books: *(As per IEEE format)***

1. J. Brownlee, Deep Learning for Natural Language Processing, 1st ed., Machine Learning Mastery, 2017.

**Reference Books: *(As per IEEE format)***

1. T. Wolf et al., “Transformers: State-of-the-art Natural Language Processing,” EMNLP 2020: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pp. 38–45, 2020.
2. B. R. Rout, M. K. Mohapatra, and S. Mohapatra, Generative AI with Python and TensorFlow 2, 1st ed., BPB Publications, 2023.
3. S. Bommasani et al., “On the Opportunities and Risks of Foundation Models,” Stanford CRFM Report, 2021.

**Course Outcomes:**

The student will be able to –

1. Understand the architecture and principles behind large language models and generative AI systems.
2. Analyze and implement transformer-based architectures and fine-tuning techniques.
3. Utilize LLMs for various real-world tasks including text generation, summarization, and chatbot development.
4. Address and mitigate ethical issues like bias, misinformation, and hallucinations in generative models.
5. Work with practical toolkits like Hugging Face to deploy and interact with LLMs.
6. Explore and build applications leveraging next-generation generative AI capabilities.

 **FF No. : 654**

# **ETH017: Edge AI and Cloud-based Systems for Real-Time Deep Learning**

**Credits: 4 Teaching Scheme:**

 **Theory: 2 Hours / Week**

 **Lab: 2 Hours / Week**

 **Tutorial: 1 Hour/Week**

**Section 1:**

**Introduction to Edge AI and Cloud AI (5 hours):** Edge AI overview: importance, challenges, and use cases; Cloud AI fundamentals: cloud services, scalability, and management; Edge vs. Cloud: trade-offs in latency, bandwidth, power, and privacy; Overview of hardware: Edge devices (Raspberry Pi, Jetson Nano, Coral TPU), Cloud GPUs/TPUs; Introduction to model compression: quantization, pruning, distillation

**Deep Learning Model Optimization for Deployment (5 hours):** Model compression techniques in detail: pruning, quantization, knowledge distillation; Lightweight architectures for edge: MobileNet, EfficientNet, TinyML concepts; Frameworks for deployment: TensorFlow Lite, ONNX, OpenVINO, PyTorch Mobile; Batch size, precision, and latency optimization; Benchmarking and profiling models on target devices

**Cloud-based Deployment and Services (4 hours):** Cloud platforms overview: AWS, Azure, Google Cloud AI services; Using serverless functions and containers for AI inference; Model serving frameworks: TensorFlow Serving, TorchServe, NVIDIA Triton; Auto-scaling and load balancing in cloud AI systems; Monitoring, logging, and managing deployed AI services

**Section2:**

**Real-Time Systems and Edge-Cloud Integration (5 hours):** Real-time constraints and system design principles; Edge-Cloud collaboration: hybrid architectures, data pipelines; Communication protocols: MQTT, gRPC, REST APIs; Case studies: autonomous vehicles, smart surveillance, IoT applications; Latency reduction techniques: caching, local inference fallback

**Security, Privacy, and Ethical Considerations (3 hours):** Security challenges in edge and cloud AI deployment; Data privacy and GDPR compliance for AI systems; Federated learning and on-device learning for privacy preservation; Ethical concerns in real-time AI applications; Best practices for secure and compliant AI deployments

**Hands-on Project and Future Trends (6 hours):** Setting up an edge device for real-time inference (e.g., Jetson Nano or Raspberry Pi); Deploying a compressed deep learning model on edge hardware; Cloud deployment of a deep learning model with auto-scaling; Integrating edge and cloud inference in a pipeline; Emerging trends: TinyML, 5G-enabled AI, AI accelerators

**List of Practicals:**

1. Setting up an Edge AI device (Raspberry Pi / Jetson Nano) with required software for AI inference.
2. Running a pre-trained deep learning model for image classification on an edge device.
3. Implementing model pruning on a CNN and evaluating performance trade-offs on edge hardware.
4. Applying quantization techniques to reduce model size and measuring inference latency.
5. Converting and deploying a TensorFlow model using TensorFlow Lite on an Edge device.
6. Using PyTorch Mobile to deploy a compressed model on an edge device.
7. Benchmarking model performance (accuracy, latency, power consumption) on different edge devices (Raspberry Pi, Jetson Nano, Coral TPU).
8. Deploying a deep learning model on a cloud platform using TensorFlow Serving and testing auto-scaling.
9. Creating a simple edge-cloud hybrid pipeline using MQTT or REST APIs for inference data exchange.
10. Implementing basic federated learning simulation for privacy-preserving AI on distributed devices.

**List of Tutorials:**

1. Setup and configure Raspberry Pi for Edge AI applications, including OS installation and AI environment setup.
2. Build and train a lightweight CNN model (MobileNet/EfficientNet) optimized for edge deployment using TensorFlow or PyTorch.
3. Apply model pruning and quantization techniques to compress a deep learning model and measure its performance on edge hardware.
4. Deploy a deep learning model on NVIDIA Jetson Nano, running inference and benchmarking latency and accuracy.
5. Convert TensorFlow models to TensorFlow Lite format and deploy them on an edge device for real-time inference.
6. Export deep learning models to ONNX format and run them on multiple deployment platforms using ONNX Runtime.
7. Create serverless AI inference services using AWS Lambda or Azure Functions and test their scalability.
8. Implement MQTT communication between edge devices and cloud servers for real-time data streaming in IoT systems.
9. Simulate federated learning using TensorFlow Federated to enable privacy-preserving distributed model training.
10. Containerize deep learning models using Docker and deploy them on Kubernetes with auto-scaling for cloud inference services.

**List of Project areas:**

1. Real-time object detection and alert system for smart surveillance using Edge AI.
2. Low-latency voice command recognition system deployed on edge devices for smart homes.
3. Autonomous vehicle sensor data fusion and decision-making in a hybrid Edge-Cloud architecture.
4. Privacy-preserving health monitoring system using federated learning on wearable devices.
5. Intelligent traffic monitoring and congestion prediction using edge and cloud collaboration.
6. Energy-efficient predictive maintenance system for industrial IoT using TinyML.
7. Cloud-based scalable AI model serving for retail demand forecasting.
8. AI-powered drone navigation and obstacle avoidance with edge inference.
9. Smart agriculture monitoring with edge-based pest detection and cloud analytics.
10. AI-based fraud detection system for financial transactions using cloud AI services and real-time alerts on edge devices.

**Text Books:**

1. S. S. Kanhere, Edge AI: Convergence of Edge Computing and Artificial Intelligence, 1st ed., Wiley, 2022.
2. A Joshi, Deep Learning on Cloud and Edge, 1st ed., Springer, 2021.

**Reference Books:**

1. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, 1st ed., MIT Press, 2016.
2. M. T. Goodrich and R. Tamassia, Introduction to Algorithms, 3rd ed., Pearson, 2009.
3. F. Chollet, Deep Learning with Python, 2nd ed., Manning, 2021.

**Course Outcomes:**

The student will be able to –

1. Analyze and compare the characteristics, benefits, and limitations of Edge AI and Cloud AI systems.
2. Apply advanced model compression techniques such as pruning, quantization, and knowledge distillation for efficient AI deployment on edge devices.
3. Utilize lightweight neural network architectures and relevant deployment frameworks for optimized edge AI applications.
4. Design and implement scalable AI inference services on cloud platforms using containerization and serverless architectures.
5. Develop hybrid edge-cloud AI systems that meet real-time constraints, optimize latency, and ensure secure data communication.
6. Incorporate privacy-preserving techniques including federated learning and on-device learning in AI systems to comply with ethical and regulatory standards.

