Department of Electronics and Communication Engineering

Details of Lesson Plan

S.No.	Particulars	Details
1.	Course Name	Microcontroller and Applications
2.	Course Code	ECE-401
3.	Academic Year	2024-2025
4.	Semester	4th
5.	Number of Lesson plans	45
6.	Faculty Assigned	Ms. Shiveta Bhat

Faculty Signature

Lesson Plan No. 1	**Course Name: Microcontroller and applications** **Topic: Introduction to Microcontrollers**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Articulate the basic concept of microcontroller and peripheral interfacing b. Explain the evolution of microcontrollers. c. Illustrate the various applications of microcontrollers d. Appreciate advantages of microcontroller-based real-life applications
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you heard about microcontrollers? - What do you mean by an embedded system? - What do you mean by a controller? - Which microcontroller is commonly used in embedded systems? - Introduce the basics of microcontrollers. - Discuss the evolution of microcontrollers. - Talk about the microcontrollers available in the market today. - Define Microcontroller formally. - Highlight the importance of interfacing. - Talk about utilities and their role in microcontrollers. - Highlight the important characteristics of microcontrollers. - Emphasize the importance of microcontrollers in our daily lives. - **Development** (30 minutes) a **Microcontroller** - Introduce the concept of a microcontroller. - Show a video on the evolution of microcontrollers. - Explain the concept of interfacing. **b. Interfacing** - Explain the concept of interfacing in microcontrollers. - Discuss projects that can be developed using microcontrollers. - Highlight major players in microcontroller manufacturing (e.g., Intel, Atmel, Microchip, etc.). **c. Advantages of Microcontrollers** - Low power consumption - Compact size - Cost-effectiveness

	- Versatility - Exercise (5 minutes) – Give students time to discuss the various applications of microcontrollers.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video Lecture https://www.youtube.com/watch?v=fS7FFOaC_iQ https://youtu.be/o6W0opScrKY 3. Homework Ask students to make a chart explaining the evolution of microcontrollers, discussing the advancement in features over time. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is a microcontroller? 2. Why is a microcontroller important? 3. Who are the key manufacturers of microcontrollers? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2	**Course Name: Microcontroller and applications** **Topic: Block Diagram of 8051**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the significance of a block diagram in microcontrollers b. Identify the key components of a microcontroller block diagram. c. Explain the function of each block in the microcontroller architecture.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you seen a block diagram before? - What do you think a microcontroller consists of internally? - Can you name any essential components of a microcontroller? - Introduce the concept of a block diagram. - Discuss the significance of block diagrams in understanding microcontroller architecture. - Explain how block diagrams help in visualizing data flow and internal operations. - - **Development** (30 minutes) **a. Understanding the Block Diagram** - Introduce the concept of a block diagram and its importance. - Explain how each block represents a functional part of a microcontroller. **b. Components of a Microcontroller Block Diagram** - **Central Processing Unit (CPU)** – Describe its function in executing instructions. - **Memory (ROM, RAM, EEPROM)** – Explain the purpose of different types of memory. - **Input/Output Ports** – Discuss their role in interfacing external devices. - **Timers and Counters** – Highlight their significance in time-based operations. - **Interrupts** – Explain how they enhance the efficiency of a microcontroller. - **Serial and Parallel Communication Interfaces** – Discuss their role in communication with peripherals. - Exercise (5 minutes) – Give students time to analyze and identify the components of a microcontroller block diagram.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	2. Suggested material https://youtu.be/2AVOxLPKjeA 3. Homework Ask Students to draw a labeled block diagram of a microcontroller and explain the function of each block. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the purpose of a block diagram? 2. Why are different blocks necessary in a microcontroller? 3. How does each block contribute to the microcontroller’s functionality? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3	**Course Name: Microcontroller and applications** **Topic: Pin Diagram**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the significance of a pin diagram in microcontrollers b. Identify the key pins of a microcontroller and their function. c. Explain the role of different pins in microcontroller operation
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you seen a pin diagram before? - What do you think the pins of a microcontroller are used for? - Can you name any specific pins you have heard of before? - Introduce the concept of a pin diagram. - Discuss the significance of pin diagrams in microcontroller - functionality. - Explain how pin configurations vary across different - microcontroller families - **Development** (30 minutes) **Understanding the Pin Diagram** - Explain what a pin diagram is and why it is essential. - Discuss the role of each pin in the microcontroller. **Key Pins in a Microcontroller** - **Power Supply Pins (Vcc, GND)** – Describe their function in providing operating voltage. - **Oscillator Pins (XTAL1, XTAL2)** – Explain their role in generating system clock pulses. - **Reset Pin** – Discuss how it is used to restart the microcontroller. - **Input/Output Ports (PORT0, PORT1, etc.)** – Explain their function in connecting external devices. - **Interrupt Pins (INT0, INT1, etc.)** – Describe their role in handling external events. - **Control Pins (ALE, PSEN, RD, WR)** – Explain their function in memory and peripheral communication. - **Serial Communication Pins (TXD, RXD)** – Highlight their - importance in UART-based data transfer.Exercise (5 minutes) – Exercise (5 minutes) – Give students time to analyze and identify the pins of a microcontroller using a sample pin diagram.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 2. Suggested material https://youtu.be/2AVOxLPKjeA 3. Homework Ask Students to draw a labeled pin diagram of a microcontroller and explain the function of each pin. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the purpose of each pin in a microcontroller? 2. Why are pin configurations crucial in designing applications? 3. How do different microcontrollers have varying pin layouts? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4	**Course Name: Microcontroller and applications** **Topic: Architecture of 8051**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Articulate the basic concept of microcontroller and peripheral interfacing. b. Explain the evolution of microcontrollers. c. Illustrate the various applications of microcontrollers. d. Appreciate advantages of microcontroller-based real-life applications.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. o Evolution of Microcontrollers - - Introduce the architecture of 8051. - **Development** (30 minutes) - Draw the Architecture of 8085 - Explain the various blocks present in the architecture of 8085 - Inside the 8051 - Registers - ALU of microcontroller 8051 - Exercise (5 minutes) – Give students time to discuss the various applications of microcontrollers. How many bytes of RAM and onchip ROM available in 8051 Microcontroller?
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/2AVOxLPKjeA Bharat Acharya: lecture 1: https://youtu.be/2-geyR_aM28 https://www.udemy.com/course/8051-architecture-bharat-acharya/ 3. Homework Ask students to make a chart explaining the evolution of microcontrollers, discussing the advancement in features over time.

	Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is a microcontroller? 2. Why is a microcontroller important? 3. Who are the key manufacturers of microcontrollers? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5	**Course Name: Microcontroller and applications** **Topic: Comparison of Microprocessor and Microcontroller**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the key differences between a microprocessor and a microcontroller. b. Identify typical applications for microprocessors and microcontrollers. c. Analyze the components of both microprocessors and microcontrollers and how they impact performance and application suitability. d. Appreciate the advantages and limitations of both microprocessors and microcontrollers in various contexts.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Briefly introduce microprocessors and microcontrollers as integral parts of electronic systems. - Define microprocessor: A central unit of a computer system responsible for executing instructions, usually a CPU. - Define microcontroller: A compact integrated circuit designed for controlling electronic devices, including a processor, memory, and input/output peripherals. - **Development** (30 minutes) **- Key Differences Between Microprocessor and Microcontroller** **- Components:** o **Microprocessor:** Typically includes only the central processing unit (CPU) without onboard memory, I/O ports, or other peripherals. o **Microcontroller:** A complete system with the CPU, memory (RAM, ROM), I/O ports, timers, and often additional peripherals like ADCs and DACs. **- Functionality and Purpose:** o **Microprocessor:** Primarily designed to perform computational tasks and is usually used in complex systems such as personal computers, laptops, and workstations. o **Microcontroller:** Designed for specific control tasks within embedded systems like robotics, home appliances, automotive systems, etc. It often runs a dedicated task and interacts directly with hardware.

