COEP Technological University Pune

(A Unitary Public University of Govt. of Maharashtra)

School of Engineering and Technology

Curriculum Structure & Detailed Syllabus

M. Tech.

Electronics- Embedded System and Computing

(Effective from: A.Y. 2026-27)

M. Tech.
Electronics- Embedded System and Computing

List of Abbreviations

Abbreviation	Title	No of courses	Credits	% of Credits
PSMC	Program Specific Mathematics Course	1	4	5 %
PSBC	Program Specific Bridge Course	1	3	3.75 %
PCC	Program Core Course	6	24	30 %
PEC	Program Specific Elective Course	3	9	11.25 %
VSEC	Vocational and Skill Enhancement Course	2	22	27.5 %
OE	Open Elective	1	3	3.75 %
SLC	Self-Learning Course	2	6	7.5 %
MLC	Mandatory Learning Course	2	5	6.25
LLC	Liberal Learning Course	1	1	1.25 %
OJT	Internship	1	3	3.75
	Total	20	80	100%

M. Tech.
Electronics- Embedded System and Computing

Semester -I

Sr. No	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	PSMC	*PSMC-01*	Statistics, Probability, Graph and Field Theory	3	1	-	1	4	30	20	50	50	50
2	PSBC	*PSBC-01*	Advanced Digital Design	2	-	2	1	3	30	20	50	50	50
3	PCC	*PCC-01*	IoT Architecture and Computing	3	-	2	1	4	30	20	50	50	50
4	PCC	*PCC-02*	Processors and Controllers: Architecture and application programming	3	-	2	1	4	30	20	50	50	50
5	PCC	*PCC-03*	Software Tools for Embedded system and Edge computing	3	-	2	1	4	30	20	50	50	50
6	PEC	*PEC-01*	Program Specific Elective –I a). Advanced Digital Signal and Image Processing. b). Hardware and Software Co-Design (Advanced Digital Design) c). IoT sensors-actuators and Communication protocols d). Automotive Embedded Product Development *	3	-	-	1	3	30	20	50	-	-
7	MLC	*MLC-01*	Research Methodology and Intellectual Property Rights	2	1	-	0	3	30	20	50	-	-
			Total	19	**2**	**8**	**7**	25					

Semester -II

Sr. No	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	OE	*OE-01*	Open Elective (Networked Embedded System Design (to be offered to other dept)	3	-	-	1	3	30	20	50	-	-
2	PCC	*PCC-04*	Embedded System Security	3	-	2	1	4	30	20	50	50	50
3	PCC	*PCC-05*	Embedded Operating system	3	-	2	1	4	30	20	50	50	50
4	PCC	*PCC-06*	Data analytics on Edge Computing	3	-	2	1	4	30	20	50	50	50
5	PEC	*PEC-02*	Program Specific Elective-II a). AD-CMOS b). Cloud Computing c). AIML d). Automotive Embedded Hardware Development *	3	-	-	1	3	30	20	50	-	-
6	PEC	*PEC-03*	Program Specific Elective-III a). Advanced VLSI architecture b). SCADA systems Applications c). Wireless Sensor Network d). Automotive Embedded Software Development *	3	-	-	1	3	30	20	50	-	-
7	MLC	*MLC-02*	Technical Communication Skills	1	1	-	-	2	50	50	-	100	-
8	LLC	*LLC-01*	Liberal Learning Course	-	-	2	2	1	-	-	-	100	-

			Total	**19**	**1**	**8**	**8**	**24**					

Semester -III

Sr. No	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	VSEC	*VSEC-01*	Dissertation Phase – I	--	--	22	12	11	-	-	-	60	40
2	SLC	*SLC-01*	Massive Open Online Course –I	3	--	--	1	3	25	-	75	-	-
3	SLC	*SLC-02*	Massive Open Online Course –II	3	--	--	1	3	25	-	75	-	-
4	OJT		Internship	--	--	--	--	3	-	-	-	-	-
			Total	**9**	**-**	**22**	**14**	**20**					

Semester -IV

Sr. No	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	VSEC	*VSEC-02*	Dissertation Phase – II	--	--	22	12	11	-	-	-	60	40
			Total	**--**	**--**	**22**	**12**	**11**					

Semester - I

(PSMC) Statistics, Probability, Graph and Field Theory						
Course Code	PSMC-01		**Examination Scheme**			
Teaching Scheme	3-1-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	ISE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
Understand and apply the linear algebra and statistical techniques. Grasp and apply Graph theory for engineering problem solving and CAD tools developments. Culminate operations in groups, rings and field theory towards applications in digital electronic systems. Characterize random variables and its functions with probability distributions and cumulative distributions.

Graph Theory:	**[10 Hrs]**

Basic concepts of Graph Theory, Digraphs, Paths and Circuits, Reachability and Connectedness, Matrix representation of graphs, Subgraphs & Quotient Graphs, Isomorphic digraphs & Transitive Closure digraph, Euler's Path & Circuit- definitions and examples, Connected graphs and shortest paths: Walks, trails, paths, connected graphs, distance, cut-vertices, cut-edges, blocks, connectivity, weighted graphs, shortest path algorithms, Special classes of graphs: Bipartite graphs, line graphs, Chordal graphs, Planar graphs

Group, Rings & Fields:	**[12 Hrs]**

Set theory Relations and Functions, Semigroups, Monoids, Subsemigroup, Submonoid, Isomorphism & Homomorphism, Fields, Integral Domain, Ring Homomorphism, Ring homomorphisms and their kernels, Ideals in commutative rings, First Isomorphism Theorem for commutative rings; the example of integers mod n, Abelian groups, Quotient Groups, fields of characteristic $p > 0$, algebraic field extension, algebraic closure, perfect field, Galois groups of polynomials, Galois groups over rationals, Field automorphisms. Automorphism group, decomposition of a permutation into a product of disjoint cycles, Homological algebra, polynomials, Algebraic extensions, Galois extensions, Galois group of a polynomial. Galois correspondence, Calculating Galois groups. Cyclotomic extensions, Applications to Cryptography, Basic definitions of cellular automata and symbolic dynamics

Probability and Statistics:	**[14 Hrs]**

Definitions, conditional probability, Bayes Theorem and independence. Random Variables: Discrete, continuous and mixed random variables, probability mass, probability density and cumulative distribution functions, mathematical expectation, moments, moment generating function, Chebyshev inequality. Special Distributions: Discrete uniform, Binomial, Geometric, Poisson, Exponential, Gamma, Normal distributions. Pseudo random sequence generation with given distribution, Functions of a Random Variable. Joint Distributions: Joint, marginal and conditional distributions, product moments, correlation, independence of random variables, bi-variate normal distribution. Stochastic Processes: Definition and classification of stochastic processes, Poisson process. Norms, Statistical methods for ranking data

Textbooks & Reference books:	
[1]	Kolman, Bernard, Robert C. Busby, and Sharon Cutler Ross, "Discrete mathematical structure" Prentice-Hall, Inc., 2003.
[2]	Dummit, David Steven, and Richard M. Foote, "Abstract algebra" Vol. 3. Hoboken: Wiley, 2004.
[3]	McIntosh, Harold V, "One dimensional Cellular Automata", Luniver Press, 2009.
[4]	Steeb, Willi-Hans, "The nonlinear workbook: Chaos, fractals, cellular automata, genetic algorithms, gene expression programming, support vector machine, wavelets, hidden Markov models, fuzzy logic with C++", World Scientific Publishing Company, 2014.
[5]	Bondy, John Adrian, and Uppaluri Siva Ramachandra Murty, "Graph theory with applications", Vol. 290. London: Macmillan, 1976.

(PSBC) Advanced Digital Design						
Course Code	PSBC-01		**Examination Scheme**			
Teaching Scheme	2-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	ISE: 50	ESE: 50	--

Course Outcomes: Students will be able to:

1. Apply principles of digital logic, Boolean algebra, and simplification techniques to design and analyze combinational circuits, considering hazards, glitches, and low-power constraints.
2. Analyze and design sequential logic circuits including flip-flops, counters, and shift registers, while evaluating timing parameters such as setup time, hold time, and clock-related issues.
3. Develop and simulate digital designs using Verilog HDL, including structural and behavioral modeling, and verify functionality using testbenches.
4. Design complex digital systems using procedural constructs and finite state machines, and apply design reuse methodologies for scalable and optimized implementations.
5. Implement and validate digital systems on FPGA/CPLD platforms, perform synthesis and timing analysis, and integrate system-level components for embedded applications.