 **FF No. : 654**

# **ETH018: Capstone Project in AI Systems**

**Credits: 4 Teaching Scheme: …. Hours / Week**

This course focuses on hands-on projects that integrate cutting-edge techniques in Vision, Language, and Speech processing. Students will apply deep learning and generative AI models to design, develop, and deploy solutions for real-world problems across diverse domains such as healthcare, autonomous systems, conversational agents, multimedia, etc.

**Section 1:**

Project Planning and Design: The capstone project begins with defining a clear problem statement and conducting an in-depth literature review to establish the research background. This phase focuses on understanding the problem domain and designing appropriate models tailored to address the identified challenges.

**Section 2:**

Implementation, Evaluation, and Reporting: In this phase, students train their models and carry out thorough evaluation to assess performance and practical applicability. The project concludes with deploying the solution, delivering a formal presentation, and submitting a comprehensive report documenting the methodology, experiments, and results.

**Course Outcomes:**

The student will be able to –

1. Formulate problem statements based on real-world challenges.
2. Conduct comprehensive literature reviews to identify existing solutions and research gaps.
3. Design and train deep learning models integrating multimodal data.
4. Evaluate model performance using evaluation metrics and benchmarks.
5. Deploy AI systems with considerations for efficiency, scalability, and usability.
6. Prepare technical reports and present findings effectively.

# **ETD001: Fundamentals of Digital Electronics**

**Credits:4 Teaching Scheme: 2 Hours / Week Lab: 2 Hours / Week**

Section 1:

**Unit 1: Number Systems, Binary Arithmetic, Boolean Algebra**

Introduction to number systems: binary, octal, decimal, hexadecimal, Conversion between number systems, Binary arithmetic: addition, subtraction, multiplication, division,1’s and 2’s complement representation and arithmetic, Boolean algebra: laws and theorems, Simplification using Boolean identities

**Unit 2: Logic Gates, K-Map Simplification, Combinational Logic Circuits**

Basic logic gates: AND, OR, NOT, Universal gates: NAND, NOR, XOR, XNOR Logic gate ICs and truth tables, Karnaugh Maps (K-Maps): 2, 3, and 4 variables, Minimization of Boolean expressions, Designing combinational logic circuits

**Unit 3: Adders, Subtractors, Multiplexers, Demultiplexers**

Half adder and full adder circuits, Half subtractor and full subtractor circuits, 4:1 and 8:1 multiplexers, cascading, 1:4 and 1:8 demultiplexers, Realization of logic functions using MUX/DEMUX

**Section2:**

**Unit 4: Flip-Flops, Counters, Shift Registers**

Types of flip-flops: SR, JK, D, T, Master-slave flip-flops, Asynchronous and synchronous counters, Mod-n counters and applications, Shift registers: SISO, SIPO, PISO, PIPO

**Unit 5: Memory Elements, PLDs, Finite State Machines**

ROM, RAM, EEPROM concepts, Memory decoding and addressing, Introduction to PLDs: PAL, PLA, Finite State Machines (FSMs): Moore and Mealy models, State transition diagrams and tables

**Unit 6: Digital Logic Design using Simulators (Logisim/Proteus)**

Introduction to simulation tools: Logisim, Proteus, Designing and testing combinational logic, Simulating sequential circuits: counters, shift registers, FSM simulation, Creating virtual testbenches and debugging

**List of Practicals:**

1. Number System Conversion and Binary Arithmetic
2. Implementation of Basic Logic Gates using ICs
3. Design and Verification of Combinational Circuits
4. Half Adder and Full Adder Design
5. Multiplexer and Demultiplexer Applications
6. Flip-Flop Operation and Verification
7. Asynchronous and Synchronous Counters
8. Shift Register Simulation
9. FSM (Finite State Machine) Design
10. Memory Simulation

**List of Project areas:**

1. Digital Locks and Authentication Systems

2. Code Converters and Display Circuits

3. Control and Counting Systems

4. Sequence Generators and Detectors

5. Voting and Quiz Systems

**Text Books: *(As per IEEE format)***

1. *M. Morris Mano – Digital Logic and Computer Design*
2. *R.P. Jain – Modern Digital Electronics*

**Reference Books: *(As per IEEE format)***

1. *Thomas L. Floyd – Digital Fundamentals*
2. *Ronald J. Tocci, Neal S. Widmer – Digital Systems: Principles and*
3. *John F. Wakerly – Digital Design: Principles and Practices*

**Course Outcomes:**

The student will be able to –

1. CO1: Understand different number systems and perform binary arithmetic operations.
2. CO2: Apply Boolean algebra and simplification techniques for logic design.
3. CO3: Design and implement basic combinational logic circuits such as adders, multiplexers, etc.
4. CO4: Analyze and design sequential circuits using flip-flops, counters, and shift registers.
5. CO5: Describe memory components and programmable logic devices with real-world applications.
6. CO6: Simulate and test digital circuits using modern tools such as Logisim or Proteus.

 **FF No. : 654**

# **ETD002: Introduction to Microcontrollers and Embedded C**

**Credits:4 Teaching Scheme: 2 Hours / Week Lab: 2 Hours / Week**

**Section 1:**

**Unit 1: Microprocessors vs Microcontrollers, Architecture**

Difference between microprocessor and microcontroller, Applications and role of microcontrollers in embedded systems, Overview of 8051/AVR/ARM Cortex-M architecture (focus on block diagram and functional units)

**Unit 2: Memory Mapping, I/O Ports, SFRs**

Harvard vs Von Neumann architecture, Memory organization and data/program separation, I/O port structure and access, Introduction to Special Function Registers (SFRs)

**Unit 3: Instruction Set and Addressing Modes**

Instruction classification: data transfer, arithmetic, logic, control, Addressing modes: immediate, direct, indirect, register. Simple program writing using embedded C for arithmetic and logic operations

**Section 2:**

**Unit 4: Basics of Embedded C, Bitwise Operations**

Embedded C structure: headers, main(), variables, Operators, control structures (if, while, for), Bitwise operations and masking techniques, Importance of volatile and static keywords in embedded context

**Unit 5: Port Programming, LED & Switch Interfacing**

Digital I/O concepts, LED control using port registers, Switch debouncing techniques and interfacing, Toggling LED based on input using polling method

**Unit 6: Timer, Delay, and Interrupt Programming**

Delay generation using timer registers, Timer modes and applications (Timer 0, Timer 1), Introduction to external interrupts, Writing ISRs (Interrupt Service Routines) in Embedded C

**List of Practicals:**

1. LED Blinking using Delay Loop
2. Switch and LED Interface
3. LED Blinking using Timer Delay
4. Switch-controlled LED using Interrupts
5. 7-Segment Display Interfacing
6. Port Reading and Bit Masking
7. Toggle LED at Fixed Intervals (Timer-based)
8. Password Entry System using Keypad
9. Counter with Button and LED Display
10. Mini Application: Digital Stopwatch using Timer

**List of Project areas:**

1. Digital Input/Output Systems

2. Password and Keypad-Based Projects

3. Timer and Interrupt-Based Control

4. Display and Indication Systems

5. Mini Automation Systems

**Text Books: *(As per IEEE format)***

1. Muhammad Ali Mazidi, The 8051 Microcontroller and Embedded Systems, Pearson
(A comprehensive and hands-on guide to embedded C and 8051)
2. Raj Kamal, Embedded Systems: Architecture, Programming and Design, McGraw Hill

**Reference Books: *(As per IEEE format)***

1. Kenneth Ayala, The 8051 Microcontroller, Cengage Learning
2. Han-Way Huang, The AVR Microcontroller and Embedded Systems
3. Michael J. Pont, Embedded C, Pearson Education
4. Shibu K.V., Introduction to Embedded Systems, McGraw Hill
5. John Morton, The PIC Microcontroller: An Introduction to Software and Hardware Interfacing

**Course Outcomes:**

The student will be able to –

1. CO1: Understand the architecture and key components of microcontrollers.
2. CO2: Differentiate between microprocessors and microcontrollers in embedded applications.
3. CO3: Interpret memory organization and I/O mechanisms in microcontrollers.
4. CO4: Write embedded C code for performing basic data operations and port
5. programming.
6. CO5: Interface digital inputs and outputs with microcontroller ports.
7. CO6: Develop simple embedded applications using timers and interrupts.