	- **Architecture:** o **Microprocessor:** May use a more complex architecture with separate buses for data, address, and control. It relies on external components (memory, I/O, etc.) to function. **Microcontroller:** Typically has a simpler architecture with integrated memory and peripherals, making it a more compact, self-contained solution. **Power Consumption:** o **Microprocessor:** Generally consumes more power due to its advanced computational abilities and external peripherals. o **Microcontroller:** Designed to consume less power, making it ideal for battery-operated devices or low-power applications. - **Cost:** o **Microprocessor:** More expensive due to higher processing power and the need for additional external components like memory and I/O devices. o **Microcontroller:** More cost-effective, as it includes everything needed in one package, making it suitable for mass-produced consumer electronics and low-cost applications. - **Speed and Processing Power:** o **Microprocessor:** Typically faster, with higher clock speeds and processing power suited for complex tasks and multitasking environments. o **Microcontroller:** Slower than microprocessors but adequate for real-time control applications where speed requirements are more modest. - **Programming:** o **Microprocessor:** Requires a full-fledged operating system (OS) and software development environment. o **Microcontroller:** Often uses simple programming environments for embedded systems and may run embedded software (like C or assembly code) directly on the hardware. **b. Applications of Microprocessors and Microcontrollers** - **Microprocessor Applications:** o Personal computers o Laptops and desktops o Servers and workstations o Gaming consoles o High-performance computing systems - **Microcontroller Applications:** o Embedded systems (e.g., washing machines, microwave

	ovens) o Automotive control systems (e.g., engine control units) o Robotics o Home automation systems o Medical devices o Smartwatches and fitness trackers **c. Advantages and Limitations (5 minutes):** **- Microprocessor Advantages:** o High processing power for complex tasks o Suitable for general-purpose computing o Can handle multitasking and complex operating systems **- Microprocessor Limitations:** o Higher power consumption o Requires external components like memory and I/O interfaces o More expensive **- Microcontroller Advantages:** o Low power consumption o Compact and cost-effective o Built-in memory and peripherals make it ideal for embedded control systems **- Microcontroller Limitations:** o Limited processing power and speed o Not suited for general-purpose computing o Limited multitasking capabilities - Exercise (5 minutes) – Ask students to compare and contrast a microprocessor-based system (e.g., a desktop computer) with a microcontroller-based system (e.g., an embedded system in a washing machine). Students should focus on aspects like power consumption, cost, complexity, and intended use.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/2AVOxLPKjeA Bharat Acharya: lecture 1: https://youtu.be/2-geyR_aM28 https://www.udemy.com/course/8051-architecture-bharat-acharya/ Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the main difference between a microprocessor and a microcontroller in terms of architecture and functionality?

	2. Why would you choose a microcontroller over a microprocessor for an embedded system? 3. Who benefits from the use of microprocessors and microcontrollers (e.g., consumers, industries, engineers)? 4. What factors should be considered when selecting a microprocessor or microcontroller for a specific application? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 6	**Course Name: Microcontroller and applications** **Topic: Assembler Directive of 8051**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Define assembler directives and their role in 8051 programming. b. Identify different types of assembler directives used in 8051. c. Explain the significance of each directive in program execution d. Apply assembler directives in simple assembly language program
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you ever written an assembly language program? - What do you think is the role of directives in programming? - Why do we need directives in microcontroller programming? - Provide a brief overview of assembler directives and their importance. - Explain how directives help in memory management, code readability, and efficiency. - **Development** (30 minutes) **a. Understanding Assembler Directives** - Define assembler directives. - Discuss the necessity of directives in 8051 assembly programming. **b. Types of Assembler Directives** - **Data Definition Directives** – DB, DW, EQU, ORG - Demonstrate a simple 8051 assembly program using directives. - Explain how directives help in defining constants, memory locations, and macros. Exercise (5 minutes) – Write a program using ORG directive to start the mainprogram from 300H address. • Ask students to write a simple assembly language program using directives and explain its function.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/Vrazx7AsutM 3. Homework

	Ask students to write an 8051 assembly language program using at least three assembler directives and explain their purpose. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why are assembler directives necessary in assembly programming? 2. How do assembler directives make programming more efficient? 3. Can we write an assembly program without using directives? Why or why not? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 7	Course Name: Microcontroller and applications Topic: Addressing Modes of 8051	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Define addressing modes and their role in 8051 programming. b. Identify different types of addressing modes used in 8051. c. Explain the significance of each addressing mode in program execution d. Apply different addressing modes in simple assembly language programs
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is addressing mode? - Have you ever written an assembly language program? - What do you think is the role of addressing modes in programming? - How does a microcontroller access memory and registers? - Provide a brief overview of addressing modes and their importance. - Explain how addressing modes help in efficient instruction execution and memory utilization. - - **Development** (30 minutes) **a. Understanding Addressing Modes** - Define addressing modes. - Discuss the necessity of addressing modes in 8051 assembly programming. **b. Types of Addressing Modes in 8051** - **Immediate Addressing Mode** – Operand is specified in the instruction itself. - **Register Addressing Mode** – Operand is stored in a register. - **Direct Addressing Mode** – Operand is stored in a specific memory location. - **Indirect Addressing Mode** – Address of the operand is stored in a register. - **Indexed Addressing Mode** – Used for accessing lookup tables and external memory. **c. Practical Applications** - Demonstrate a simple 8051 assembly program utilizing different addressing modes. - Explain how addressing modes impact program efficiency and

	execution speed. - Exercise (5 minutes) – Ask students to write a simple assembly language program using at least two different addressing modes and explain their function.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/7q4VVR33p2g 3. Homework Ask students to write an 8051 assembly language program demonstrating all five addressing modes and explain their significance. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why do we need different addressing modes in assembly programming? 2. How do addressing modes affect memory usage and execution time? 3. Can you write an assembly program using only one addressing mode? Why or why not? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 8	**Course Name: Microcontroller and applications** **Topic: Instructions Set of 8051 Microcontroller**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Define the instruction set of the 8051 microcontroller. b. Classify different types of instructions in 8051. c. Write simple assembly programs using different instruction types
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you ever written an assembly language program? - What do you understand by the term 'instruction'? - Why do microcontrollers require specific instructions? - Provide a brief overview of the 8051 instruction set. - Explain the role of instructions in programming the 8051 microcontroller. - - **Development** (30 minutes) **a. Understanding Instruction Sets** - Define the instruction set. - Discuss the importance of instructions in microcontroller programming. **b. Classification of Instructions in 8051** - **Data Transfer Instructions** – MOV, PUSH, POP, XCH, etc. - **Arithmetic Instructions** – ADD, SUB, MUL, DIV, INC, DEC, etc. - **Logical Instructions** – AND, OR, XOR, CPL, CLR, etc. - **Branching Instructions** – LJMP, SJMP, JZ, JNZ, DJNZ, etc. - **Bit Manipulation Instructions** – SETB, CLR, CPL, MOV, etc. c. Exercise (5 minutes) – Write a looping command (Program) in order to better understand the looping instruction set.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/7q4VVR33p2g