Digital Logic Foundations & Combinational Design	**[3 Hrs]**
Review of number systems, code converters, Boolean algebra, and simplification methods (K-map for 2–4 variables). Combinational logic circuits: adders, multiplexers, decoders, encoders, comparators. Low-power combinational design techniques. Hazards, glitches, and metastability in embedded circuits. Design basic combinational circuits (adders, multiplexers).	
Sequential Logic Design & Timing	**[3 Hrs]**
Latches and flip-flops (SR, D, JK, T). Counters and shift registers. Setup/Hold time, clock skew, and metastability solutions (synchronizers). Low-power sequential design considerations. Design counters and shift registers; measure timing parameters.	
Verilog HDL Fundamentals	**[4 Hrs]**
Datatypes (nets, registers, bit vectors, arrays, parameters). Modules: ports, hierarchical naming. Continuous assignments and basic gates. Testbenches, debugging, and simulation basics. Write and simulate Verilog modules for arithmetic and logic circuits.	
Behavioral & Procedural Design	**[4 Hrs]**
The always block: blocking vs. non-blocking assignments. Functions, tasks, and control constructs for synthesis. Design examples: counters, unsigned multipliers, ALUs. Validation: verification vectors, post-synthesis verification. Design a parameterized ALU; developing a testbench for verification.	
Finite State Machines (FSM) & Design Reuse	**[4 Hrs]**

FSM design methodologies (Mealy and Moore). Explicit and implicit FSM specification styles. Example: Booth multiplier, FIFO buffer design. Design reuse: parameterized modules, vendor IP, and control-point style. Designing of FSM, State reduction methods, sequence detector.	
Synthesis & Implementation on FPGA/CPLD	**[3 Hrs]**
Synthesis flow: from RTL to standard cells/FPGA. Resource optimization: timing, area, and power. Advanced memory concepts: basic SRAM/Flash interfacing for embedded systems. Introduction to bus interfacing protocols (SPI, I^2C, UART). Synthesize a Verilog design on FPGA; analyze timing report.	
System-Level Integration & Mini-Project	**[3 Hrs]**
Focus on low-power design techniques (clock gating, power gating). Design for testability (DFT) basics. Introduction to SoC design concepts and integration. Mini-project: End-to-end design of a controller-based module (e.g., PWM generator, communication module, or embedded datapath). Project: Final FPGA implementation and demonstration.	
Textbooks & Reference books:	

[1]	Stephen Brown and Zvonko Vranesic, "Fundamentals of Digital logic with Verilog Design", Mc-GrawHill, 3rd edition.
[2]	Michael D. Ciletti: Advanced Digital Design with Verilog HDL, PHI, 2013.
[3]	Peter J. Ashenden: Digital Design – An Embedded Systems Approach Using Verilog, Elsevier, 2013.
[4]	Verilog HDL-A Guide to Digital Design and Synthesis, Samir Palnitkar.

(LC) Advanced Digital Design lab

Course Code			**Examination Scheme**			
Teaching Scheme	0-0-2-0		**Theory**	TA: --	MSE: --	ESE: --
Credits	1		**Laboratory**	CIE: 50	ESE: 50	--

Course Outcomes:

1. Design, simulate, and verify combinational and sequential digital circuits using Verilog HDL in the Xilinx Design Suite environment.
2. Develop hierarchical and modular digital system designs using Verilog constructs such as conditional statements, loops, functions, and module instantiation for FPGA-based applications.

List of Experiments:

1**).** Installation and Environment Setup of Xilinx Design Suite for Digital Circuit Implementation.
2). Implementation and Verification of Basic Logic Gates using Verilog.
3). Design and Simulation of Half Adder and Full Adder Circuits using Verilog.
4). Hierarchical Design Using Module Instantiation Techniques in Xilinx *(Implementation of Full Adder Using Half Adder / Multi-Input Logic Using Two-Input Gates)*.
5). Design of Digital Circuits Using Conditional Statements and Looping Constructs in Xilinx.
6). Design and Implementation of a Digital Calculator using Verilog.
7). Design, Simulation, and Analysis of Flip-Flops (JK, T, D) and SR Latch Using Xilinx.
8). Design an Application of User-Defined Functions in Xilinx for Digital Systems.
9). Design, Simulation, and Verification of Counters using Verilog.

(PCC) IoT Architecture and Computing						
Course Code	PCC-01		**Examination Scheme**			
Teaching Scheme	3-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	ISE: 50	ESE: 50	--

Course Outcomes: Students will be able to:

1. To understand basic concepts and communication protocols of IoT.
2. To understand types of technologies that can be utilized to implement IoT solutions.
3. Develop IoT solutions based on popular embedded platforms to address real life problems.

IoT Overview	**[5 Hrs]**
Internet of Things Framework, Architecture, Technology, Machine 2 Machine Communication (M2M)	
Sensors, Actuators, Embedded Platforms	**[8 Hrs]**
Sensors, Actuators, Embedded platforms, Wireless Sensor Networks, RFID	
Embedded Devices & Communication Protocols	**[8 Hrs]**
Embedded Devices: Platforms, computing, Operating Systems-TinyOS, Contiki OS. Communication Protocols: IPV6, 6LOWPAN, MQTT, COAP, AMPQ, Zigbee Smart energy 2.0, LoRa WAN	
Cloud, Fog Computing & Emerging Technologies	**[8 Hrs]**
Cloud Computing, Fog computing, Big Data, NoSQL Database, Security and Privacy issues in IoT. IoT and Artificial Intelligence, Machine Learning, Blockchain, 5G Technology IoT	
Case Studies	**[5 Hrs]**
Smart grid, Smart agriculture, Smart healthcare, Smart transportation, Smart City challenges	
Textbooks & Reference books:	

[1]	A Bahaga, V. Madisetti, "Internet of Things- Hands on approach", 1st Edition, VPT publisher, 2014.
[2]	Raj Kamal, "Internet of Things-Architecture and Design Principles", McGraw Hill.
[3]	Buyya, Rajkumar, and Amir Vahid Dastjerdi, eds. "Internet of Things: Principles and paradigms", Elsevier, 2016.

[4]	Hassan, Qusay F., ed. "Internet of things A to Z: technologies and applications", John Wiley & Sons, 2018.
[5]	Samuel Greenguard, "Internet of things", MIT Press.

(LC) IoT Architecture and Computing Lab						
Course Code	LC-01		**Examination Scheme**			
Teaching Scheme	0-0-2-0		**Theory**	TA: --	MSE: --	ESE: --
Credits	1		**Laboratory**	CIE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
Develop programming ability using C/C++/VB/Python. Explore the interconnection and integration of the real time parameters and the smart application.

List of experiments:

1) To study Arduino IDE and different types of Arduino Board.
2) Write a program to calculate the distance using ultrasonic sensor.
3) Write a program to measure room-temperature, humidity and heat index.
4) To determine the gas level/smoke using MQ2 Gas sensor.
5) Write a program to detect motion using PIR sensor.
6) To detect the presence with the RFID Tag and Reader.
7) Cloud Computing, Fog computing, Big Data, NoSQL Database, Security and Privacy issues in IoT, IoT and Artificial Intelligence, Machine Learning, Blockchain, 5G Technology IoT.
8) To study GSM/GPRS Module interfacing with Arduino.
9) To study Thingspeak cloud.
10) To design a Graphical user interface using VB/Python.

(PCC) Processors and Controllers: Architecture and Application Programming						
Course Code	PCC-02		**Examination Scheme**			
Teaching Scheme	3-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	ISE: 50	--	ESE: 50

Course Outcomes: Students will be able to:

1. Employ ARM Cortex Mx processor core architectural features and its programmer's model in developing applications.
2. Design and Develop code having sections with different processing priority using nested vectored interrupt controller features supported by ARM Cortex Mx processor core.
3. Apply the signal processing capability native to ARM Cortex Mx processor in developing small applications.
4. Identify and utilize on-chip/onboard peripherals on microcontroller evaluation board and their corresponding memory mapped addresses.

5. Knowhow of advanced edge computing and security features supported by recently introduced ARM Cortex Mx processors.	
ARM Cortex-M3/M4 Processor	**[6 Hrs]**
Applications, Programming model - Registers, Operation modes, Exceptions and Interrupts, Reset Sequence, Introduction to the Tiva microcontrollers, Tiva block diagram, System Clocks and Control, Hibernation Module on Tiva	
Overview of Instruction Set and Memory System	**[6 Hrs]**
Unified Assembler Language, Memory Maps, Memory Access Attributes, Permissions, BitBand Operations, Unaligned and Exclusive Transfers. Cortex-M4 specific instructions, Barrel shifter, Pipeline, Bus Interfaces	
Nested Vectored Interrupt Controller	**[7 Hrs]**
Exceptions, Types, Priority, Vector Tables, Interrupt Inputs and Pending behavior, Fault Exceptions, Supervisor and Pendable Service Call, Nested Vectored Interrupt Controller, Basic Configuration, SYSTICK Timer, Interrupt Sequences, Exits, Tail Chaining, Interrupt Latency	
Floating Point Unit and DSP Support	**[6 Hrs]**
Floating point operations, Floating point unit (FPU), Registers, Lazy stacking, DSP on a Micro-controller, Architecture of traditional DSP processor, DSP instructions – fractional arithmetic, SIMD data, load store instructions, arithmetic instructions, optimization strategies, DSP applications	
TM4C123x Micro-controller	**[6 Hrs]**
Internal memory, GPIOs, Timers, ADC, UART and other serial interfaces, PWM, RTC, WDT	
Introduction to ARM Cortex M-55/85 Processor	**[7 Hrs]**
Architectural features supporting enhanced, energy-efficient digital signal processing (DSP) and machine learning (ML) performance, Enhanced software security with TrustZone and PACBTI extension.	
Textbooks & Reference books:	

[1]	Joseph Yiu, "The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors", Elsevier, 3rd Edition, 2013.
[2]	Andrew Sloss, Dominic Symes, Chris Wright, "ARM System Developer's Guide", Elsevier.