 **FF No. : 654**

# **ETD003: Embedded System Interfacing and Applications**

**Credits:4 Teaching Scheme: 2 Hours / Week Lab: 2 Hours / Week**

**Section 1:**

**Unit 1: Interfacing LEDs, Switches, Buzzers**

GPIO concepts in microcontrollers, Active high/low logic, Interfacing push-button switches and buzzers

**Unit 2: LCD and Keypad Interfacing**

16x2 LCD pinout and commands, 4x4 matrix keypad interfacing,Scanning technique and display integration

**Unit 3: ADC and DAC Interfacing**

Sampling, quantization basics, ADC interfacing and programming (e.g., LM35 temperature sensor), DAC basics and waveform generation

**Section 2:**

**Unit 4: Serial Communication (UART, SPI, I2C)**

UART protocol and serial port setup, SPI communication basics, I2C communication and sensor integration

**Unit 5: Motor and Relay Interfacing**

DC motor control using transistor/relay, PWM generation using timers, Servo and stepper motor interfacing

**Unit 6: Embedded Applications and Testing**

Use of debugging tools (serial monitor, logic analyzer), Case studies on small embedded control systems, Energy-efficient embedded design techniques

**List of Practicals:**

1. LED and Buzzer interfacing
2. Switch-based counter
3. 16x2 LCD display interface
4. 4x4 Keypad input scanning
5. Temperature sensor (LM35) with ADC
6. DAC waveform generation
7. UART-based PC communication
8. SPI-based EEPROM data storage
9. DC Motor control using relay
10. Servo motor positioning with PWM

**List of Project areas:**

1. Password-protected access system
2. Digital temperature-controlled fan
3. Home automation switch control
4. Wireless sensor data logger
5. Mini vending machine controller

**Text Books: *(As per IEEE format)***

1. Mazidi – The 8051 Microcontroller and Embedded Systems
2. John Uffenbeck – Microcomputers and Microprocessors

**Reference Books: *(As per IEEE format)***

1. Shibu K.V. – Introduction to Embedded Systems
2. Han-Way Huang – Embedded System Design with C8051
3. Raj Kamal – Embedded Systems

**Course Outcomes:**

The student will be able to –

1. CO1: Interface digital and analog peripherals with microcontrollers.
2. CO2: Operate display and keypad modules through port programming.
3. CO3: Read analog inputs using ADC and generate analog signals with DAC.
4. CO4: Implement serial communication protocols like UART, SPI, and I2C.
5. CO5: Control actuators like motors and relays using PWM and logic signals.
6. CO6: Test and debug small-scale embedded systems with standard tools.

 **FF No. : 654**

# **ETD004: Embedded System Development with Arduino**

**Credits:4 Teaching Scheme: 2 Hours / Week Lab: 2 Hours / Week**

**Section 1:**

**Unit 1: Arduino Platform and IDE**

Arduino boards overview (UNO, Nano, Mega), Arduino IDE and sketch structure, Digital Read, Digital Write, pin Mode basics

**Unit 2: Sensor Interfacing**

DHT11 (humidity/temperature), IR sensor, Ultrasonic sensor basics and distance calculation, Motion sensor (PIR) integration

**Unit 3: Actuator and Display Control**

Relays and LEDs, Controlling DC/Servo motors, OLED/7-segment display modules

**Section 2:**

**Unit 4: Serial and Wireless Communication**

Serial monitor communication, Bluetooth module (HC-05), Wi-Fi module (ESP8266/NodeMCU intro)

**Unit 5: IoT Applications and Data Logging**

ThingSpeak/IFTTT basics, SD card data logging, Real-time monitoring dashboards

**Unit 6: Mini Project Development**

Problem definition and use-case design, Circuit design, code development, Testing and final report preparation

**List of Practicals:**

1. LED control using pushbutton
2. DHT11-based temperature display
3. Distance measurement using Ultrasonic sensor
4. Motor control with potentiometer
5. Light-activated LED using LDR
6. IR remote-controlled device
7. Bluetooth-controlled relay
8. Real-time sensor data to ThingSpeak
9. SD card data logger
10. Mini Arduino project integration

**List of Project areas:**

1. Smart dustbin with ultrasonic sensor
2. Smart blind stick
3. Home automation using mobile app
4. Voice-controlled appliances
5. Weather monitoring and logging system

**Text Books: *(As per IEEE format)***

1. Simon Monk – Programming Arduino: Getting Started with Sketches
2. Michael Margolis – Arduino Cookbook

**Reference Books: *(As per IEEE format)***

1. John Boxall – Arduino Workshop
2. Banzi & Shiloh – Getting Started with Arduino
3. Tyler James – Arduino Projects for Beginners

**Course Outcomes:**

The student will be able to –

1. CO1: Develop programs using the Arduino IDE.
2. CO2: Interface various analog and digital sensors to Arduino.
3. CO3: Control actuators and display elements through code.
4. CO4: Implement wireless communication for data transfer.
5. CO5: Design simple IoT applications and data logging systems.
6. CO6: Build and present working prototypes of embedded projects.

 **FF No. : 654**

# **ETD005: Advanced Embedded Systems with Raspberry Pi**

**Credits:4 Teaching Scheme:**

 **Theory: 2 Hours / Week Lab: 2 Hours / Week Tutorial: 1 Hours/week**

**Section 1:**

**Unit 1: Raspberry Pi Architecture and OS**

Raspberry Pi models and hardware specs, Raspberry Pi OS and setup Terminal basics and file management

**Unit 2: Python Programming for GPIO**

Basics of Python (variables, loops, functions), GPIO pin configuration using RPi.GPIO, LED and switch control with Python

**Unit 3: Sensor Interfacing with Python**

Temperature, light, and motion sensors, Reading and displaying data on terminal/LCD, Real-time logging with timestamps

**Section 2:**

**Unit 4: Web Server and Remote Access**

Setting up Apache web server, Controlling GPIO via web interface SSH, VNC, and remote desktop

**Unit 5: Camera and Audio Interfacing**

Raspberry Pi camera setup, Capturing image/video using Python Playing/recording audio using Python libraries

**Unit 6: Final Project Planning and Deployment**

System design for IoT solution, Code modularization and testing Documentation and deployment

**List of Practicals:**

1. GPIO LED blinking using Python
2. Pushbutton to toggle LED
3. Reading DHT11 and displaying on LCD
4. LDR-based lighting control
5. Host web server and GPIO toggle
6. Remote access via VNC/SSH
7. Image capture using Pi Camera
8. Motion detection with PIR
9. Voice recording and playback
10. Final IoT deployment using Raspberry Pi

**List of Project areas:**

1. Raspberry Pi-based smart surveillance
2. IoT weather dashboard
3. Web-controlled home appliances
4. Automated greenhouse monitor
5. Face recognition door lock system

**Text Books: *(As per IEEE format)***

1. Eben Upton – Raspberry Pi User Guide
2. Simon Monk – Programming the Raspberry Pi

**Reference Books: *(As per IEEE format)***

1. Derek Molloy – Exploring Raspberry Pi
2. Wolfram Donat – Learn Raspberry Pi Programming
3. Ashwin Pajankar – Raspberry Pi Cookbook for Python Programmers

**Course Outcomes:**

The student will be able to –

1. CO1: Understand Raspberry Pi architecture and OS setup.
2. CO2: Write Python programs for GPIO-based control and sensing.
3. CO3: Interface and manage real-world sensors on Raspberry Pi.
4. CO4: Host local web servers and enable remote monitoring.
5. CO5: Handle multimedia interfaces like camera and audio.
6. CO6: Design and deploy end-to-end embedded applications.