	3. Homework Ask students to create a chart classifying 8051 instructions with examples and usage. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why does the 8051 microcontroller have different categories of instructions? 2. How do instructions affect program execution time and efficiency? 3. Can you create a program using only logical instructions? Why or why not? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 9	Course Name: Microcontroller and applications Topic: 8051 Assembly Language Programming	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the basics of 8051 Assembly Language Programming. b. Identify and use different instruction types in 8051 assembly programming. c. Write and execute simple assembly language programs for the 8051 microcontroller
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you ever written an assembly language program? - What do you understand by low-level programming? - Why do microcontrollers use assembly language programming? - Provide an overview of assembly language and its role in microcontroller programming. - Explain how assembly language differs from high-level programming languages. - - **Development** (30 minutes) **a. Basics of Assembly Language Programming** - Explain the structure of an 8051 assembly language program. - Discuss the key components: Opcode, Operands, Labels, Comments. **b. Writing Simple Assembly Language Programs** - Demonstrate simple programs such as: ○ Moving data between registers and memory. ○ Performing arithmetic operations (addition, subtraction, multiplication, division). ○ Logical operations (AND, OR, XOR, NOT). ○ Jump and loop instructions. - Exercise (5 minutes) – Ask students to write a simple assembly program for adding two numbers and store the result in a register.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/Gru27jV1mms 3. Homework

	Ask students to write an 8051 assembly program that performs multiplication of two numbers and stores the result in a memory location. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What are the key benefits of using assembly language in microcontroller programming? 2. How does writing in assembly compare to using high-level languages for microcontroller programming? 3. What are the challenges faced when debugging assembly programs? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 10	**Course Name: Microcontroller and applications** **Topic: 8051 Assembly Language Programming**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the basics of 8051 Assembly Language Programming. b. Identify and use different instruction types in 8051 assembly programming. c. Write and execute simple assembly language programs for the 8051 microcontroller
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Provide an overview of assembly language and its role in microcontroller programming. - Explain how assembly language differs from high-level programming languages. - - **Development** (30 minutes) Introduction to Assembly Language Writing programs for : - Loops - Jumps - Arithmetic and logical instructions - Exercise (5 minutes) – WAP to add first 10 numbers using assembly language
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/Gru27jV1mms 3. Homework Ask students to write an 8051 assembly program that performs multiplication of two numbers and stores the result in a memory location. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What are the key benefits of using assembly language in microcontroller programming? 2. How does writing in assembly compare to using high-level

	languages for microcontroller programming? 3. What are the challenges faced when debugging assembly programs? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 11	**Course Name: Microcontroller and applications** **Topic: Data Types of 8051 Programming In Embedded C**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Familiarize with the difference between C, embedded C and Assembly programming b. Understan the different types of Data Types in embedded C c. Articulate the application of data types to different situations
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is a data type? - What data types are you aware of? - How do different data types impact programming? - Explain the difference between C, Embedded C, and Assembly programming with real-world examples. - - **Development** (30 minutes) - Introduction to Embedded C Language Programming - Explain Embedded C and why it is used for microcontroller programming (e.g., 8051). - Discuss the difference between high-level C and Embedded C. - Introduce data types in Embedded C, particularly in the context of 8051 microcontrollers. - Types of Data Types in Embedded C Basic Data Types in 8051 Microcontrollers: o char (signed and unsigned) o int (signed and unsigned) o float (limited use in 8051) o bit (for single-bit storage) o sbit, sfr (for direct bit and special function register access) - Signed and Unsigned Character in Embedded C - Explain signed vs. unsigned characters: o signed char stores -128 to 127 o unsigned char stores 0 to 255 - Exercise (5 minutes) – WAP to add first 10 numbers using embedded c language
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested video https://youtu.be/Y1Q1Cq1dduA

	Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why do we use different data types in Embedded C? 2. How do signed and unsigned characters impact memory usage? 3. Who uses Embedded C in real-world applications? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 12	**Course Name: Microcontroller and applications** **Topic: Data Types of 8051 Programming In Embedded C**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the different types of Data Types in embedded C b. Articulate the application of data types to different situations
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is a data type? - What data types are you aware of? - What is the difference between the types of data types? - **Development** (30 minutes) - Explanation of different types of data types used in 8051 microcontrollers: - **Signed and Unsigned Int**: Explain the difference and usage in 8051 programming. - **Sbit (Single Bit Variables)**: Introduce sbit for bit-level operations. - **SFR (Special Function Register)**: Explain how SFRs help in controlling hardware. - Provide examples demonstrating the use of these data types. - - Exercise (5 minutes) – Write an 8051 C program using a **signed int** data type.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/Y1Q1Cq1dduA Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What are the different data types in Embedded C? 2. Why is data type selection important? 3. Who decides the data type choice in embedded programming? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 13	**Course Name: Microcontroller and applications** **Topic: Time Delay and Loop Operation**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept of time delay in embedded systems. b. Explore different methods used for delay generation in 8051 microcontrollers. c. Comprehend loop operations and their role in programming
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is a delay in a microcontroller? - Why do we need delays in embedded systems? - How can delays be generated in a microcontroller? - What is a loop, and how is it used in programming? - - **Development** (30 minutes) - **Introduction to Time Delay** - Define time delay and explain its importance in embedded systems (e.g., LED blinking, sensor readings). - Explain how 8051 microcontrollers generate delays. - **Methods of Delay Generation in 8051** - Software delay (using loops) - Timer-based delay (using Timer 0 and Timer 1 in different modes) - **Introduction to Looping Operations** - Explain how loops (for, while, do-while) are used to create delays. - Demonstrate a simple loop-based delay program. - Exercise (5 minutes) – Write a program to add a 2-microsecond delay in 8051.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/lRa3kDroZhI Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss.

	1. What is the best method for generating delays in 8051? 2. Why do we use loops for delays? 3. How can timers improve delay accuracy? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 14	**Course Name: Microcontroller and applications** **Topic**: **Logic Operations, I/O programming and Data Conversion**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand logical operations and their applications in 8051 programming. b. Learn the basics of I/O programming in the 8051 microcontroller. c. Familiarize themselves with data conversions in 8051 Embedded
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are logical operations? - How many I/O ports does the 8051 have? - Which port is used only for I/O operations? - **Development** (30 minutes) - **Introduction to Logical Operations in 8051** - Explain **bitwise logical operations** in Embedded C: - **AND (&)** – Used for bit masking - **OR (|)** – Used to set bits - **XOR (^)** – Used for toggling bits - **NOT (~)** – Used for bit inversion - **Rotate Right (RR)** and **Rotate Left (RL)** – Used in bitwise manipulation - Exercise (5 minutes) – Write 8051 C programming based on Data conversion, and logical programming. Perform **data conversion (binary to decimal)** Use logical operations to manipulate an **I/O port**
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested material https://youtu.be/dGOUcka1g4Q https://youtu.be/ZNbhMZ0Uj74 https://youtu.be/AIME92jvANQ Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why are logical operations essential in embedded programming?