(LC) Processors and Controllers Programming Lab						
Course Code	LC-02		**Examination Scheme**			
Teaching Scheme	0-0-2-0		**Theory**	TA: --	MSE: --	ESE: --
Credits	1		**Laboratory**	CIE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
Develop programming ability using C/C++/VB/Python.

List of experiments:

1) Blink an LED with software delay, delay generated using the SysTick timer.
2) System clock real time alteration using the PLL modules.
3) Control intensity of an LED using PWM implemented in software and hardware.
4) Control an LED using switch by polling method, by interrupt method and flash the LED once every five switch presses.
5) UART Echo Test.
6) Take analog readings on rotation of rotary potentiometer connected to an ADC channel.
7) Program - Read, Write and Verify - serial EEPROM using I2C interface.
8) Calling C function from assembly program and vice versa.
9) Evaluate the various sleep modes by putting core in sleep and deep sleep modes.
10) System reset using watchdog timer in case something goes wrong.
11) To develop an assembly code and C code to compute Euclidian distance between any two points.
12) To develop assembly and C code for implementation of convolution operation.
13) To design and implement filters in C to enhance the features of given input sequence/signal.

(PCC) Software Tools for Embedded System and Edge Computing						
Course Code	PCC-03		**Examination Scheme**			
Teaching Scheme	3-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	ISE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
1. Write an embedded C application of moderate complexity.
2. Develop and analyze algorithms in C++.
3. Differentiate interpreted languages from compiled languages.
4. Acquire programming skills in core Python.
5. Develop applications using scripting languages such as Python.

Embedded C Programming: **[10 Hrs]**

Bitwise operations, Dynamic memory allocation, OS services. Linked stack and queue, Sparse matrices, Binary tree. Interrupt handling in C, Code optimization issues. Writing

LCD drives, LED drivers, Drivers for serial port communication. Embedded Software Development Cycle and Methods (Waterfall, Agile). Assignments: Develop a simple embedded application using C to be deployed on any hardware platform. Write an LCD/LED/Serial port device driver in C.	
Object Oriented Programming with Embedded C++:	**[12 Hrs]**
Differences between C and C++, Fundamentals of object oriented programming; OOP vs. Procedure oriented programming. OOP concepts: classes, objects, abstraction, polymorphism, inheritance, data binding and encapsulation. Basics of C++: features of C++, data types, standard I/O, arrays and strings in C++. Classes in C++, instantiation, creating objects and object scope, data abstraction, data encapsulation, constructors and destructors, methods and access modifiers, function and operator overloading. Inheritance-Base and Derived classes, Inheritance types, Scope Resolution operator; polymorphism and virtual functions, exception handling. Assignments: Implement a C++ program which reverses the words in a given sentence. Write a complex application in C++ (e.g. CRC implementation) using Object Orientated design principles.	
Scripting Languages:	**[14 Hrs]**
Overview of Scripting Languages – Python, Overview, History and Features. Python: Variables, Operators, Decision Making, Loops, Strings, Lists, Tuples, Functions, Modules, Packages, Exceptions, Classes/Objects, Regular Expressions, CGI Programming, Database Access, Networking, Sending Email, Multithreading, XML Processing, GUI programming. Assignments: Write a program for Encryption and Decryption of messages in Python. Write a Python program to read, write and append contents to the text and binary file. Write a simple Python script that serves a simple HTTP Response and a simple HTML Page. Develop GUI for an Expression Calculator using 'tk'.	
Textbooks & Reference books:	

[1]	Michael J. Pont, "Embedded C", Pearson Education, 2nd Edition, 2008.
[2]	Martin C. Brown, "Python: The Complete Reference", McGraw-Hill Education, 2001.
[3]	A. Michael Berman, "Data structures via C++", Oxford University Press, 2002.
[4]	Robert Sedgewick, "Algorithms in C++", Addison Wesley Publishing Company, 1999.
[5]	Abraham Silberschatz, Peter B, Greg Gagne, "Operating System Concepts", John Willey & Sons, 2005.
[6]	E. Balaguruswami, "Object-Oriented Programming With C++", McGraw-Hill Education, 7th Edition, 2017.

(PEC) IoT Sensors, Actuators and Communication Protocols							
Course Code	PEC-01a		**Examination Scheme**				
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50	
Credits	3		**Laboratory**	--	--	--	

Course Outcomes: Students will be able to:
1. To understand the working principles and types of IoT sensors and actuators.
2. To study the interfacing of sensors and actuators with microcontrollers.
3. To explore different communication technologies used in IoT.
4. To compare and implement various IoT communication protocols.
5. To analyze the constraints and design challenges in IoT environments.

Introduction to IoT and Sensors **[6 Hrs]**

Introduction to IoT: Definition, architecture, applications, IoT device and sensor node architecture, Role of sensors and actuators in IoT, Classification of sensors: physical, chemical, biological, etc. Characteristics of sensors: sensitivity, resolution, accuracy, precision, etc.

Sensor and Actuator Technologies **[8 Hrs]**

Commonly used sensors in IoT: Temperature (DHT11/22), Motion (PIR), Gas (MQ series), Proximity (IR/Ultrasonic), Light (LDR), etc. Actuators: Relays, Motors (DC, Stepper, Servo), Solenoids. Interfacing sensors/actuators with microcontrollers (Arduino, ESP32, STM32). Signal conditioning: filtering, amplification, ADC, DAC.

IoT Communication Fundamentals **[8 Hrs]**

Communication requirements in IoT: low power, low latency, scalability. OSI & TCP/IP model overview relevant to IoT. Physical and MAC layers in IoT context. Serial communication: UART, SPI, I2C. Wireless communication basics: RF, modulation, bandwidth, range.

Wireless Communication Protocols **[6 Hrs]**

IEEE 802.15.4, ZigBee, Bluetooth and BLE, LoRa, LoRaWAN, Wi-Fi (802.11), Wi-Fi HaLow, RFID, NFC, Z-Wave, Sigfox. Comparison based on range, power, throughput, topology.

Network and Application Layer Protocols **[8 Hrs]**

IPv4, IPv6, 6LoWPAN, MQTT, CoAP, AMQP, HTTP/HTTPS, WebSockets, RESTful APIs in IoT. Message queuing and data serialization (JSON, XML).

Textbooks & Reference books:

[1] "Internet of Things: A Hands-On Approach" – Arshdeep Bahga and Vijay Madisetti.

[2]	"Getting Started with the Internet of Things" – Cuno Pfister.
[3]	"Protocols and Architectures for Wireless Sensor Networks" – Holger Karl & Andreas Willig.
[4]	Research papers from IEEE IoT Journal, ACM IoT Transactions.

(PEC) Advanced Digital Signal and Image Processing

Course Code	PEC-01b		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:
1. Understand and apply knowledge of various transforms and probability theory in signal and image processing.
2. Perform fundamental operations in digital image processing like enhancement, encoding, feature extraction, and segmentation.
3. Analyze, apply and critically evaluate various signal/image processing algorithms appropriate for practical applications.

Review of Discrete Time Signals and Systems:	**[4 Hrs]**
Characterization in time and Z and Fourier domain, Fast Fourier Transform algorithms – In-place computations, Butterfly computations, bit reversals.	
Digital Filter Design:	**[4 Hrs]**
FIR and IIR filters – Impulse invariance, bilinear transformation.	
Representation of Signal Processing Algorithms:	**[6 Hrs]**
Signal flow, Data flow and Dependence Graphs. Iteration Bound, algorithms for computing iteration bound, Pipelining, Parallel processing.	
Introduction to Image Processing:	**[6 Hrs]**
Applications and fields of image processing, Fundamental steps in Digital image processing, Elements of visual perception, Image sensing and acquisition, Basic Concepts in Sampling and Quantization, representing digital images.	
Image Enhancement:	**[5 Hrs]**
Some basic gray level transformations, Histogram Processing, Sharpening Spatial filters, Image Enhancement in the spatial and Frequency domain, Pseudo-colouring.	