**FF No. : 654**

# **ETD007: Fundamental of Robotics**

**Teaching Scheme:**

Theory: 2 Hours / Week ;

Laboratory: 2 Hours / Week;

Total Credits: 3

**Syllabus**

**Theory**

**Unit 1: Fundamentals of Robotics** - ( 6 Hours)

Definition, scope, and multidisciplinary nature of robotics, Historical evolution and key milestones in robotics development, Classification of robots: Fixed, Mobile, Humanoid, Swarm, and Collaborative robots, Robot anatomy: Links, joints, end-effectors, and structural configurations, Degrees of Freedom (DOF): Concept and calculation across robot types, Application-based classification: Industrial, Medical, Service, and Military robots, Essential components: Controllers, sensors, actuators, power supplies, and software, Advantages, limitations, and emerging trends in robotics, Laws of Robotics: Asimov’s laws and modern ethical guidelines, Safety standards, operational hazards, and risk mitigation in robotic systems

**Unit 2: Kinematics of Robots -** ( 6 Hours)

Cartesian space vs. Joint space representations, Forward kinematics: Mathematical modeling and position analysis, Inverse kinematics: Analytical and numerical solutions, Homogeneous transformation matrices: Representation and manipulation, Denavit-Hartenberg (DH) convention for kinematic modelling, Kinematic chains, links, and joint coordinate frames, Workspace analysis: Reachable space, dexterity, and manipulability, Redundant manipulators: Control challenges and optimization, Basics of trajectory planning: Position, velocity, and acceleration profiles, Simulation of kinematic models using tools like MATLAB, RoboAnalyzer, or ROS

**Unit 3: Sensors and Interfacing -** ( 6 Hours)

Fundamentals of robotic sensing and perception, Proprioceptive sensors: Encoders, IMUs, gyroscopes, and accelerometers, Exteroceptive sensors: Ultrasonic, infrared, LIDAR, and vision sensors, Distance and proximity measurement techniques, Force, torque, and tactile sensing for manipulation tasks, Environmental perception: 2D/3D mapping and object detection, Signal conditioning: Amplification, filtering, and data conversion, Sensor integration with microcontrollers and robotic platforms, Basics of sensor fusion: Kalman filtering and multi-sensor data handling, Real-time data acquisition and interfacing techniques

**Unit 4: Actuators and Drives** - ( 6 Hours)

Electrical actuators: DC motors, Servo motors, and Stepper motors, Pneumatic and hydraulic actuators: Working principles and applications, Performance characteristics: Torque, speed, load capacity, and efficiency, Power transmission elements: Gears, belts, chains, and linkages, Motor driver circuits: H-bridge, ESC, and other control interfaces, Control signal generation: PWM, analog, and digital control techniques, Torque and speed control strategies: Open-loop vs Closed-loop, Feedback mechanisms: Position, velocity, and current sensing, Performance evaluation metrics: Response time, stability, and accuracy, Selection criteria for actuators based on robotic application

**List of Experiments**

1. Introduction to Robotics: Types, Structure, and Applications
2. Robot Kinematic Modeling and Degrees of Freedom (DOF) Analysis
3. Forward Kinematics: Workspace Determination and Visualization
4. Inverse Kinematics Solutions for a Planar 2-DOF Robotic Arm
5. Homogeneous Transformation Matrices and Denavit-Hartenberg (DH) Parameter Assignment
6. Introduction to Sensors in Robotics: Types and Applications
7. Implementation and Analysis of Proprioceptive Sensors
8. Electrical Actuators: Interfacing and Control Techniques
9. Motor Driver Circuits and Closed-Loop Feedback Control Systems
10. Trajectory Planning and Motion Simulation for a 3-DOF Robotic Manipulator

**Course Outcomes**

1. CO1: Describe the fundamental concepts, types, and structure of robots along with their historical evolution and ethical considerations.
2. CO2: Apply forward and inverse kinematics techniques to model the motion of robotic arms using transformation matrices and DH parameters.
3. CO3: Analyse the role of various sensors used in robotics by interfacing and interpreting data from proximity, force, and motion sensors.
4. CO4: Demonstrate the working principles of electrical, pneumatic, and hydraulic actuators, and evaluate their performance in robotic applications.
5. CO5: Simulate robot movements and trajectories in virtual environments and validate kinematic models through software-based experiments.
6. CO6: Design and implement basic robotic systems incorporating sensing, actuation, and motion control using appropriate software and hardware tools.

 **For Reference Print Book -**

**Textbooks:**

1. J. Craig, M. Spong; Introduction to Robotics: Mechanics and Control; 4th Ed.; Pearson; 2017; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us
2. M. Groover, E. Zimmers; Industrial Robotics: Technology, Programming and Applications; 1st Ed.; McGraw-Hill; 2014; Accessed – May 10, 2025; Available: https://www.mheducation.com
3. B. Siciliano, L. Sciavicco; Modelling and Control of Robot Manipulators; 2nd Ed.; Springer; 2010; DOI: 10.1007/978-1-84996-033-2
4. R. Siegwart, I. Nourbakhsh; Introduction to Autonomous Mobile Robots; 2nd Ed.; MIT Press; 2011; Accessed – May 10, 2025; Available: https://mitpress.mit.edu

**References:**

1. P. Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
2. S. Niku; Introduction to Robotics: Analysis, Control, Applications; 2nd Ed.; Wiley; 2010; https://www.wiley.com
3. K. Fu, R. Gonzalez; Robotics: Control, Sensing, Vision and Intelligence; McGraw-Hill; 1987
4. J. Norberto; Robotics: Modelling, Planning and Control; Springer; 2006; DOI: 10.1007/978-1-84628-642-8

**MOOCs and Other Learning Resources:**

1. P. Newman, M. Fallon; Modern Robotics: Mechanics, Planning, and Control; Coursera; https://www.coursera.org/learn/modernrobotics; Accessed – May 10, 2025
2. M. J. Mataric; Control of Mobile Robots; Coursera; https://www.coursera.org/learn/mobile-robot; Accessed – May 10, 2025
3. S. Thrun; Programming a Robotic Car; Udacity; https://www.udacity.com/course/programming-a-robotic-car--cs373; Accessed – May 10, 2025
4. D. Rus, J. R. Walter; Robotics: Fundamentals; edX; https://www.edx.org/course/robotics-fundamentals

FF:654

# **ETD008: Control Systems and Embedded Programming for Intelligent Robotics**

Teaching Scheme:

Theory: 2 Hours / Week ;

Laboratory: 2 Hours / Week;

Total Credits: 3

**Syllabus**

**Theory**

**Unit 1: Foundations of Control Systems-**

Taxonomy of control system architectures; Distinction between open-loop and closed-loop paradigms; Fundamental doctrines of feedback mechanisms; Derivation and manipulation of transfer functions; Analytical techniques in both temporal and spectral domains; Implementation of Proportional-Integral-Derivative (PID) regulatory schemes; State-space formulation and its multidimensional representation; Rigorous stability assessment methodologies; Deployment of sensors within feedback control loops; Algorithmic approaches for system tuning and parametric optimization.