	2. How does 8051 handle I/O operations? 3. What are some real-world applications of data conversions in microcontrollers? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 15	Course Name: Microcontroller and applications Topic: Accessing Code Space, Data Serialization	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept of accessing Code ROM Space in 8051. b. Differentiate between RAM data space and Code data space. c. Familiarize themselves with Data Serialization in 8051.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. What do you mean by code memory/ program memory? What is the size of RAM & ROM in 8051? - **Development** (30 minutes) - Introduction to Accessing Code ROM Space in 8051 - 8051 memory architecture: - RAM: Used for temporary storage, registers, and stack. (128 bytes in standard 8051) - ROM (Code Space): Stores program instructions and constant data. (4 KB in standard 8051) - Accessing Code Space vs RAM Data Space: - Code memory is accessed using the MOVC instruction. - RAM data is accessed using MOV instructions. - **What is Data Serialization?** - Converting parallel data (multiple bits at once) into serial data (one bit at a time). - Used in UART, SPI, I2C communication. - Serialization Process: - Data is sent bit by bit using shift registers. - Important for communication with peripherals like sensors, displays, and memory modules. - - Exercise (5 minutes) –

	Write 8051 C programming based on Data Serialization Store a constant array in **ROM** and read it. Serialize an **8-bit data** and send it through a port pin.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/4JGlqt_FYjA Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why is code memory used for storing constants? 2. How does serialization help in microcontroller communication? 3. What are real-world applications of serialization in 8051? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 16	**Course Name: Microcontroller and applications** **Topic**: **Counter/Timer Programming in Embedded C.**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept of Timer/Counter programming in Embedded C. b. Demonstrate all programming modes in Timers. c. Familiarize themselves with the registers used in Timers and Counters in the 8051 microcontroller.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What do you mean by a counter? - What is a timer, and how is it different from a counter? - Why are timers and counters important in embedded systems? - What is the need for delay in the 8051 microcontroller? - - **Development** (30 minutes) - **Introduction to Timers in 8051** - The **8051 microcontroller** has **two timers/counters**: **Timer 0 and Timer 1**. - These timers can operate in **two modes**: - **Timer Mode:** Generates precise time delays. - **Timer 0 and Timer 1 in 8051** - **Timer 0 and Timer 1** are **16-bit registers** used for timing operations. - They can be configured as **Timers (for delays)** or **Counters (for event counting)**. - **Modes of Timer Programming in 8051** - **Mode 0 (13-bit Timer Mode):** - Timer operates with **13-bit registers**. - Rarely used due to limited range. - **Mode 1 (16-bit Timer Mode):** - Uses **all 16 bits** of the timer. - Most commonly used for generating long delays. - **Mode 2 (8-bit Auto-reload Mode):** - Useful for **periodic** timing applications. - The **timer reloads automatically** after reaching overflow.

	- **Mode 3 (Split Mode):** - **Timer 0** operates as **two independent 8-bit timers.** - **Useful in dual timer applications.** - Exercise (5 minutes) – • Write assembly program to use Timer 0 and Timer 1 in 16-bit mode to generate a delay.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/4t6TTi2scWI Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why is Timer Mode 1 commonly used in 8051? 2. What is the difference between a timer and a counter? 3. How does the TMOD register control timer operations? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 17	**Course Name: Microcontroller and applications** **Topic: Counter/Timer Programming in Embedded C.**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept of Counter/Timer Programming in Embedded C. b. Demonstrate all the programming modes in Timer. c. Familiarize with the registers used in Timer and Counter.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. What do you mean by counter? What is the meaning of Timer? Highlight the significance of Timer/Counter in 8051. What is the need for delay in 8051 microcontroller? Highlight the significance of TMOD register in using timers - **Development** (30 minutes) - Finding values to be loaded into the Timer - Programs & Problems based on it - Modes of Programming in Timer - Mode 0 - Mode 1 - Programs based on Mode 0 and Mode 1 - Exercise (5 minutes) – Write a program to generate a frequency of 100KHz on Pin P2.3 using timer 1 in mode 1, with a frequency of 22MHz.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/4t6TTi2scWI Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why is Timer programming important in embedded systems? 2. Where can the knowledge of timers be used in real-world applications? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 18	**Course Name: Microcontroller and applications** **Topic: Counter/Timer Programming in Embedded C.**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept and role of counters in the 8051 microcontroller. b. Differentiate between counters and timers. c. Learn how to configure 8051 counters using the TMOD and TCON registers. d. Implement counter operations in Embedded C
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Define counters and explain their significance in embedded systems. - Discuss the difference between counters and timers: • Timers: Count clock pulses (internal events). • Counters: Count external events (e.g., pulses from a sensor). Explain how counters are used in real-world applications like frequency measurement, event counting. - **Development** (30 minutes) - Counter Registers in 8051 - TMOD Register: Configures the mode and function of timers/counters. - TCON Register: Controls start, stop, and overflow flags. - THx/TLx Registers: Store counter values. - Counter Modes - Mode 0: 13-bit counter. - Mode 1: 16-bit counter (commonly used). - Mode 2: 8-bit auto-reload counter. - Mode 3: Splits Timer 0 into two 8-bit counters - Exercise (5 minutes) – WAP Count external pulses and display the value on an LCD/7-segment display.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/4t6TTi2scWI

	Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the main difference between counters and timers in 8051? 2. How does the TCON register help in controlling counters? 3. In which mode does a counter auto-reload its value? 4. Why is Mode 1 (16-bit) preferred for counting applications? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 19	**Course Name: Microcontroller and applications** **Topic: Basics of serial communication**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Familiarize with the basics of serial communication. b. Understand the importance of serial communication.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. What is communication? Highlight the importance of communication in any embedded system What is the difference between serial and parallel communication? - **Development** (30 minutes) - Introduction to 8051 serial communication and parallel communication - Compare and contrast serial versus parallel communication - List the advantages of serial communication over parallel - Explain serial communication protocol - Contrast synchronous vs asynchronous communication - Highlight the transmission modes - Simplex - Half Duplex - Full Duplex - Explain the process of data framing - Specify the importance of start bit and stop bit in data framing - - Exercise (5 minutes) – - Ask students to provide real-world examples of **Simplex, Half-Duplex, and Full-Duplex** communication. - Encourage discussion on where these modes are used in embedded systems.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/5Dn69HZ_w7c Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the significance of serial communication? 2. Why is serial communication preferred in embedded systems over

	parallel? 3. Who uses different transmission modes in real-world applications? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 20	Course Name: Microcontroller and applications Topic: 8051 connections to RS232	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the importance of serial communication. b. Explain the RS-232 standard and its role in serial communication. c. Understand the function of MAX232 and MAX233 chips in interfacing embedded systems.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. What is the difference between serial and parallel communication? How is the connection between the embedded system established for serial communication? What are the dates rates offered in serial communication? - **Development** (30 minutes) - **Understanding Data Rate and bps (Bits per Second)** - Define **data rate** and **bps (bits per second)**. - Explain common serial communication speeds (e.g., **9600 bps, 115200 bps**). - **Introduction to RS-232 Standard** - Explain the **RS-232** standard, its purpose, and how it facilitates serial communication. - Discuss its historical significance and modern alternatives. - **RS-232 Connectors: DB25 & DB9** - **DB25 Connector:** Explain its 25 pins and their roles. - **DB9 Connector:** More commonly used, explain its pinout and functions. - Compare **DB25 vs. DB9**, highlighting why DB9 is more popular in modern systems. - **Role of MAX232 and MAX233 Chips in Serial Communication** - Explain the **voltage level conversion** between **RS-232 (±12V) and TTL logic (0–5V)**. - Importance of **MAX232 and MAX233 chips** in interfacing microcontrollers with RS-232. - Comparison: - **MAX232:** Requires external capacitors. - **MAX233:** Does not require external capacitors. - **Interfacing the 8051 Microcontroller with RS-232 Connector**