Segmentation:	**[5 Hrs]**
Some Basic Relationships between pixels, point, Edge based segmentation, Boundary detection, extraction and representation, Threshold based segmentation, Region based segmentation, Texture based segmentation. Morphological operations.	
Image Compression:	**[5 Hrs]**
Data redundancies, Variable-length coding, Predictive coding, Transform coding, Image compression standards.	
Case Studies:	**[4 Hrs]**
VLSI architectures for implementation of Image Processing algorithms.	
Textbooks & References:	

[1]	J.G. Proakis, Manolakis, "Digital Signal Processing", Pearson, 4th Edition.
[2]	Rafael C. Gonzalez, Richard E. Woods, "Digital Image Processing", Pearson, 4th Edition.
[3]	Keshab Parhi, "VLSI Digital Signal Processing Systems – Design and Implementation", Wiley India.

(PEC) Hardware and Software Co-Design(Advanced Digital Design)						
Course Code	PEC-01c		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand hardware-software co-design principles.
2. Analyze system partitioning strategies.
3. Design embedded systems using co-design methodologies.
4. Evaluate performance trade-offs.
5. Implement systems using HDL and embedded software frameworks.

Introduction to Hardware-Software Co-Design	**[6 Hrs]**
Embedded system design flow, hardware-software partitioning, abstraction levels, system specification, modeling techniques, performance metrics: latency, throughput, power, co-design methodologies, design challenges.	

System Modeling and Design Methodologies	**[7 Hrs]**
Behavioral and structural modelling, finite state machines, dataflow modelling, high-level synthesis, system-level design languages, transaction-level modelling, UML for embedded systems, design space exploration.	
Hardware Design and Implementation	**[6 Hrs]**
RTL design using Verilog/System Verilog, synthesis and timing constraints, FPGA-based design, hardware accelerators, interfacing with peripherals, optimization techniques, pipelining and parallelism.	
Software Design and Integration	**[8 Hrs]**
Embedded software development, real-time operating systems, task scheduling, device drivers, communication interfaces, hardware abstraction layers, co-simulation techniques, debugging and profiling.	
Co-Design Platforms and Applications	**[6 Hrs]**
FPGA-SoC platforms, ARM-based systems, hardware-software integration, performance optimization, verification and validation, case studies in multimedia, IoT, and signal processing systems.	
Self-Study	**[3 Hrs]**
High-level synthesis tools, system-level design using MATLAB/Simulink, emerging trends in heterogeneous computing and edge systems.	
Textbooks & Reference books:	

[1]	Jorgen Staunstrup, Wayne Wolf, *Hardware/Software Co-Design: Principles and Practice*, Springer.
[2]	Patrick R. Schaumont, *A Practical Introduction to Hardware/Software Co-Design*, Springer.
[3]	Frank Vahid, Tony Givargis, *Embedded System Design: A Unified Hardware/Software Approach*, Wiley.
[4]	Wayne Wolf, *Computers as Components*, Morgan Kaufmann.
[5]	Daniel D. Gajski et al., *Specification and Design of Embedded Systems*, Prentice Hall

(PEC) Automotive Embedded Product Development						
Course Code	PEC-01d		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:
1. Understand automotive product development lifecycle.
2. Analyze system requirements and architectures.
3. Design embedded automotive products.
4. Evaluate safety and compliance.
5. Apply validation and deployment strategies.

Automotive Product Development Process	**[6 Hrs]**
Automotive industry overview; product lifecycle; V-model development process; requirement engineering; system specification; design documentation; standards and compliance.	
System Architecture and Design	**[7 Hrs]**
Automotive system architecture; ECU design; hardware-software integration; communication networks; CAN, LIN, FlexRay, Ethernet; system partitioning; design trade-offs.	
Development Frameworks and Implementation	**[6 Hrs]**
AUTOSAR architecture; software components; runtime environment; embedded software development; model-based design using MATLAB/Simulink; code generation; integration techniques.	
Functional Safety and Quality Assurance	**[8 Hrs]**
ISO 26262 standard; hazard analysis; ASIL classification; safety lifecycle; fault detection and diagnostics; reliability engineering; quality standards; compliance verification.	
Testing, Validation and Deployment	**[6 Hrs]**
Verification and validation; Software-in-the-loop (SIL); Hardware-in-the-loop (HIL); system testing; calibration; production and deployment; over-the-air updates; automotive cybersecurity basics.	
Case Study	**[3 Hrs]**
Case studies of automotive product development; electric and autonomous vehicle platforms; software-defined vehicles; future mobility systems.	
Textbooks & References:	

[1]	Nicolas Navet, Francoise Simonot-Lion, *Automotive Embedded Systems Handbook*, CRC Press.
[2]	Ronald K. Jurgen, *Automotive Electronics Handbook*, McGraw-Hill.
[3]	Robert Bosch GmbH, *Automotive Handbook*, Wiley.
[4]	AUTOSAR Consortium, *AUTOSAR Specifications*.
[5]	ISO 26262 Standard Documentation.
[6]	H. Kopetz, *Real-Time Systems*, Springer.

(MLC) Research Methodology and Intellectual Property Rights

Course Code	MLC-01		Examination Scheme			
Teaching Scheme	2-1-0-0		**Continuous**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:
Understand research problem formulation and approaches of investigation of solutions for research problems. Learn ethical practices to be followed in research and apply research methodology in case studies and acquire skills required for presentation of research outcomes. Discover how IPR is regarded as a source of national wealth and mark of an economic leadership in context of global market scenario. Summarize that it is an incentive for further research work and investment in R & D, leading to creation of new and better products and generation of economic and social benefits.

Unit 1: **[6 Hrs]**

Meaning of research problem, Sources of research problem, Criteria, Characteristics of a good research problem, Errors in selecting a research problem, Scope and objectives of research problem. Approaches of investigation of solutions for research problem, data collection, analysis, interpretation, necessary instrumentations.

Unit 2: **[6 Hrs]**

Effective literature studies approaches, analysis. Use Design of Experiments/Taguchi Method to plan a set of experiments or simulations or build prototype. Analyze your results and draw conclusions or Build Prototype, Test and Redesign.

Unit 3: **[6 Hrs]**

Plagiarism, Research ethics, Effective technical writing, how to write report, Paper. Developing a Research Proposal, Format of research proposal, a presentation and assessment by a review committee.

Unit 4: **[6 Hrs]**

Introduction to the concepts Property and Intellectual Property, Nature and Importance of Intellectual Property Rights, Objectives and Importance of understanding Intellectual Property Rights.

Unit 5: **[6 Hrs]**

Understanding the types of Intellectual Property Rights: Patents – Indian Patent Office and its Administration, Administration of Patent System, Patenting under Indian Patent Act, Patent Rights and its Scope, Licensing and transfer of technology, Patent information and database. Provisional and Non Provisional Patent Application and Specification, Plant Patenting, Idea Patenting. Integrated Circuits, Industrial Designs, Trademarks (Registered and unregistered trademarks), Copyrights, Traditional Knowledge, Geographical Indications, Trade Secrets, Case Studies.

Unit 6:		[6 Hrs]
New Developments in IPR, Process of Patenting and Development: technological research, innovation, patenting, development. International Scenario: WIPO, TRIPs, Patenting under PCT.		
Textbooks & References:		
[1]	Aswani Kumar Bansal, "Law of Trademarks in India".	
[2]	B L Wadehra, "Law Relating to Patents, Trademarks, Copyright, Designs and Geographical Indications".	
[3]	G.V.G Krishnamurthy, "The Law of Trademarks, Copyright, Patents and Design".	
[4]	Satyawrat Ponkse, "The Management of Intellectual Property".	
[5]	T. Ramappa, S. Chand, "Intellectual Property Rights under WTO".	
[6]	WIPO: "WIPO Guide To Using Patent Information".	

Semester - II

(OE) [OE-01] Networked Embedded System Design						
Course Code			**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Implement simple sketches on the Arduino boards involving several peripherals
2. Identify, design and implement applications on the Arduino boards producing custom shields.
3. Deploy low-end applications using low- and high-level languages on microcontroller platform.

Introduction to processors:	**[6 Hrs]**
Introduction of Microprocessors and Microcontrollers, Introduction of Arduino Microcontrollers.	
Introduction to architecture:	**[6 Hrs]**
Atmega328: Basics and Architecture, Instruction Set	
Arduino programming:	**[6 Hrs]**
Arduino programming basics, Analog/Digital components and its application with Arduino, IDE for Arduino.	
Other utilities in Arduino:	**[4 Hrs]**
Timers, Analog comparators and hardware interrupts	
Interfacing with peripherals:	**[6 Hrs]**
Communication buses, Interfacing of I/O devices	
Case studies:	**[4 Hrs]**
Case studies of a few projects using Arduino boards and Shields	

Textbooks:

[1]	Brian Evans, “Beginning Arduino Programming”, Springer, 2011

Reference Books:

[1]	Michael J. Pont , “Embedded C”, Pearson Education, 2nd Edition, 2008
[2]	Raj Kamal, “ Embedded Systems – Architecture: Programming and Design”, TMH
[3]	Frank Vahid and Tony Givargis, “Embedded System Design”, Wiley

Note:

[1]	To measure CO1, questions may be of the type – implement simple programs, demonstrate interfacing with peripherals, and solve problems based on basic microcontroller operations.
[2]	To measure CO2, questions may be of the type – identify system requirements, design applications, and develop solutions involving hardware interfacing and custom configurations.
[3]	To measure CO3, questions may be based on deployment of applications, programming using low- and high-level languages, system-level implementation, and real-world embedded applications.