**Unit 2: Microcontroller Architectures and Embedded Computational Frameworks in Robotics**

Conceptualization and architecture of microcontrollers (Arduino, STM32 platforms); General Purpose Input/Output (GPIO) interfacing schematics; Pulse Width Modulation (PWM) methodologies for actuation control; Analog and digital sensor interfacing protocols; Utilization of timers and hardware interrupts; Master-slave synchronous and asynchronous communication protocols: I2C, SPI, UART; Techniques for embedded power governance; Fundamentals of Embedded C programming for real-time control systems; Diagnostic and debugging instrumentation; Constraint modeling for real-time embedded systems in robotics.

**Unit 3: Algorithmic Robot Programming and Behavioral Control Scripting**

Programming constructs for motion primitives (translation, rotation, halting); Logical structures employing conditional statements and iterative loops; Modular programming methodologies; Formulation of obstacle avoidance algorithms; Sensor-driven behavioral orchestration; Development of line-following heuristics; Operational differentiation between teleoperated and autonomous robotic modes; Source code optimization strategies for execution efficiency; Advanced debugging methodologies for embedded robotic codebases; Practical code deployment on Arduino and Raspberry Pi platforms.

**Unit 4: Computational Modeling and Simulation Tools in Robotics**

Comprehensive overview of contemporary robotic simulation environments; Foundational introduction to Gazebo simulation suite; Kinematic and dynamic simulation of articulated robotic manipulators; Configuration of virtual sensors and synthetic environmental setups; Preliminary exposure to Robot Operating System (ROS) ecosystem; Authoring of Unified Robot Description Format (URDF) models; Visualization and data interpretation using RViz modules; Algorithmic motion planning in simulated domains; Virtualized obstacle detection methodologies; Quantitative and qualitative performance metrics for simulation-based validation.

**Syllabus**

**Laboratory**

**List of Experiments**

1. Open Loop vs Closed Loop Control Demonstration
2. Implementation and Tuning of a PID Controller
3. Transfer Function and Stability Analysis using MATLAB/Simulink
4. GPIO and Sensor Interfacing with Arduino
5. Motor Speed and Direction Control using PWM and Timers
6. Serial Communication using UART, I2C, and SPI Protocols
7. Programming Basic Motion and Obstacle Avoidance using Arduino
8. Line-Following Robot with Sensor-Based Control Logic
9. Robot Arm Simulation in Gazebo with ROS Integration
10. Creating URDF Files and Simulated Obstacle Detection in RViz

**Course Outcomes**

1. CO1: Analyze control system types, transfer functions, PID control, and system stability in time and frequency domains to optimize robotic performance.
2. CO2: Implement embedded system fundamentals including GPIO, PWM, and communication protocols for robotic hardware control using microcontrollers.
3. CO3: Develop and debug robotic programs using sensor-driven logic, motion control commands, and conditional algorithms on embedded platforms.
4. CO4: Simulate and evaluate robotic systems using Gazebo, ROS, URDF, and RViz to analyze robot motion and sensor integration in virtual environments.
5. CO5: Simulate robot movements and trajectories in virtual environments and validate kinematic models through software-based experiments.
6. CO6: Develop embedded code to control basic robot motion and implement obstacle avoidance using sensors and Construct and program a line-following robot utilizing sensor feedback and control algorithms.

**Books and E-Resources**

 **For Reference Print Book -**

**Textbooks:**

1. J. Craig, M. Spong; Introduction to Robotics: Mechanics and Control; 4th Ed.; Pearson; 2017; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us
2. M. Groover, E. Zimmers; Industrial Robotics: Technology, Programming and Applications; 1st Ed.; McGraw-Hill; 2014; Accessed – May 10, 2025; Available: https://www.mheducation.com
3. B. Siciliano, L. Sciavicco; Modelling and Control of Robot Manipulators; 2nd Ed.; Springer; 2010; DOI: 10.1007/978-1-84996-033-2
4. R. Siegwart, I. Nourbakhsh; Introduction to Autonomous Mobile Robots; 2nd Ed.; MIT Press; 2011; Accessed – May 10, 2025; Available: https://mitpress.mit.edu

 **References:**

1. P. Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
2. S. Niku; Introduction to Robotics: Analysis, Control, Applications; 2nd Ed.; Wiley; 2010; https://www.wiley.com
3. K. Fu, R. Gonzalez; Robotics: Control, Sensing, Vision and Intelligence; McGraw-Hill; 1987
4. J. Norberto; Robotics: Modelling, Planning and Control; Springer; 2006; DOI: 10.1007/978-1-84628-642-8

**MOOCs and Other Learning Resources:**

1. P. Newman, M. Fallon; Modern Robotics: Mechanics, Planning, and Control; Coursera; https://www.coursera.org/learn/modernrobotics; Accessed – May 10, 2025
2. M. J. Mataric; Control of Mobile Robots; Coursera; https://www.coursera.org/learn/mobile-robot; Accessed – May 10, 2025
3. S. Thrun; Programming a Robotic Car; Udacity; https://www.udacity.com/course/programming-a-robotic-car--cs373; Accessed – May 10, 2025
4. D. Rus, J. R. Walter; Robotics: Fundamentals; edX; https://www.edx.org/course/robotics-fundamentals; Accessed – May 10, 2025

**FF: 654**

# **ETD009: Advanced Robotics: Dynamics, Perception, Artificial Intelligence, and Path Planning**

Teaching Scheme:

Theory: 2 Hours / Week ; Laboratory: 2 Hours / Week;

Total Credits: 3

**Syllabus**

**Theory**

**Unit 1: Analytical Dynamics of Robotic Systems-**

Newton-Euler algorithmic frameworks; Lagrangian energy-based formulations; Inertia tensor and mass matrix computations; Articulation of joint-space torques; Force and torque regulatory schemes; Computational models for dynamic simulation; Frictional parameterization and modeling; Load-dependent dynamic behavior and payload variations; Kinematic stability during motion trajectories; Empirical case studies on robotic dynamic responses.

**Unit 2: Computational Vision Paradigms for Robotic Perception-**

Foundational constructs in digital image processing; Installation and operationalization of OpenCV libraries; Image acquisition pipelines and noise filtration techniques; Gradient-based edge detection and corner point extraction; Global and adaptive thresholding procedures; Region-based image segmentation algorithms; Feature extraction using Scale-Invariant Feature Transform (SIFT) and Oriented FAST and Rotated BRIEF (ORB); Object detection and recognition pipelines; Geometric camera calibration methodologies; Depth map computation and stereo vision frameworks; Vision-assisted navigational control strategies.