	- Explain how to connect **8051 with RS-232 using MAX232**. - Discuss the **serial communication registers in 8051 (SCON, SBUF, etc.)**. - Show a simple circuit diagram of **8051 connected to a DB9 connector via MAX232**. - - Exercise (5 minutes) – Ask students to provide summary of lecture and also emphasize - Why is **MAX232 necessary** when using RS-232? - What are the key **differences between DB9 and DB25 connectors**?
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/-UXpfAggE5Q Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. **What** is the role of RS-232 in serial communication? 2. **Why** do we need a voltage level converter (MAX232) in serial communication? 3. **Who** commonly uses RS-232 communication today? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 21	**Course Name: Microcontroller and applications** **Topic**: **Serial port operations and programming**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept of **serial port operations** in embedded systems. b. Explain the **importance of MAX232/MAX233 chips** in serial communication. c. Program the **8051 serial port in Assembly and C**.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Explain the use of MAX232 and MAX 233 chips - Interface the 8051 with a RS-232 connector - importance of using MAX 232 with RS 232 in serial communication - **Development** (30 minutes) - Discuss the baud rate of 8051 - Describe serial communication features of 8051 - Program the 8051serial port in assembly and C - Program the second serial port of DS89C4x0 in Assembly and C. - Program based on the serial ports - Exercise (5 minutes) – WAP to receive bytes od data serially and put them in P1. Set the baud rate at 4800, 8-bit data, and 1 stop bit.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/LD2Fbhk2fAI Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. **What** is the role of the serial port in 8051? 2. **Why** do we need to configure Timer 1 for baud rate control? 3. **Who** commonly uses RS-232 serial communication today? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 22	**Course Name: Microcontroller and applications** **Topic**: **Introduction to basics of 8051 interrupts**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the fundamentals of 8051 interrupts and their role in embedded systems. b. Differentiate between polling and interrupts and explain the advantages of interrupt-driven execution. c. Identify and describe the six interrupts of the 8051, including their vector addresses and priorities.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is an **interrupt** in a microcontroller? - How does **polling differ from interrupts**? - Why are interrupts essential in embedded systems? - **Explain Polling**: - A **continuous checking mechanism** where the CPU actively checks device status. - **Disadvantage**: Wastes CPU time, making it inefficient for real-time applications. - **Introduce Interrupts**: - An **event-driven mechanism** that stops the main program flow to handle high-priority tasks. - **Development** (30 minutes) - Introduction to 8051 interrupts - Contrast and compare interrupts versus polling - Explain the purpose of interrupt service routine (ISR) - List the 6 interrupts of 8051 - Explain the purpose of interrupt vector table - - Exercise (5 minutes) – Ask students to summarize the steps involved in executing an interrupt
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/7l3-iq6OtEE Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the purpose of an ISR? 2. Why is polling inefficient compared to interrupts?

	3. Who assigns interrupt vector addresses in a microcontroller? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 23	Course Name: Microcontroller and applications Topic: Types of 8051 interrupts	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the fundamentals of 8051 interrupts and their significance in embedded systems. b. Enable or disable 8051 interrupts and program the timers using interrupts. c. Describe and differentiate external hardware interrupts (INT0 & INT1), including level-triggered and edge-triggered interrupts.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - List the 6 interrupts of 8051 - Highlight the purpose of interrupt vector table - Discuss the steps involved in executing an interrupt - **Development** (30 minutes) - Understanding 8051 Interrupts - Definition and working of interrupts - How interrupts improve efficiency over polling - Enabling and Disabling Interrupts in 8051 - Explain the IE (Interrupt Enable) register. - Demonstrate enabling/disabling specific interrupts. - Timer Interrupts in 8051 - Explain how to program 8051 timers using interrupts. - Discuss applications of timer interrupts in real-time systems. - External Hardware Interrupts in 8051 - Introduce INT0 (P3.2) and INT1 (P3.3). - Explain Level-triggered vs Edge-triggered interrupts. - Explain how to configure interrupts using the TCON register. - - Exercise (5 minutes) – Ask students to write a program to generate two square waves one of 5KHz frequency at pin P1.3, and another frequency 25KHz at pin P2.3. Assume XTAL=22MHz.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/UJytXyg6wpM https://youtu.be/DPsiTe0KAlI Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss.

	What are interrupts, and why are they used? 1. How do edge-triggered and level-triggered interrupts differ? 2. How do you enable and disable an interrupt in 8051? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 24	Course Name: Microcontroller and applications Topic: Interrupt priority in 8051, Interrupt programming using embedded C	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Familiarize with the different types of interrupts in the 8051 microcontroller. b. Understand and articulate the interrupt priority in 8051, including interrupt nesting and priority assignment. c. Demonstrate the concept of interrupt programming in Embedded C by writing and analyzing programs.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is an interrupt? - How does an interrupt differ from polling? - Why are interrupts important in real-time applications? - Discuss enabling and disabling 8051 interrupts. - Explain the external interrupts INT0 (P3.2) & INT1 (P3.3). - Introduce level-triggered vs edge-triggered interrupts. - **Development** (30 minutes) - Contrast edge-triggered with level-triggered interrupts - Program th 8051 for interrupt-based serial communication - Define the interrupt priority of the 8051 - Interrupt priority upon reset - Setting interrupt priority with the IP register - Interrupt inside an interrupt - Triggering the interrupt by software - Program 8051 interrupts in C - Exercise (5 minutes) – Ask students to write a program based on interrupt programming in 8051 using embedded c.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/CSKnrL4milM Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss.

	1. What are the differences between edge-triggered and level-triggered interrupts? 2. How does the 8051 prioritize interrupts? 3. How can interrupts be used in real-time applications? 4. How do you write an interrupt-based program in Embedded C? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 25	**Course Name: Microcontroller and applications** **Topic**: **LCD Interfacing**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the working principle of an LCD. b. Explain the pin configuration and command structure of a 16x2 LCD. c. Interface a 16x2 LCD with the 8051 microcontroller. d. Write and analyze Embedded C programs for displaying text on an LCD.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is an LCD? - Where do we use LCDs in real-world applications? - Why do we need an LCD in embedded systems? - Briefly explain how an LCD functions and why it is used instead of LEDs and seven-segment displays. - **Development** (30 minutes) - **a. LCD Basics and Pin Description** - Explain the structure of a 16x2 LCD (16 characters per line, 2 lines). - Discuss the pin configuration (data pins, control pins – RS, RW, E). - Differentiate between command mode and data mode. - **b. LCD Command Structure** - Explain important LCD commands for initialization, clearing the display, cursor movement, and writing data. - **c. Interfacing LCD with the 8051** - Discuss the connection between LCD pins and the 8051 microcontroller. - Explain timing requirements and control signal operations (Enable, Read/Write, Register Select). - **d. LCD Programming in Embedded C** - Discuss the steps to write an LCD program in C. - Explain functions for initializing the LCD, sending commands, and displaying characters. - Discuss possible errors and debugging techniques in LCD programming.