(PCC) Embedded Systems Security						
Course Code	PCC-04		**Examination Scheme**			
Teaching Scheme	3-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	CIE: 50	--	ESE: 50

Course Outcomes: Students will be able to:

1. Recognize vulnerabilities, attacks and need of protection mechanisms for embedded systems.
2. Analyze and evaluate software vulnerabilities and attacks on operating systems.
3. Identify terms/concepts relevant to embedded cryptography.
4. Develop and deploy solutions for security of embedded software and data protection.

Systems Software Considerations:	**[7 Hrs]**
Role of the Operating System, Multiple Independent Levels of Security, Microkernel versus Monolith, Core Embedded Operating System Security Requirements, Access Control and Capabilities, Hypervisors and System Virtualization, I/O Virtualization, Remote Management, Assuring Integrity of the TCB.	
Secure Embedded Software Development:	**[7 Hrs]**
PHASE—Principles of High-Assurance Software Engineering, Minimal Implementation, Component Architecture, Least Privilege, Secure Development Process, Independent Expert Validation, Case Study: HAWS—High-Assurance Web Server, Model-Driven Design.	

Embedded Cryptography:	**[8 Hrs]**
Cryptographic Modes, Block Ciphers, Authenticated Encryption, Public Key Cryptography, Key Agreement, Public Key Authentication, Elliptic Curve Cryptography, Cryptographic Hashes, Message Authentication Codes, Random Number Generation, Key Management for Embedded Systems, Cryptographic Certifications.	
Data Protection Protocols for Embedded Systems:	**[7 Hrs]**
Data-in-Motion Protocols, Data-at-Rest Protocols.	
Emerging Applications:	**[7 Hrs]**
Embedded Network Transactions, Automotive Security, Secure Android, next generation software defined radio.	
Textbooks & Reference books:	

[1]	David Kleidermacher and Mike Kleidermacher, "Embedded Systems Security", Elsevier.
[2]	Gebotys, Catherine H., "Security in Embedded Devices", Springer.
[3]	Stapko T., "Practical Embedded Security", Elsevier/Newnes.

(LC) Embedded System Security Lab						
Course Code	LC-03		**Examination Scheme**			
Teaching Scheme	0-0-2-2		**Theory**	--	--	--
Credits	1		**Laboratory**	CIE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
Implement image enhancement and transformation techniques using computational tools. Apply image segmentation and feature extraction techniques for analysis. Implement image and video compression and motion analysis techniques. Analyze and interpret results of image and video processing algorithms.

List of Experiments:

1) AES implementation in OpenSSL
2) Hash implementation in OpenSSL
3) AES implementation in X-cubeCrypto
4) Hash implementation in X-cubeCrypto
5) PKCS1v1.5_SignVerify in X-cubeCrypto
6) TrustZone creation of Project GPIO toggle
7) MPU Flash Protection
8) True Number Generator
9) AES-ECB on STM32
10) AES-CBC on STM32
11) RDP on STM32
12) Secure boot
13) Secure firmware upgradation
14) TRNG
15) Proprietary code Protection
16) Tamper Protection
17) PKA_ECDSA_Sign
18) Debug authentication using Password
19) HMAC
20) Unique ID

(PCC) Embedded Operating System						
Course Code	PCC-05		**Examination Scheme**			
Teaching Scheme	3-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Interact with and control the board using simple Linux commands.
2. Perform Linux system administration tasks on the Beagle boards and use a range of Linux commands for file and process management.
3. Compare and contrast scripting, hybrid, and compiled programming languages, and their application to the embedded Linux applications.
4. Develop shell scripts and C++ code to control a Beagle board GPIOs, PWM pin and read from analog inputs.
5. Implement C/C++ code that interfaces to and wraps the functionality of devices attached to the UART, I2C and SPI buses.
6. Build circuits that interface to the Beagle board CAN buses and use Linux Socket CAN to send and receive messages.
7. Interface actuators, analog sensors, low-cost display modules to Beagle boards.

Beagle Board Basics	**[5 Hrs]**
Beagle hardware, accessories, capes, handling the board, Beagle software, Linux on Beagle board, Communicating with the board, Controlling the board, basic Linux commands, package management, Updating the kernel, Shutdown.	
Exploring Embedded Linux System	**[6 Hrs]**
Introduction, Booting, Kernel space, User space, Managing Linux system, System administration, Linux commands, Processes.	
Beagle Board Programming	**[6 Hrs]**
Performance of different languages, setting clock frequency, Scripting languages, C and C++ on Beagle boards, LED flashing program using C and C++, Writing multi-call binary, Interfacing to Linux OS – Glibc and Syscall.	
Interfacing to Beagle Board Inputs/Outputs	**[6 Hrs]**
GPIOs, Introduction, Configuration, Linux device tree, Analog Inputs and Outputs, examples such as light meters, controlling servo motors, Callback functions, POSIX threads, Linux poll.	
Interfacing to Beagle Board Buses	**[8 Hrs]**
Introduction, I2C Hardware, Test circuit, I2C tools, I2C communication in C and C++, SPI Hardware, Testing, Bidirectional SPI communication in C and C++, Multiple Slave	

devices, UART, examples LED serial driver, application such as GPS, CAN bus, SocketCAN, Linux CAN utils.	
Interfacing with Physical Environment	**[6 Hrs]**
Interfacing to actuators – DC motors, Stepper motors, Relays, Interfacing to analog sensors, protecting ADC inputs, Signal conditioning, Interfacing local displays, Building C/C++ libraries – shared and static.	
Textbooks & Reference books:	

[1]	Derek Molloy, "Exploring Beaglebone Tools and Techniques for building with Embedded Linux", Wiley, Second Edition, 2019.
[2]	W. Richard Stevens and Stephen A. Rago, "Advanced Programming in Unix Environment", Pearson, Second edition, 2011.
[3]	Robert Love, "Linux System Programming", O'Reilly, Second Edition, 2013.
[4]	Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau, "Operating System: Three Easy Pieces", Version 1.0, 2018.

(LC) Embedded OS Lab						
Course Code	LC-04		**Examination Scheme**			
Teaching Scheme	0-0-2-2		**Theory**			
Credits	1		**Laboratory**	CIE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
Build and boot the board with custom boot loader, Linux kernel and root file system.
Develop applications using user interfaces and device files supported by the drivers managing GPIOs, ADC, PWM and Asynchronous and synchronous communication.

List of Experiments (on Beaglebone boards and suitable open source tool-

1) Booting the board with latest headless image available online and interacting with the board using virtual serial and Ethernet over USB buses.
2) Experimenting with the booting process with step by step loading of multistage bootloader files, kernel image, DTB files and root file system.
3) Creating and using custom root files system using the Busybox.
4) Building the cross tool chain, bootloader, kernel image, DTB files and root file system using the Buildroot.
5) Installing, configuring the latest Eclipse IDE for development and debugging of applications.
6) Interfacing and programming onboard and off-board peripherals connected to GPIO lines.
7) Programming ADCs and PWM peripherals to realize a small real-world application.
8) Programming UART, I2C and SPI buses to carry out transactions with devices connected.

(PCC) Data Analytics on Edge Computing						
Course Code	PCC-06		**Examination Scheme**			
Teaching Scheme	3-0-2-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:
1. Understand the basics of organization of big data.
2. Analyze the architectural issues of big data tools.
3. Design and implement various search methods and visualization techniques for big data analytics.

Fundamentals of Big Data Analytics	**[9 Hrs]**

Overview & analytics life cycle, Need, Structured and multistructured data analysis, Big-data analytics major components, Analytical models and approaches, Big data challenges.

Big Data Architecture	**[9 Hrs]**
Designing and building big data applications, Big data architecture, Distributed Computing platforms and Data Storage, Security and Data Privacy, Application Areas, Application Tools and Platforms.	
Hadoop and MapReduce	**[9 Hrs]**
Clustered Hadoop environment, HDFS and data managements using HDFS, Analytics Using Map Reduce and programming, Map Reduce design patterns.	
Modern Databases and Programming	**[9 Hrs]**
Introduction to Modern databases – No SQL, New SQL, NoSQL Vs RDBMS databases Tradeoffs, Working with MongoDB, Data warehouse system for Hadoop. Introduction to Pig and HIVE – Programming Pig: Engine for executing data flows in parallel on Hadoop, Programming with Hive.	
Textbooks & References:	

[1]	Dean Wampler, Jason Rutherglen, Edward Capriolo, "Programming Hive", O'Reilly Media, 1st Edition, 2012.
[2]	Sawant, Nitin, Shah, Himanshu, "Big Data Application Architecture Q & A: A Problem Solution Approach", Apress, 1st Edition, 2013.
[3]	David Loshin, "Big Data Analytics: From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL, and Graph", Morgan Kaufmann, 1st Edition, 2013.
[4]	Jonathan R. Owens, Jon Lentz, Brian Femiano, "Hadoop Real, World Solutions Cookbook", Packt, 2nd Edition, 2016.