**Unit 3: Artificial Intelligence Methodologies in Robotic Autonomy** **-**

Artificial intelligence paradigms in autonomous decision frameworks; Classical and heuristic-based search algorithms: A\*, Dijkstra’s algorithm; Finite State Machine (FSM) constructs for behavioral control; Behavior trees for hierarchical decision structures; Rule-based inferential systems; Elementary reinforcement learning concepts for robotic adaptation; Implementation of fuzzy logic controllers; AI-driven perception modules; AI integration in path planning algorithms; Domain-specific AI deployment case studies in autonomous robotics.

**Unit 4: Algorithmic Approaches to Robotic Path Planning-**

Multi-dimensional configuration space representations; Roadmap generation and graph-theoretic path planning techniques; Artificial potential field methodologies; Grid-based discretization and occupancy mapping algorithms; Probabilistic Roadmap (PRM) strategies; Rapidly-exploring Random Trees (RRT) and optimized RRT\* variants; Real-time obstacle avoidance frameworks; Trajectory smoothing and path length optimization techniques; Computational simulation of path planning scenarios; Temporal constraints and real-time execution considerations.

**Syllabus**

**Laboratory**

**List of Experiments**

1. Derive dynamic equations using Newton-Euler and simulate torque for 2-DOF robot
2. Compute Lagrangian dynamics and analyze effect of payload on robotic arm motion
3. Design and simulate force/torque control for robotic joint using MATLAB/Simulink
4. Perform basic image processing and feature extraction using OpenCV
5. Implement edge, corner detection, and segmentation on real-time video
6. Perform camera calibration and depth estimation using stereo vision setup
7. Implement A\* and Dijkstra search algorithms for grid-based robot navigation
8. Design a behavior tree for simple robotic decision-making task
9. Implement Rapidly-Exploring Random Trees (RRT) for obstacle-avoidance path
10. Simulate and compare potential field and probabilistic roadmap planning in ROS/MATLAB

**Course Outcomes**

1. CO1: Analyze and simulate robot dynamics using Newton-Euler and Lagrangian methods for evaluating joint torques, payload, and motion stability.
2. CO2: Apply computer vision techniques such as filtering, feature detection, calibration, and depth estimation for robotic perception and navigation.
3. CO3: Develop and implement AI-based decision-making systems using search algorithms, FSMs, and behavior trees for autonomous robots.
4. CO4: Design and simulate various robotic path planning techniques such as RRT and potential fields under real-time and constraint-based environments.
5. CO5: Integrate theoretical concepts with simulation tools to model, analyze, and control robotic systems effectively across dynamics, vision, and planning domains.
6. CO6: Develop, implement, and validate real-world robotics solutions using AI algorithms, computer vision, and advanced path planning techniques.

**Books and E-Resources**

 **For Reference Print Book -**

**Textbooks:**

1. J. Craig, M. Spong; Introduction to Robotics: Mechanics and Control; 4th Ed.; Pearson; 2017; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us
2. M. Groover, E. Zimmers; Industrial Robotics: Technology, Programming and Applications; 1st Ed.; McGraw-Hill; 2014; Accessed – May 10, 2025; Available: https://www.mheducation.com
3. B. Siciliano, L. Sciavicco; Modelling and Control of Robot Manipulators; 2nd Ed.; Springer; 2010; DOI: 10.1007/978-1-84996-033-2
4. R. Siegwart, I. Nourbakhsh, D. Scaramuzza; Introduction to Autonomous Mobile Robots; 2nd Ed.; MIT Press; 2011; Accessed – May 10, 2025; Available: https://mitpress.mit.edu
5. R. Szeliski; Computer Vision: Algorithms and Applications; 2nd Ed.; Springer; 2022; DOI: 10.1007/978-3-030-34372-9

**References:**

1. P. Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
2. S. Niku; Introduction to Robotics: Analysis, Control, Applications; 2nd Ed.; Wiley; 2010; Accessed – May 10, 2025; Available: https://www.wiley.com
3. K. Fu, R. Gonzalez, C. Lee; Robotics: Control, Sensing, Vision and Intelligence; McGraw-Hill; 1987
4. J. Norberto, B. Siciliano, L. Sciavicco; Robotics: Modelling, Planning and Control; Springer; 2006; DOI: 10.1007/978-1-84628-642-8
5. S. Russell, P. Norvig; Artificial Intelligence: A Modern Approach; 4th Ed.; Pearson; 2020; Accessed – May 10, 2025; Available: https://www.pearson.com/en-us

**MOOCs and Other Learning Resources:**

1. **P. Newman, M. Fallon**; Modern Robotics: Mechanics, Planning, and Control; Coursera; https://www.coursera.org/learn/modernrobotics; Accessed – May 10, 2025
2. **M. J. Mataric**; Control of Mobile Robots; Coursera; https://www.coursera.org/learn/mobile-robot; Accessed – May 10, 2025
3. **S. Thrun**; Programming a Robotic Car; Udacity; https://www.udacity.com/course/programming-a-robotic-car--cs373; Accessed – May 10, 2025
4. **D. Rus, J. R. Walter**; Robotics: Fundamentals; edX; https://www.edx.org/course/robotics-fundamentals; Accessed – May 10, 2025
5. **Zisserman, A. Vedaldi**; Introduction to Computer Vision; Coursera; https://www.coursera.org/learn/introduction-computer-vision; Accessed – May 10, 2025

**FF:654**

# **ETD010: Intelligent Robotic Systems: IoT, Machine Learning, Mobile Robotics, and ROS Frameworks**

Teaching Scheme:

Theory: 2 Hours / Week ; Laboratory: 2 Hours / Week;

Total Credits: 3

**Syllabus**

**Theory**

**Unit 1: Internet of Things (IoT) Integration in Robotic Systems**

Layered IoT architectural frameworks tailored for robotic platforms; Deployment of heterogeneous sensors and actuators within IoT-enabled robots; Implementation and utilization of Message Queuing Telemetry Transport (MQTT) communication protocols; Graphical interface configuration using Blynk and Node-RED environments; Remote operability and teleoperation control of robotic systems; IoT-driven telemetry and data logging mechanisms; Real-time bidirectional communication with cloud infrastructures; Application domains encompassing smart robotics and cyber-physical interactions; Security vulnerabilities, encryption strategies, and authentication challenges within IoT for robotics; Case studies demonstrating IoT-based control in autonomous and mobile robotic agents.

**Unit 2: Machine Learning Algorithms for Robotic Intelligence**

Conceptual overview of machine learning paradigms in robotics; Supervised learning frameworks for classification and regression tasks; Unsupervised learning methodologies for clustering and pattern discovery; Decision tree induction and Support Vector Machine (SVM) classification mechanisms; K-Nearest Neighbor (KNN) algorithms and centroid-based clustering techniques; Linear and non-linear regression modeling; Dimensionality reduction, feature selection, and feature extraction processes; Real-time inferencing and prediction for adaptive robotic behaviors; Compilation and preprocessing of training datasets tailored for robotic platforms; End-to-end machine learning pipeline for robotics: Data ingestion, training, validation, and deployment phases.

**Unit 3: Mobile Robotics and Swarm-Based Collective Behaviors**

Design and operational principles of wheeled and legged mobile robotic configurations; Kinematic modeling and trajectory estimation for mobile platforms; Control strategies for navigation, path following, and obstacle avoidance; Foundational concepts of Simultaneous Localization and Mapping (SLAM); Swarm intelligence methodologies for distributed robotic coordination; Geometric and algorithmic models for formation control; Leader-follower behavior protocols within multi-agent systems; Bio-inspired optimization algorithms for collective task execution; Consensus algorithms for distributed decision-making; Architectural and communication challenges in scalable multi-robot systems.