	- Exercise (5 minutes) – Ask students to write an Embedded C program to display their name on the LCD.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/P6MO-n3MUAYS https://youtu.be/LtevacTk7Ww 3. Homework Write an Embedded C program to scroll a message (e.g., "WELCOME") continuously on the LCD. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What are the advantages of using an LCD over a seven-segment display? 2. How do we send commands and data to the LCD? 3. What role do the control pins (RS, RW, E) play in LCD operation? 4. How do we interface an LCD with the 8051 using Embedded C? Spend 5 minutes to evaluate student assimilation of the lesson contents

a

Lesson Plan No. 26	**Course Name: Microcontroller and applications** **Topic: LCD Interfacing**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the basics of LCD Interfacing. b. Demonstrate various examples on LCD using 8051.
Teaching Aids (if any)	a. ICT Usages
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Describe the functions of the pins of a typical LCD - List instruction command codes for programming an LCD - **Development** (30 minutes) - Interface an LCD to the 8051 - LCD Data sheet - LCD Addressing - Program an LCD in Assembly and C - Sending information to LCD using MOVC instructionss - Exercise (5 minutes) – Ask students to write a program to send letters ‘M’, ‘D’, & ‘E’ to the LCD using delays.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/P6MO-n3MUAY 3. Homework Modify the program to display two different messages alternately every 5 seconds. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What are the advantages of using an LCD over a seven-segment display? 2. How do we send commands and data to the LCD? 3. What role do the control pins (RS, RW, E) play in LCD operation? 4. How do we interface an LCD with the 8051 using Embedded C Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 27	Course Name: Microcontroller and applications Topic: Keyboard Interfacing	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the working principle of a matrix keyboard. b. Explain the pin configuration and scanning mechanism of a 4x4 keypad. c. Interface a matrix keyboard with the 8051 microcontrollers. d. Write and analyze Embedded C programs for detecting key presses.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is a keyboard? - How does a computer detect a keypress? - Where are keypads used in embedded systems? - Explain different types of keypads (e.g., mechanical, capacitive, membrane, matrix). - Introduce the concept of a matrix keypad (4x3 or 4x4 arrangement). - **Development** (30 minutes) - **a. Matrix Keyboard Basics and Pin Description** - Explain the structure of a 4x4 keypad (4 rows and 4 columns). - Discuss the working principle of a row-column scanning mechanism to detect keypresses. - **b. Interfacing a Keypad with the 8051** - Discuss how to connect the rows and columns of a keypad to the 8051 microcontroller. - Explain the process of scanning a keypad using row scanning and column detection. - **c. Keyboard Programming in Embedded C** - Discuss the logic for detecting a keypress using scanning techniques. - Explain how to write an Embedded C program to: - Continuously scan the keypad. - Detect a keypress and display the corresponding character on an LCD or serial monitor. - Implement debouncing techniques to avoid false detection - Exercise (5 minutes) – Two switches are connected to pins P0.1 & P0.2. They are also vectored to interrupt location 0003H, i.e., INT0. Write a program to test which key is pressed or to verify if both keys are pressed.

Closurse	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/dqo-PA-_a0Q https://youtu.be/f2wra_y5EQw https://youtu.be/yp-D8S-5ECk https://youtu.be/975CS27w16w 3. Homework 4. Write an Embedded C program to detect a numeric keypress (0-9) and display it on an LCD Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. How does a matrix keypad reduce the number of GPIO pins needed for key detection? 2. What is the purpose of scanning in keypad interfacing? 3. How can we implement a debounce mechanism in keypad programming? 4. What happens if two keys are pressed simultaneously in a matrix keypad? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 28	**Course Name: Microcontroller and applications** **Topic: Debouncing Effect**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept of switch bouncing and the need for debouncing. b. Explain the different methods of debouncing (software and hardware). c. Analyze the impact of debouncing on real-time embedded applications.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What happens when you press a button on a remote or keyboard? - Have you ever noticed unintended multiple key presses? - How does a microcontroller detect a keypress? - Define switch bouncing: A mechanical switch does not close smoothly but oscillates for a few milliseconds. - Explain why debouncing is necessary to ensure correct key detection. - **Development** (30 minutes) - a. Understanding the Switch Bouncing Effect - Explain the working of mechanical switches and why bouncing occurs. - Show a graphical representation of bouncing signals using an oscilloscope waveform. - - Exercise (5 minutes) – Ask students to modify a button press program to include a debouncing delay.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/yA6fAATLrMU 3. Homework Ask Students to compare hardware and software debouncing methods. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What happens if we do not debounce a switch?

	2. Which method is better: software or hardware debouncing? Why? 3. How can debouncing improve real-time applications like ATMs and keyboards? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 29	Course Name: Microcontroller and applications Topic: ADC 0804 interfacing	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Explain the concept of Analog-to-Digital Conversion (ADC). b. Understand the working principle and architecture of the ADC 0804. c. Describe the pin configuration and operation of ADC 0804. d. Identify the role of ADC in embedded systems.
Teaching Aids (if any)	a. ICT Usages
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is the difference between analog and digital signals? - Can microcontrollers directly process analog signals? - Where are ADCs used in real-life applications? - Explain that ADC converts analog signals (e.g., voltage, temperature) into digital data that a microcontroller can process. - Introduce ADC 0804 as an 8-bit ADC used in embedded applications. - **Development** (30 minutes) - a. Understanding ADC and Its Need in Embedded Systems - Explain why Analog-to-Digital Conversion (ADC) is essential in electronics. - Discuss the successive approximation method used in ADC 0804. - b. Block Diagram & Working of ADC 0804 - Explain the internal architecture and how ADC 0804 converts analog signals into digital values. - c. Pin Configuration and Functionality - Describe key pins of ADC 0804, including: - Vin+, Vin- (Analog Input & Ground) - Vref (Reference Voltage) - CS, RD, WR, INTR (Control Pins) - DB0 – DB7 (Data Output) - d. External Clock Requirement - Discuss why ADC 0804 needs an external clock (e.g., 555 Timer or software-generated clock). - Exercise (5 minutes) – Ask students to label and explain each pin of ADC 0804.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 2. Suggested Video https://www.youtube.com/watch?v=UODCHRdyb8Y https://youtu.be/yA6fAATLrMU 3. Homework Ask students to find real-world applications of ADCs in embedded systems. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What happens if the clock signal is missing? 2. How does ADC 0804 generate digital output? 3. What factors affect ADC accuracy? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 30	**Course Name: Microcontroller and applications** **Topic**: **ADC 0804 interfacing**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Interface ADC 0804 with the 8051 microcontroller. b. Write an Embedded C program to read analog data from ADC 0804. c. Analyze the role of interrupts and control signals in ADC 0804.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - How does a microcontroller **read sensor values**? - What steps are involved in reading analog data using ADC 0804? - **Development** (30 minutes) - a. Circuit Connections for ADC 0804 Interfacing - Explain how to connect ADC 0804 to the 8051, including: - Analog input source (e.g., potentiometer or sensor). - Control pins (WR, RD, CS, INTR). - Data bus (DB0 – DB7) connections to the microcontroller. - b. Steps for Analog-to-Digital Conversion - Provide an analog input. - Start conversion by setting WR = LOW. - Wait for INTR to go LOW (conversion complete). - Read digital output using DB0 – DB7. - c. Writing Embedded C Code for ADC 0804 - Explain the logic for programming ADC 0804 in Embedded C. - Exercise (5 minutes) – Ask students to practice programs based on ADC 0804
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/yA6fAATLrMU Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What happens if the INTR pin is not checked before reading data? 2. How does changing Vref affect ADC readings? 3. How can ADC be used in real-time applications Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 31	**Course Name: Microcontroller and applications** **Topic**: DAC **0808 interfacing**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Familiarize with the concept of DAC 0808. b. Understand the basics of DAC 0808 interfacing. c. Articulate the programming of interfacing DAC with 8051.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is the difference between **digital** and **analog** signals? - Can microcontrollers directly generate **analog outputs**? - Where are **DACs used** in real-life applications? - Explain that **DAC converts digital signals (binary data) into analog signals (e.g., voltage, current, sound)**. - Introduce **DAC 0808** as an 8-bit DAC commonly used in embedded applications. - **Development** (30 minutes) - a. Understanding DAC and Its Need in Embedded Systems - Explain why Digital-to-Analog Conversion (DAC) is necessary for systems like audio processing, motor control, and waveform generation. - Discuss different types of DACs (Binary-Weighted DAC, R-2R Ladder DAC) and explain why DAC 0808 uses an R-2R ladder network. - b. Block Diagram & Working of DAC 0808 - Explain the internal architecture and how DAC 0808 converts digital data into an analog voltage output. - Discuss the reference current (Iref) and how it affects the analog output. - c. Pin Configuration and Functionality - Describe key pins of DAC 0808, including: - D0 – D7 (8-bit digital input). - Vref+ and Vref- (Reference voltage pins). - Iout (Analog Current Output). - GND & Vcc (Power Supply Pins). - d. Output Characteristics - Explain how DAC 0808 provides an analog current output (Iout) and requires an operational amplifier (Op-Amp) to convert it into voltage output (Vout). - - Exercise (5 minutes) –