(LC) Data Analytics on Edge Computing Lab						
Course Code	LC-05		**Examination Scheme**			
Teaching Scheme	0-0-2-2		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	4		**Laboratory**	CIE: 50	ESE: 50	--

Course Outcomes: Students will be able to:
Implement and analyze big data processing tasks using Hadoop and MapReduce. Work with NoSQL databases and data analytics tools for edge computing applications.

List of Experiments:

Practicals based on:
1. Hadoop
2. HDFS
3. MapReduce
4. Pig, Hive
5. MongoDB

6. Big data analytics tools for IoT Edge Computing applications.

(PEC) Cloud Computing							
Course Code	PEC-02a		**Examination Scheme**				
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50	
Credits	3		**Laboratory**	--	--	--	

Course Outcomes: Students will be able to:
Deploy Cloud Services using different cloud technologies. Implement and establish cloud computing elements such as virtual machines, web apps, mobile services using data migration techniques. Implement virtualization and availability optimization on the Cloud.

Cloud Computing Overview	**[5 Hrs]**
Introduction to Cloud, Characteristics, Trends in Computing – Distributed computing, Grid computing, Cluster computing, Utility Computing.	
Cloud Computing Models	**[7 Hrs]**
Cloud Deployment models, Cloud service models: PaaS, SaaS, IaaS, Architecture of cloud computing, Containerizing applications, Virtual Machine Provisioning & Manageability.	
IoT Cloud Hosting	**[6 Hrs]**
Service Level Agreement, Map reduce, VM Migration, Container, Docker, Virtual Machine, Hypervisor.	
Fog Computing	**[6 Hrs]**
Mobile Cloud computing, Fog computing, Geospatial cloud, Green cloud, IoT cloud.	
Cloud Security	**[6 Hrs]**
Attacks, Threats, Policy and mechanism, Virtualization, Security issues in SAAS.	
Case Studies	**[7 Hrs]**
Python web App in Microsoft Azure, Google Cloud Platform, Amazon Web Services.	

Textbooks & References:

[1]	Rajkumar Buyya, Christian Vecchiola, S. Thamarai Selvi, (2013), "Mastering Cloud Computing, Foundations and Applications Programming", Morgan Kaufmann.
[2]	Judith Hurwitz, R. Bloor, M. Kanfman, F. Halper (2012), "Cloud Computing for Dummies", Wiley India Edition.
[3]	Rajkumar Buyya, James Broberg, Andrzej M. Goscinski, "Cloud Computing: Principles and Paradigms", Wiley, 2011.
[4]	Gautam Shroff, "Enterprise Cloud Computing - Technology, Architecture, Applications", Cambridge University Press, 2010.

(PEC) AD-CMOS						
Course Code	PEC-02b		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand CMOS device physics and scaling
2. Analyze analog and digital CMOS circuits
3. Design CMOS amplifiers and logic circuits
4. Evaluate performance metrics such as power, noise and speed
5. Interpret layout and design flow using modern tools.

CMOS Device Fundamentals and Scaling	**[6 Hrs]**
MOS transistor operation, second order effects, short channel effects, CMOS scaling trends, threshold voltage variations, mobility degradation, channel length modulation, CMOS inverter characteristics, noise margins, propagation delay, power dissipation.	
Digital CMOS Circuit Design	**[7 Hrs]**
Static CMOS logic, dynamic CMOS logic, pass transistor logic, transmission gates, ratioed circuits, timing analysis, logical effort, delay optimization, low power design techniques, leakage and dynamic power reduction, CMOS logic design trade-offs.	
Analog CMOS Circuit Design	**[6 Hrs]**
Single-stage amplifiers, differential amplifiers, current mirrors, active loads, frequency response, gain-bandwidth product, slew rate, noise analysis, biasing techniques, analog design trade-offs.	
Advanced Analog Circuits	**[8 Hrs]**
Operational amplifiers, multi-stage amplifiers, frequency compensation, stability analysis, switched capacitor circuits, data converters basics (ADC/DAC), mixed-signal design considerations, layout effects in analog design.	
Layout, Design Flow and Reliability	**[6 Hrs]**
CMOS layout design rules, parasitic effects, matching techniques, layout versus schematic (LVS), design rule checking (DRC), EDA tools overview, reliability issues, electromigration, latch-up, process variations, yield considerations.	
Case Study	**[3 Hrs]**
FinFET CMOS technology, analog design in nanoscale, low-voltage design, RF CMOS basics.	

Textbooks & Reference books:

[1]	Behzad Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw-Hill
[2]	Neil H. E. Weste, David Harris, *CMOS VLSI Design*, Pearson.
[3]	Allen and Holberg, *CMOS Analog Circuit Design*, Oxford.

[4]	Razavi, *Fundamentals of Microelectronics*, Wiley.
[5]	Kang and Leblebici, *CMOS Digital Integrated Circuits*, McGraw-Hill.

(PEC) AI-ML						
Course Code	PEC-02c		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand AI and ML fundamentals; apply machine learning algorithms
2. Analyze model performance
3. Design intelligent systems
4. Evaluate deep learning techniques for real-world applications.

Foundations of AI and Machine Learning	**[6 Hrs]**
Introduction to AI, intelligent agents; problem formulation, search techniques, supervised and unsupervised learning, data preprocessing, feature extraction, model evaluation metrics, bias-variance trade-off.	
Supervised Learning Techniques	**[7 Hrs]**
Linear regression, logistic regression, decision trees, random forests, support vector machines, k-nearest neighbors, classification and regression problems, overfitting and regularization techniques.	
Unsupervised Learning and Probabilistic Models	**[6 Hrs]**
Clustering techniques: k-means, hierarchical clustering, dimensionality reduction: PCA, Gaussian mixture models, expectation maximization, anomaly detection, probabilistic reasoning.	
Deep Learning and Neural Networks	**[8 Hrs]**
Artificial neural networks, backpropagation, weight Initialization techniques, activation functions, optimization techniques, convolutional neural networks, recurrent neural networks, LSTM, GRU, sequence modelling, introduction to transformers, model training and tuning.	
Embedded AI and Applications	**[6 Hrs]**
AI for embedded systems, edge AI, model compression, quantization, deployment on embedded platforms, applications in vision, speech, autonomous systems, industrial automation; ethical considerations.	
Case Study	**[3 Hrs]**
Recent trends in AI, generative AI, reinforcement learning, AI in IoT and cyber-physical systems.	
Textbooks & Reference books:	

[1]	Tom M. Mitchell, *Machine Learning*, McGraw-Hill.
[2]	Ian Goodfellow, Yoshua Bengio, Aaron Courville, *Deep Learning*, MIT Press.
[3]	Christopher Bishop, *Pattern Recognition and Machine Learning*, Springer.
[4]	Aurélien Géron, *Hands-On Machine Learning*, O'Reilly.
[5]	Stuart Russell, Peter Norvig, *Artificial Intelligence: A Modern Approach*, Pearson.

(PEC) Automotive Embedded Hardware Development						
Course Code	PEC-02d		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand automotive hardware architecture
2. Analyze ECU hardware design
3. Design embedded hardware systems
4. Evaluate communication interfaces
5. Apply hardware validation techniques in automotive systems.

Automotive Electronics and Hardware Architecture	**[7 Hrs]**
Automotive electronics overview; ECUs; distributed architectures; hardware components; sensors and actuators; powertrain, body and infotainment systems; automotive standards.	
Microcontrollers and Hardware Design	**[6 Hrs]**
Automotive microcontrollers; ARM architecture; peripherals; memory systems; clocking and reset circuits; power supply design; PCB design considerations; hardware-software co-design.	
Communication Interfaces and Networking	**[6 Hrs]**
CAN, LIN, FlexRay, Automotive Ethernet; physical layer design; transceivers; signal integrity; communication protocols; network architecture; fault tolerance in communication.	
Power Electronics and Safety Hardware	**[8 Hrs]**
Power management circuits; voltage regulators; battery management systems; protection circuits; EMI/EMC considerations; ISO 26262 hardware safety concepts; redundancy and diagnostics.	
Hardware Testing and Validation	**[7 Hrs]**
Hardware testing methodologies; debugging techniques; measurement tools; hardware-in-the-loop (HIL); environmental testing; reliability testing; automotive standards compliance.	
Case Study	**[3 Hrs]**
Electric vehicle hardware systems; advanced driver assistance hardware; automotive cybersecurity hardware aspects.	

Textbooks & References:		
[1]	Robert Bosch GmbH, *Automotive Handbook*, Wiley.	
[2]	Nicolas Navet, *Automotive Embedded Systems Handbook*, CRC Press.	
[3]	William Wolf, *Computers as Components*, Morgan Kaufmann	
[4]	Tim Wilmshurst, *Designing Embedded Systems*, Newnes	
[5]	ISO 26262 Standard Documentation.	