**Unit 4: Robotic Operating System (ROS 1 and ROS 2) Architectures**

Software architecture and operational paradigms of ROS middleware; Configuration and deployment of ROS nodes and inter-process communication topics; ROS message structures and service-oriented communication patterns; Formulation and execution of ROS launch files for modular system orchestration; Development and packaging of custom ROS modules and packages; Utilization of ROS navigation stack for autonomous path planning; Architectural distinctions and communication models in ROS 2 utilizing Data Distribution Service (DDS); Migration strategies and interoperability frameworks between ROS 1 and ROS 2 ecosystems; Hardware abstraction and integration of robotic sensors and actuators within ROS-controlled platforms.

**Laboratory**

**List of Experiments**

1. Remote Control of Robot Using MQTT & Blynk/Node-RED Interface
2. IoT-Based Sensor Data Logging and Visualization on Cloud
3. Object Classification using Supervised Learning (SVM/Decision Trees)
4. Real-time Obstacle Type Prediction using KNN or Clustering Techniques
5. Training and Deploying a Regression Model for Speed Prediction
6. Mobile Robot Navigation using SLAM and Path Following
7. Swarm Robot Formation and Leader-Follower Behavior
8. Bio-Inspired Control using Particle Swarm Optimization (PSO)
9. ROS 1 Node and Topic Communication with Sensor Integration
10. Migration of ROS1 Package to ROS2 and DDS Communication Test

**Course Outcomes**

1. CO1: Demonstrate the architecture, communication protocols, and secure integration of IoT components for remote robotic control and real-time data acquisition.
2. CO2: Apply supervised and unsupervised learning techniques to enable real-time prediction and decision-making capabilities in robotic systems.
3. CO3: Analyze and implement kinematic modeling, control strategies, and bio-inspired coordination techniques for mobile and multi-robot systems.
4. CO4: Develop and deploy robotic applications using ROS 1 and ROS 2, including node communication, package creation, and hardware integration.
5. CO5: Design and implement IoT and ML-based robotic systems for remote control, prediction, and cloud-based monitoring.
6. CO6: Develop and test robotic simulations and real-time applications using ROS frameworks and swarm behavior models.

**Books and E-Resources**

 **For Reference Print Book -**

**Textbooks:**

1. Rajkumar Buyya, Amir Vahid Dastjerdi; Internet of Things: Principles and Paradigms; 1st Ed.; Morgan Kaufmann; 2016;
2. Tom White; Hadoop: The Definitive Guide – Big Data and Machine Learning; 4th Ed.; O'Reilly Media; 2015;
3. Sebastian Thrun, Peter Norvig; Artificial Intelligence: A Modern Approach; 3rd Ed.; Pearson; 2010;
4. Morgan Quigley, Brian Gerkey, William D. Smart; Programming Robots with ROS; 1st Ed.; O’Reilly Media; 2015;

**References:**

1. Vijay Madisetti, Arshdeep Bahga; Internet of Things: A Hands-on Approach; 1st Ed.; VPT; 2014;
2. Peter Corke; Robotics, Vision and Control; 2nd Ed.; Springer; 2017; DOI: 10.1007/978-3-319-54413-7
3. Russell C. Eberhart, Yuhui Shi, James Kennedy; Swarm Intelligence; 1st Ed.; Morgan Kaufmann; 2001;
4. Jason Brownlee; Machine Learning Mastery With Python; 1st Ed.; Machine Learning Mastery; 2016;

MOOCs and Other Learning Resources:

* 1. **IoT Programming and Big Data Analytics; NPTEL – Prof. Sudip Misra, IIT Kharagpur;https://onlinecourses.nptel.ac.in/noc22\_cs88/preview;**
	2. **Machine Learning for Robotics; Coursera – Stanford University (Andrew Ng);**
	3. **https://www.coursera.org/learn/machine-learning;**
	4. **Robotics Software Engineering (ROS); Udacity; https://www.udacity.com/course/robotics-software-engineer--nd209;**

**FF:654**

# **ETD011: Advanced Robotics: Humanoid Design, Nano-Scale Control, AI Integration, and Ethical Governance**

Teaching Scheme:

Theory: 3 Hours / Week ; Laboratory: 2 Hours / Week;

Total Credits: 4

**Syllabus**

**Theory**

**Unit 1: Humanoid Robotics Design and Cognitive Integration-**

Architectural synthesis and structural design of humanoid robotic systems; Multijoint coordination and kinematic sequencing; Dynamic stabilization techniques for bipedal locomotion and equilibrium maintenance; Algorithmic frameworks for gait cycle generation; Kinematic modeling of hand gestures and facial expression articulation; Multimodal integration of speech recognition and computer vision systems; Human-Robot Interaction (HRI) strategies and behavioral modeling; Emotion recognition algorithms and affective computing; Development of socially assistive robotic systems; Technological foundations of rehabilitation robotics and therapeutic interfaces.

**Unit 2: Micro-Scale and Nano-Scale Robotic Engineering -**

Conceptual and structural overview of micro and nano-scale robotic entities; Application of Micro-Electro-Mechanical Systems (MEMS)-based sensor-actuator arrays; Fabrication methodologies for miniature robotic platforms; Energy efficiency and power constraint management at micro-scales; Nano-scale actuation techniques employing electrostatic, magnetic, and piezoelectric mechanisms; Precision control strategies for micro-robotic locomotion; Navigation algorithms within confined micro-environmental domains; Targeted biomedical deployment scenarios; Magnetically and optically mediated control systems; Technological and operational challenges unique to nano-robotic architectures.

**Unit 3: Image Processing and Simultaneous Localization and Mapping (SLAM)-**

Algorithmic foundations of Simultaneous Localization and Mapping (SLAM) in autonomous navigation systems; Visual SLAM methodologies leveraging monocular, stereo, and RGB-D data streams; ORB-SLAM and Real-Time Appearance-Based Mapping (RTAB-Map) algorithmic frameworks; Feature extraction and multiframe feature tracking methodologies; Optimization strategies for map consistency and error minimization; Multisensor data fusion techniques integrating IMU, LiDAR, and vision sensors for SLAM enhancement; Navigation algorithmics across indoor and outdoor operational domains; Loop closure detection and drift correction mechanisms; Robotic Operating System (ROS)-based SLAM pipeline implementation; Case studies demonstrating real-world SLAM deployments.

**Unit 4: Industrial and Healthcare-Centric Robotic Applications -**

Robotic automation in industrial manufacturing environments; Kinematic control and programming of industrial robotic manipulators; Automated Guided Vehicles (AGVs) and Autonomous Mobile Robots (AMRs) within logistics and warehousing; Deployment of surgical robotics in minimally invasive procedures; Telemedicine, telerobotics, and remote medical intervention systems; Agri-robotics for precision farming and yield optimization; Service robotics for hospitality and customer engagement; Robotics in mining, planetary exploration, and space operations; Search-and-rescue robots for disaster response scenarios; Technological evolution and emerging paradigms in the global robotics industry.

**Unit 5: Ethical, Regulatory, and Standardization Frameworks in Robotics - ( 6 Hours)**

Philosophical and operational ethical frameworks for autonomous systems; Privacy violations and surveillance risk management in robotic deployments; Strategic and ethical debates on military robotics and autonomous weaponry; Socioeconomic implications of robotic automation: Job displacement versus productivity enhancement; Global regulatory standards including ISO and IEEE robotic safety directives; Legal frameworks and safety compliance protocols; Human-in-the-loop control systems for ethical decision oversight; Certification, validation, and performance testing methodologies; Policy formulation trends for autonomous system governance; International discourse on future robotic standards and legislations.