	Ask students to label and explain each pin of DAC 0808.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/yFIiJ6bJUNE https://youtu.be/M_vXUBX3-nM 3. Homework Ask students to find real-world applications of DACs in embedded systems. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the difference between **ADC and DAC**? 2. Why does DAC 0808 output **current instead of voltage**? 3. How can we convert the **current output (Iout) to voltage**? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 32	**Course Name: Microcontroller and applications** **Topic: Sensor and actuator interfacing**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Define sensors and actuators and explain their functions in embedded systems. b. Classify different types of sensors (analog and digital) and actuators. c. Interface common sensors and actuators with a microcontroller. d. Write and analyze an Embedded C program to interface sensors and actuators.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are Sensors? - What are actuators? - What is the difference between the two? - Where have you seen **sensors and actuators** used in real life (e.g., home automation, robotics, medical devices)? - - **Development** (30 minutes) - Introduction to Sensor and actuator - Difference between Sensor and actuator - **Analog Sensors** (Continuous output): LM35 (Temperature), LDR (Light), Potentiometer. - **Digital Sensors** (Discrete HIGH/LOW output): IR Sensor, Ultrasonic Sensor, PIR Motion Sensor.S - **Actuators:** DC Motor, Servo Motor, Relay, Solenoid, Buzzer. - Temperature sensor - LM34 & LM35 Temperature Sensor - Explain the function of precision IC temperature sensors - Interfacing of different sensors with 8051 (LM35) - Reading and displaying temperature - Exercise (5 minutes) – Ask students to practice programs based on Temperature sensor using assembly langauge
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/jrjMQ_GW3-c https://youtu.be/eQLYQIkm1_4 3. Homework

	Explain the difference between analog and digital sensors. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why do some sensors need an ADC while others work with digital pins? 2. How do actuators receive enough power from a microcontroller? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 33	Course Name: Microcontroller and applications Topic: Speed and direction control of DC	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the working principles of DC motors b. Explain different techniques for controlling speed and direction of this motors. c. Interface DC motor with a microcontroller and analyze an Embedded C program for motor control using PWM and driver circuits.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	• **Introduction** (5 minutes) • Ask questions. How does a microcontroller control a motor? • Why is motor speed and direction control important in automation? • **Developmsent** (30 minutes) • Introduction to DC Motor\DC Motor Speed and Direction Control • Direction Control: • Unidirectional Control • Bidirectional Control • Pulse Width Modulation (PWM) • Use of H-Bridge circuit (L293D / L298N motor driver) to control motor rotation direction. • Changing logic levels of the driver inputs determines forward/reverse rotation. • Speed Control: ○ Use Pulse Width Modulation (PWM) to vary motor speed. ○ Higher duty cycle → faster speed, Lower duty cycle → slower speed. DC Motor: ○ Connect via H-Bridge motor driver (e.g., L293D, L298N). ○ Control speed using PWM output from the microcontroller. • Exercise (5 minutes) – Ask students to writing an Embedded C Program for controlling DC motor speed with PWM.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://www.youtube.com/watch?v=iJCTMoQ4e6c&pp=ygUyc3

	BlZWQgYW5kIGRpcmVjdGlvbiBjb250cm9sIG9mIERDIE1vdG9ycyBXSVRIIDgwNTE%3D Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. How does PWM control the **speed** of a DC motor? 2. Why is PWM (Pulse Width Modulation) used for speed control instead of varying voltage directly? 3. How does changing the duty cycle of a PWM signal affect the speed of a DC motor? 4. What are the advantages and disadvantages of using a DC motor in embedded systems? 5. What are some real-world applications where precise DC motor speed control? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 34	**Course Name: Microcontroller and applications** **Topic**: **Speed and direction control of Stepper Motors**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Explain the working principle and types of stepper motors. b. Demonstrate speed and direction control of a stepper motor using a microcontroller. c. Write and analyze an Embedded C program for stepper motor control.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	• **Introduction** (5 minutes) • Ask questions. • What is a stepper motor, and how does it differ from a DC motor? • Where are stepper motors used in real-world applications? • Why is precise control of speed and direction important in stepper motors? • **Development** (30 minutes) • Introduction to stepper Motor • Types of Stepper Motors • Unipolar Stepper Motor (uses center-tapped coils, easier control). • Bipolar Stepper Motor (higher torque, requires H-Bridge). b. Stepper Motor Control Basics • Direction Control: o Changing the sequence of coil activation reverses the motor direction. • Speed Control: o Adjusting the time delay between step pulses controls the speed. • Step angle • Steps per revolution • Tooth pitch • Rotation speed • RPM • Motor speed • Stepper motor control with 8051 C • Programming of stepper Motor • Interfacing of motors with 8051 • Exercise (5 minutes) – Ask students to practice programs based on Stepper Motor and its speed and direction controls

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/EOAXox9XzTI 3. Homework Compare Unipolar and Bipolar stepper motors in terms of structure and control. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why do stepper motors require a driver circuit instead of direct microcontroller control? 2. How does the stepping sequence affect motor movement and torque? 3. What are the advantages and disadvantages of stepper motors compared to D Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 35	**Course Name: Microcontroller and applications** **Topic**: **Introduction to Arduino**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the basics of Arduino and its components. b. Demonstrate basic programming of Arduino using Arduino IDE. c. Introduce Arduino and its significance in electronics and robotics.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What is Arduino? - Why is Arduino popular among professionals? - What are some common projects you can create with Arduino? Introduction to Arduino IDE Basic structure of an Arduino program (setup and loop functions) - **Development** (30 minutes) - Introduction to Arduino - History and evolution of Arduino - Types of Arduino boards (Uno, Nano, Mega, etc.) - Basic Electronics Concepts (5 minutes) - Voltage, current, and resistance - Breadboard and circuit connections - Basic Components of Arduino - Introduction to Arduino development board - Why Arduino? - Different types of Arduino boards - Different types of Arduino shields - Exercise (5 minutes) – Ask students to modify the Blink LED program to change the blinking speed.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/H9OEAn3Uc2w 3. Homework Write an Arduino program to control an LED with a push button Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. Why is Arduino preferred for beginners in electronics?