(PEC) Wireless Sensor Network						
Course Code	PEC-03a		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. To understand the fundamentals of wireless sensor networks and its application to critical real time scenarios.
2. To study the protocols at various layers and its differences with traditional protocols.
3. To understand the issues pertaining to sensor networks and the challenges involved in managing a sensor network.

Unit I: Introduction	**[4 Hrs]**
Sensor network architecture and its applications, sensor network comparison with Ad Hoc Networks, Sensor node architecture with hardware and software details. Hardware: Examples like mica2, micaZ, telosB, cricket, Imote2, tmote, btnode, and Sun SPOT.	
Unit II: MAC Protocols	**[5 Hrs]**
Fundamentals of MAC Protocols, MAC Protocols for WSNs, Contention-Based protocols: Power Aware Multi-Access with Signaling, Data-Gathering MAC, Contention-Free Protocols: Low-Energy Adaptive Clustering Hierarchy, B-MAC, S-MAC. Dissemination Protocol for Large Sensor Network.	
Unit III: Routing Protocols	**[7 Hrs]**
Issues in designing a routing protocol, classification of routing protocols, table-driven, on-demand, hybrid, flooding, hierarchical, and power aware routing protocols. Routing Challenges and Design Issues, data centric, hierarchical, location based energy efficient routing.	
Unit IV: Deployment and Configuration	**[6 Hrs]**
Localization and Positioning, Coverage and Connectivity, Single-hop and Multi-hop Localization, Self-Configuring Localization Systems, Querying, Data Dissemination and Gathering.	
Unit V: QoS and Energy Management	**[7 Hrs]**
Issues and Challenges in providing QoS, classifications, MAC, network layer solutions, QoS frameworks, need for energy management, classification, battery, transmission power, and system power management schemes.	
Unit VI: Operating Systems	**[6 Hrs]**

Operating System Design Issues, TinyOS, Contiki – Task management, Protothreads, Memory and IO management.		
Unit VII: Case Studies		**[3 Hrs]**
Forest Fire, Disaster Management, Precision Agriculture, Environment monitoring, Military Surveillance.		
Textbooks & Reference books:		
[1]	Shashikant Athawale, "AdHoc and Wireless sensor Network", Pearson, 2022.	
[2]	C. Siva Ram Murthy, and B. S. Manoj, "AdHoc Wireless networks", Pearson Education, 2008.	
[3]	Waltenegus Dargie, Christian Poellabauer, "Fundamentals of Wireless sensor Network", Wiley, 2010.	

(PEC) Advanced VLSI Architecture						
Course Code	PEC-03b		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand advanced VLSI design methodologies and architectures.
2. Analyze performance metrics such as power, delay and area,
3. Design arithmetic and memory subsystems.
4. Evaluate trade-offs in VLSI systems.
5. Interpret modern VLSI design flow using HDL and EDA tools.

VLSI Design Fundamentals and Design Flow	**[7 Hrs]**

VLSI design flow; abstraction levels; Moore's law and scaling trends; CMOS scaling issues; design metrics: area, delay, power; static and dynamic power dissipation; interconnect effects; full custom, semi-custom and FPGA design approaches; introduction to Hardware Description Languages: Verilog/SystemVerilog; RTL design concepts.

Arithmetic and RTL Design	**[7 Hrs]**

High-speed adders: ripple carry, carry look-ahead, carry select, carry skip; multipliers: array, Booth, Wallace tree; division algorithms; pipelining and parallelism; RTL modeling of arithmetic units using Verilog; timing considerations and synthesis constraints.

Memory and Interconnect Architectures	**[6 Hrs]**

SRAM and DRAM architectures; memory organization; cache memory; mapping and replacement policies; memory hierarchy; low power memory techniques; bus-based interconnects; Network-on-Chip (NoC); interconnect optimization.

Low Power Design and EDA Flow	**[8 Hrs]**

Sources of power consumption; clock gating; power gating; multi-threshold CMOS; dynamic voltage and frequency scaling; leakage reduction; introduction to EDA tools; synthesis, placement and routing; timing analysis; RTL-to-GDSII flow; design verification basics.

System-Level Design and Applications	**[6 Hrs]**

System-on-Chip (SoC) architectures; hardware accelerators; VLSI architectures for DSP and AI applications; FPGA-based prototyping; reliability and fault tolerance; design trade-offs in embedded VLSI systems.

Case Study	**[3 Hrs]**

FinFET and GAA technologies; AI/ML hardware accelerators; emerging trends in VLSI design automation.

Textbooks & References:

[1]	Wayne Wolf, *Modern VLSI Design: System-on-Chip Design*, Pearson.

[2]	Samir Palnitkar, *Verilog HDL: A Guide to Digital Design and Synthesis*, Pearson.
[3]	Sung-Mo Kang, Yusuf Leblebici, *CMOS Digital Integrated Circuits: Analysis and Design*,

(PEC) SCADA Systems Applications						
Course Code	PEC-03c		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand SCADA architecture and components
2. Analyze industrial communication protocols
3. Design SCADA systems for automation
4. Evaluate cybersecurity and reliability
5. Integrate SCADA with IoT platforms.

SCADA Fundamentals and Architecture	[6 Hrs]
SCADA overview; industrial automation systems; SCADA architecture; centralized and distributed systems; components: RTU, PLC, HMI, master station; applications in power, oil and gas, water systems.	
Data Acquisition and Field Systems	**[7 Hrs]**
Sensors and actuators; signal conditioning; data acquisition systems; analog and digital I/O; PLC and RTU operation; interfacing; real-time data acquisition; industrial instrumentation.	
Communication Protocols and Networking	**[6 Hrs]**
Industrial communication requirements; wired and wireless communication; protocols: Modbus, DNP3, IEC 60870, OPC; network topologies; latency and reliability; MQTT protocol; edge computing concepts for industrial systems	
SCADA Software and Visualization	**[8 Hrs]**
Supervisory control; HMI design; alarm management; data logging and trending; database management; visualization tools; SCADA software platforms such as Wonderware and Siemens WinCC; report generation; real-time monitoring.	
Security, IoT Integration and Applications	**[6 Hrs]**
Cybersecurity in SCADA; threats and vulnerabilities; encryption and authentication; redundancy and fault tolerance; Industrial IoT integration; cloud-based SCADA; smart grid applications; Industry 4.0 concepts.	

Case Study		**[3 Hrs]**
Case studies of SCADA cyber-attacks; digital twins in industrial automation; recent trends in industrial IoT systems.		
Textbooks & References:		
[1]	Stuart A. Boyer, *SCADA: Supervisory Control and Data Acquisition*, ISA Publications	
[2]	Gordon Clarke, Deon Reynders, *Practical Modern SCADA Protocols*, Newnes (Elsevier).	
[3]	Bela G. Liptak, *Instrument Engineers' Handbook: Process Control and Optimization*, CRC Press	
[4]	David Bailey, Edwin Wright, *Practical SCADA for Industry*, Newnes	
[5]	Sandip Lahiri, *Industry 4.0 and Industrial IoT*, Springer.	
[6]	Andrew Ginter, Eric Byres, *SCADA Security: What You Need to Know*, Industrial Defender.	

(PEC) Automotive Embedded Software						
Course Code	PEC-03d		**Examination Scheme**			
Teaching Scheme	3-0-0-1		**Theory**	TA: 20	MSE: 30	ESE: 50
Credits	3		**Laboratory**	--	--	--

Course Outcomes: Students will be able to:

1. Understand automotive embedded system architecture and standards
2. Analyze real-time constraints
3. Develop automotive software using standard frameworks
4. Evaluate safety and reliability
5. Apply model-based design and testing techniques.

Automotive Embedded Systems and Networks	**[6 Hrs]**
Automotive electronics overview; ECUs; distributed architectures; in-vehicle networks; CAN, LIN, FlexRay, Automotive Ethernet; communication protocols and standards.	
Automotive Software Architecture	**[7 Hrs]**
Layered architecture; AUTOSAR architecture; software components; runtime environment; basic software modules; communication stack; abstraction layers; configuration and integration.	
Real-Time Systems and Model-Based Design	**[6 Hrs]**
Real-time systems; scheduling algorithms; task management; inter-process communication; OSEK/VDX; timing constraints; MATLAB/Simulink based model-based design; code generation concepts.	
Functional Safety and Diagnostics	**[8 Hrs]**
ISO 26262 standard; hazard analysis and risk assessment; ASIL classification; fault detection; diagnostics; redundancy; fail-safe and fail-operational systems.	
Testing, Validation and Advanced Applications	**[6 Hrs]**
Software testing methodologies; Software-in-the-loop (SIL); Hardware-in-the-loop (HIL); validation and verification; ADAS systems; sensor integration; V2X communication; autonomous driving concepts; automotive cybersecurity basics.	