**Unit 6: Capstone Project Formulation and Execution Planning- ( 6 Hours)**

Definitional scoping and objective setting for capstone project execution; Academic research methodologies for literature survey and gap identification; Structured proposal development with problem definition; Resource budgeting and project timeline planning; Comprehensive risk identification and mitigation strategies; Mentor and domain expert selection protocols; Deployment of engineering design tools and development frameworks; Implementation of version control systems for collaborative development; Formal documentation of system design and specifications; Iterative review processes and critical feedback incorporation.

**Laboratory**

**List of Experiments**

1. Gait Generation for a Bipedal Robot using Simulated Environment (e.g., Gazebo/ROS)
2. Design and Simulation of MEMS-Based Micro Robot with Actuation Control
3. Real-Time Object Tracking and Facial Gesture Recognition using OpenCV and DLib
4. Visual SLAM Implementation using ORB-SLAM or RTAB-Map in ROS
5. Mobile Robot Navigation with Sensor Fusion (IMU + Lidar)
6. ROS-Based Control of Robotic Arm for Pick-and-Place in Simulated Industry Setting
7. Simulated Tele-Surgical Robot with Delay Modeling and Camera Feedback
8. Case Study and Simulation: Search and Rescue Robot Navigation in Post-Disaster Zone
9. Ethical Decision-Making Scenario Simulation in Autonomous Robots
10. Capstone Proposal Presentation and Version Control Setup using Git/GitHub

**Course Outcomes**

1. CO1: Students will be able to design and implement IoT architectures for robotic systems using protocols like MQTT and interfaces like Blynk/Node-RED to achieve real-time remote control and cloud communication.
2. CO2: Students will be able to apply supervised and unsupervised machine learning techniques, including classification and regression models, to enable real-time decision-making in robotic systems.
3. CO3: Students will be able to analyze and implement control strategies for mobile robots and multi-agent systems, including SLAM, swarm intelligence, and bio-inspired algorithms for collaborative tasks.
4. CO4: Students will be able to develop and deploy robotic applications using ROS 1 and ROS 2 frameworks, incorporating topics, nodes, services, launch files, and hardware integration for real-world tasks.
5. CO5: Students will be able to simulate, implement, and evaluate robotic systems incorporating humanoid gait generation, SLAM, micro-robot actuation, and human-robot interaction techniques.
6. CO6: Students will be able to develop and present integrated robotic projects using ROS, SLAM, and ethical frameworks, demonstrating competence in real-world applications, version control, and proposal development.

**Books and E-Resources**

**Textbooks:**

1. B. Siciliano, O. Khatib; Springer Handbook of Robotics; 2nd Ed.; Springer; 2016; DOI: 10.1007/978-3-319-32552-1
2. K. M. Lynch, F. C. Park; Modern Robotics: Mechanics, Planning, and Control; 1st Ed.; Cambridge University Press; 2017;
3. **S. Thrun, W. Burgard, D. Fox;** Probabilistic Robotics; 1st Ed.; MIT Press; 2005;
4. Dehon, M. A. Favorov, Y. Bar-Cohen; Micro- and Nano-Robotics: Technologies and Applications; 1st Ed.; Springer; 2010;

**References:**

1. J. Chestnutt, M. Lau, G. Nishiwaki, J. Kuffner; Humanoid Locomotion Planning for Dynamic Environments; Springer Tracts in Advanced Robotics; 2007;DOI: 10.1007/978-3-540-36124-1
2. G. Bekey; Autonomous Robots: From Biological Inspiration to Implementation and Control; 1st Ed.; MIT Press; 2005;
3. P. Corke; Robotics, Vision and Control: Fundamental Algorithms in MATLAB; 2nd Ed.; Springer; 2017;
4. M. J. Mataric; The Robotics Primer; 1st Ed.; MIT Press; 2007;

**MOOCs and Other Learning Resources:**

1. **D. Scaramuzza;** Visual SLAM for Autonomous Vehicles; edX (ETH Zurich);
**URL:** https://www.edx.org/course/visual-perception-for-autonomous-driving
2. **S. Thrun;** Programming a Robotic Car; Udacity;
**URL:** https://www.udacity.com/course/programming-a-robotic-car--cs373
3. **Saxena, P. Abbeel;** Artificial Intelligence for Robotics; Coursera;
**URL:** https://www.coursera.org/learn/ai-for-robotics

**FF: 654**

# **ETD012: Capstone Robotics Project**

**Teaching Scheme:**

**Theory:** Hours / Week ; Laboratory: 8 Hours / Week;

**Total Credits:** 4

Contact Hours: Independent guided work (with regular mentor reviews and presentations)

Evaluation: Project proposal (10%), mid-evaluation (20%), final presentation + demo (40%), report (20%), viva (10%)

**Syllabus**

**Objective**

To enable students to integrate multi-disciplinary concepts in robotics, electronics, AI/ML, IoT, and computer vision to design and implement a full-fledged intelligent robotic system that addresses a real-world challenge or research problem.

Suggested Project Areas (based on prior semester topics)

1. AI in Robotics

* Intelligent robot assistants using NLP and computer vision
* Predictive maintenance bots using ML

2. IoT-Enabled Robotics

* Remote robotic control and monitoring systems for smart homes or industries
* IoT-enabled robotic greenhouse or smart surveillance system

3. Swarm & Convoy Robotics

* Multi-robot coordination for search & rescue missions
* Convoy of autonomous delivery robots

4. Humanoid & Social Robotics

* Assistive humanoid robots for elderly care or education
* Emotion-aware robotic interface using ML

5. Micro & Mobile Robotics

* Medical micro-bot navigation simulation
* Indoor mobile delivery bot with object detection

6. Vision-Based Path Planning and Control

* Drone path planning using deep reinforcement learning
* Obstacle-aware vision navigation robot with SLAM

7. ROS 1/2 Integration Projects

* Full ROS-based control system for warehouse robots
* ROS2 navigation stack-based mapping robot for autonomous ground vehicles

**Expected Deliverables**

1. Project Proposal Document (Problem statement, objectives, methodology, timeline)
2. Literature Survey (10–15 key references)
3. System Design and Implementation (Hardware & software architecture, modules)
4. Mid-Term Demonstration
5. Final Live Demo / Video Submission
6. Project Report (including test results and future scope)
7. Code Repository & Documentation (on GitHub/GitLab or equivalent)

**Recommended Tools and Platforms**

1. Simulation: Gazebo, Webots, RViz, V-REP
2. Programming: Python, ROS/ROS2, Arduino IDE, TensorFlow, OpenCV
3. Hardware: Raspberry Pi, NVIDIA Jetson, ESP32, Arduino, sensors, actuators, camera modules
4. AI/ML: Scikit-learn, Keras, PyTorch
5. IoT: MQTT, Blynk, Node-RED, ThingSpeak

**Course Outcomes**

1. Develop intelligent robotic systems by integrating natural language processing and machine learning for real-time interaction and predictive functionality.
2. Design IoT-based robotic platforms for remote control and monitoring in smart home or industrial environments.
3. Implement coordinated multi-robot systems for collaborative tasks such as search and rescue or autonomous delivery.
4. Design emotion-aware humanoid robots capable of interacting with users in assistive or educational settings.
5. Develop autonomous robots with vision-based path planning and SLAM for real-time obstacle-aware navigation.
6. Implement a complete robotic system using ROS1/ROS2 for autonomous control, mapping, and navigation