Case Study	**[3 Hrs]**
Electric vehicle control systems; over-the-air updates; software-defined vehicles; case studies of modern automotive platforms.	
Textbooks & References:	

[1]	Nicolas Navet, Francoise Simonot-Lion, *Automotive Embedded Systems Handbook*, CRC Press.
[2]	Ronald K. Jurgen, *Automotive Electronics Handbook*, McGraw-Hill.
[3]	Robert Bosch GmbH, *Automotive Handbook*, Wiley.
[4]	AUTOSAR Consortium, *AUTOSAR Technical Specifications* (online resources).
[5]	Michael Barr, Anthony Massa, *Programming Embedded Systems*, O'Reilly.
[6]	ISO 26262 Standard Documentation (Functional Safety for Road Vehicles).

(MLC) Technical Communication Skills

Course Code	MLC-02		**Examination Scheme**			
Teaching Scheme	1-1-0-0		**Theory**	TA: 50	MSE: 50	--
Credits	2		**Laboratory**	ISE: 100	--	--

Course Outcomes: Students will be able to:

1. Produce effective dialogue for business related situations.
2. Use listening, speaking, reading and writing skills for communication purposes and attempt tasks by using functional grammar and vocabulary effectively.
3. Analyze critically different concepts/principles of communication skills.
4. Demonstrate productive skills and have a knack for structured conversations.
5. Appreciate, analyze, evaluate business reports and research papers.

Fundamentals of Communication	**[8 Hrs]**
7 Cs of communication, common errors in English, enriching vocabulary, styles and registers.	
Aural-Oral Communication	**[8 Hrs]**
The art of listening, stress and intonation, group discussion, oral presentation skills.	
Reading and Writing	**[8 Hrs]**
Types of reading, effective writing, business correspondence, interpretation of technical reports and research papers.	
Textbooks & References:	

[1]	Raman Sharma, "Technical Communication", Oxford University Press.
[2]	Raymond Murphy, "Essential English Grammar" (Elementary & Intermediate), Cambridge University Press.

[3]	Mark Hancock, "English Pronunciation in Use", Cambridge University Press.
[4]	Shirley Taylor, "Model Business Letters, Emails and Other Business Documents", Prentice Hall, Seventh Edition.
[5]	Thomas Huckin, Leslie Olsen, "Technical writing and Professional Communications for Nonnative speakers of English", McGraw Hill.

(LLC) Liberal Learning Course

Course Code	LLC-01		**Examination Scheme**			
Teaching Scheme	0-0-2-2		**Theory**	--	--	--
Credits	1		**Laboratory**	ISE: 100	--	--

Course Outcomes: Students will be able to:
Learn new topics from various disciplines without any structured teaching or tutoring. Understand qualitative attributes of a good learner. Understand quantitative measurements of learning approaches and learning styles. Understand various sources and avenues to harvest/gather information. Assess yourself at various stages of learning.

Areas:	**[12 Hrs]**

Agriculture (Landscaping, Farming, etc.), Business (Management, Entrepreneurship, etc.), Defense (Study about functioning of Armed Forces), Education (Education system, Policies, Importance, etc.), Fine Arts (Painting, Sculpting, Sketching, etc.), Linguistics, Medicine and Health (Diseases, Remedies, Nutrition, Dietetics, etc.), Performing Arts (Music, Dance, Instruments, Drama, etc.), Philosophy, Social Sciences (History, Political Science, Archaeology, Geography, Civics, Economics, etc.)

Evaluation:	**[6 Hrs]**

T1: A brief format about your reason for selecting the area, sub area, topic and a list of 5 questions (20 marks). T2: Identify and meet an expert (in or outside college) in your choice of topic and give a write up about their ideas regarding your topic (video/audio recording of your conversation permitted) (20 marks). ESE: Presentation in the form of PPT, demonstration, performance, charts, etc. in front of everyone involved in your sub area and one external expert (60 marks).

Resources:	**[6 Hrs]**

Expert(s), Books, Texts, Newspaper, Magazines, Research Papers, Journal, Discussion with peers or faculty, Internet, etc.

Semester - III

Dissertation Phase – I						
Course Code			**Examination Scheme**			
Teaching Scheme	0-0-22-12		**Theory**	TA: -	MSE: -	ESE: -
Credits	11		**Laboratory**	CIE: 60	--	ESE-40

Course Outcomes: Students will be able to:

1. Conceive a problem statement either from rigorous literature surveys or from the requirements raised by external entity.
2. Design, implement and test the prototype/algorithm to solve the conceived problem.
3. Publish the research work in journals/conferences of repute contributing to growth of technology in the domain.

Guidelines:

As per the AICTE directives, the dissertation is a yearlong activity, to be carried out and evaluated in two phases i.e. Phase – I: July to December and Phase – II: January to June.

The dissertation may be carried out preferably in-house i.e. department's laboratories and centres OR in industry allotted through department's T & P coordinator.

After multiple interactions with guide and based on comprehensive literature survey, the student shall identify the domain and define dissertation objectives. The referred literature should preferably include IEEE/IET/IETE/Springer/Science Direct/ACM journals in the areas of Computing and Processing (Hardware and Software), Circuits-Devices and Systems, Communication- Networking and Security, Robotics and Control Systems, Signal Processing and Analysis, Machine Learning, IoT and any other related domain. In case of Industry sponsored projects, the relevant application notes, white papers, product catalogues should be referred and reported.

Student is expected to detail out specifications, methodology, resources required, critical issues involved in design and implementation and phase wise work distribution and submit the proposal within a month from the date of registration.

Phase – I deliverables:
A document report comprising of summary of literature survey, detailed objectives, project specifications, paper and/or computer aided design, proof of concept/functionality, part results, A record of continuous progress.

Phase – I evaluation:
A committee comprising of guides of respective specialization shall assess the progress/performance of the student based on report, presentation and Q & A. In case of unsatisfactory performance, committee may recommend repeating the phase-I work.

Guide along with appointed external examiner shall assess the progress/performance of the student based on report, presentation and Q & A. In case of unsatisfactory performance, committee may recommend for extension or repeating the work.

Semester - IV

Dissertation Phase – II						
Course Code			**Examination Scheme**			
Teaching Scheme	0-0-22-12		**Theory**	TA: -	MSE: -	ESE: -
Credits	11		**Laboratory**	CIE: 60	--	ESE-40

Course Outcomes: Students will be able to:

1. Conceive a problem statement either from rigorous literature surveys or from the requirements raised by external entity.
2. Design, implement and test the prototype/algorithm to solve the conceived problem.
3. Publish the research work in journals/conferences of repute contributing to growth of technology in the domain.

Guidelines:

As per the AICTE directives, the dissertation is a yearlong activity, to be carried out and evaluated in two phases i.e. Phase – I: July to December and Phase – II: January to June.

The dissertation may be carried out preferably in-house i.e. department's laboratories and centres OR in industry allotted through department's T & P coordinator.

After multiple interactions with guide and based on comprehensive literature survey, the student shall identify the domain and define dissertation objectives. The referred literature should preferably include IEEE/IET/IETE/Springer/Science Direct/ACM journals in the areas of Computing and Processing (Hardware and Software), Circuits-Devices and Systems, Communication- Networking and Security, Robotics and Control Systems, Signal Processing and Analysis, Machine Learning, IoT and any other related domain. In case of Industry sponsored projects, the relevant application notes, white papers, product catalogues should be referred and reported.

Student is expected to detail out specifications, methodology, resources required, critical issues involved in design and implementation and phase wise work distribution and submit the proposal within a month from the date of registration.

During phase – II, student is expected to exert on design, development and testing of the proposed work as per the schedule. Accomplished results/contributions/innovations should be published in terms of research papers in reputed journals and reviewed focused conferences OR IP/Patents.

Phase – II deliverables:
A dissertation report as per the specified format, developed system in the form of hardware and/or software, A record of continuous progress.

Phase – II evaluation:
Guide along with appointed external examiner shall assess the progress/performance of the student based on report, presentation and Q & A. In case of unsatisfactory performance, committee may recommend for extension or repeating the work.

<table>
<tr><td colspan="8">(SLC) Massive Open Online Course – I & II</td></tr>
<tr><td>Course Code</td><td colspan="2">SLC-01/02</td><td colspan="5">Examination Scheme</td></tr>
<tr><td>Teaching Scheme</td><td>3-0-0-3</td><td></td><td>Theory</td><td>--</td><td>MSE: 25</td><td colspan="2">ESE: 75</td></tr>
<tr><td>Credits</td><td>3</td><td></td><td>Laboratory</td><td>--</td><td>--</td><td colspan="2">--</td></tr>
<tr><td colspan="8">Course Outcomes: Students will be able to:
Enhance subject knowledge through self-paced online learning platforms. Apply knowledge gained through MOOC courses to embedded systems and computing applications.</td></tr>
<tr><td colspan="7">MOOC Courses Identified:</td><td>[36 Hrs]</td></tr>
<tr><td colspan="8">1) Real Time Embedded Systems
2) CMOS Design
3) Edge Computing
4) Advanced IoT Applications</td></tr>
</table>