	2. How does the Arduino IDE simplify programming? 3. What real-world applications can you develop using Arduino? 4. Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 36	Course Name: Microcontroller and applications Topic: Programming Tools and Programming language for development board	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the various programming tools available for development boards. b. Demonstrate basic programming using suitable language for a development board.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are development boards used for? - Can you name some programming languages used for these boards? - Why is it important to choose the right tools for development? - **Development** (30 minutes) - Introduction to Programming Tools - Detailed explanation of IDEs (e.g., Arduino IDE, Thonny). - Advantages of using specific tools (ease of use, community support). - Setting Up Development Environment - Step-by-step installation guide for Arduino IDE. - Basic configuration for different development boards. - Writing Your First Program - Example 1: Blinking an LED on Arduino using Arduino IDE. - Exercise (5 minutes) – Ask students to write a simple program using their chosen tool and language, such as blinking an LED
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/Ab9U7NQB1kA https://youtu.be/SX8z3-BEuWQ?list=PLwWF-ICTWmB7-b9bsE3UcQzz-7ipI5tbR 3. Homework Compare Arduino IDE and PlatformIO in terms of features. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss.

	1. Why is C/C++ preferred for microcontrollers? 2. How does Python simplify embedded programming? 3. Which development board would you choose for an IoT project and why? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 37	Course Name: Microcontroller and applications Topic: Interfacing with LDR's,	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the differences between analog and digital sensors. b. Demonstrate the interfacing of analog sensors (LDR) with Arduino. c. Write a program to read data from an LDR and perform basic data processing
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are sensors used for? - Can you name some analog and digital sensors? - Overview of sensors in electronics and their importance - Difference between analog and digital sensors. - Analog: Continuous signals (e.g., LDR, LM35). - Digital: Discrete signals (e.g., IR sensor, PIR sensor). - Why is it important to differentiate between analog and digital sensors? - **Development** (30 minutes) - **A. Understanding LDR (Light Dependent Resistor)** - **Type**: Analog sensor. - **Working Principle**: As light increases, resistance decreases, voltage across it changes — which is read by Arduino as an analog value. - **Applications**: - Street lights that switch ON/OFF automatically. - Light meters. - Automatic brightness control in devices. - - **B. Circuit Diagram and Hardware Connections** - **Components Required**: - Arduino UNO, LDR, 10kΩ resistor, Breadboard, Jumper wires. - Circuit Description - Connect one end of the LDR to 5V. - Connect the other end to one leg of the 10kΩ resistor and to analog pin A0. - Connect the other leg of the resistor to GND. - The junction between the LDR and resistor is the voltage divider output, which goes to **A0**. - Explain voltage divider principle used to convert resistance changes into voltage changes. -

	- - Exercise (5 minutes) – Ask students to write a program to read LDR sensor values and display them on the Serial Monitor.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/nE1C4ghfvac?list=PLgMDNELGJ1CbufZjqWa8uoSlQWKqVwPN7 https://youtu.be/CFL3E0or2HU 3. Homework Write an Arduino program to turn an LED ON when it’s dark and OFF when it's bright, using data from an LDR. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the working principle of an LDR and how does it behave under different lighting conditions? 2. Why do we use a resistor with the LDR while connecting it to the Arduino? 3. How does the voltage divider help convert changes in light intensity into readable analog values? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 38	**Course Name: Microcontroller and applications** **Topic**: **Interfacing with Temperature Sensors**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the characteristics of analog temperature sensors. b. Demonstrate the interfacing of LM35 with Arduino. c. Write a program to read and convert analog temperature data into Celsius using Arduino.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Why do we need to measure temperature in electronics or automation? - Can you name some temperature sensors used in industries or homes? - Applications of temperature sensors: weather monitoring, industrial systems, smart homes. - Analog vs Digital Temperature Sensors: LM35 (Analog), DHT11 (Digital). - Why is the LM35 a preferred choice for basic temperature sensing? - **Development** (30 minutes) - **Understanding the LM35 Sensor** - **Specification Highlights**: - Output: 10 mV per °C - Range: -55°C to 150°C - Accuracy: ±0.5°C - Working Principle: - For every 1°C rise, the output voltage increases by 10 mV. - Example: 25°C → 250 mV output. - **B. Circuit Diagram and Connections** - Components Required: - Arduino UNO, LM35 Sensor, Breadboard, Jumper Wires - LM35 Pinout: - Pin 1 (Vcc) → 5V - Pin 2 (Vout) → A0 - Pin 3 (GND) → GND - - **Exercise (5 minutes)** – Ask students to write an Arduino program that reads data from LM35 and displays the temperature on the Serial Monitor.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/CFL3E0or2HU 3. Homework Write an Arduino program that reads temperature from LM35 and controls a fan (use an LED as a placeholder) when the temperature exceeds 28°C. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the output behaviour of LM35, and how is it mapped to temperature in °C? 2. Why do we use analogRead() for LM35, and how do we convert its readings to human-readable temperature values? 3. How does sensor resolution (0–1023) affect the accuracy of temperature readings? 4. What are some potential sources of error or noise in temperature measurements, and how can you minimize them in a real-world system? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 39	Course Name: Microcontroller and applications Topic: Interfacing Temperature Sensor (DHT-11)	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the characteristics and working of digital temperature and humidity sensors. b. Demonstrate the interfacing of DHT11 with Arduino. c. Write and execute a program to read temperature and humidity data from DHT11 and display it on the serial monitor.
Teaching Aids (if any)	a. PowerPoint Presentation
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are the limitations of analog sensors like LM35? - How are digital sensors different from analog ones in signal processing? - Applications of temperature and humidity monitoring: weather stations, greenhouses, smart homes. - Introduction to DHT11 sensor as a digital environmental sensor. - Brief comparison: DHT11 vs LM35. - **Development** (30 minutes) - **A. Understanding the DHT11 Sensor** - Specification Highlights: - Temperature Range: 0–50°C (±2°C accuracy) - Humidity Range: 20–90% RH (±5% accuracy) - Interface: Digital single-wire communication - Output format: Integer values (°C and % RH) - - **B. Circuit Diagram and Connections** - Components Required: - Arduino UNO, DHT11 Sensor Module (or raw sensor), 10kΩ resistor (for raw DHT11), Breadboard, Jumper Wires - DHT11 Pinout: - Vcc → 5V - Data → Digital pin (e.g., D2) - GND → GND - - Exercise (5 minutes) – Write a program that reads temperature and humidity from the DHT11 and displays both values on the Serial Monitor every 2 seconds.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 2. Suggested Video https://youtu.be/CFL3E0or2HU 3. Modify the DHT11 Arduino code to display temperature and humidity on LCD display instead of the Serial Monitor. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. **What** is the key difference in data output between the DHT11 and LM35 sensors? 2. **Why** is the DHT11 considered a digital sensor even though it senses analog quantities like temperature and humidity? 3. **How** does using a digital sensor simplify signal processing and reduce error compared to analog sensors? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 40	Course Name: Microcontroller and applications Topic: Interfacing with Ultrasonic Sensor	Course No.: ECE-401

Objectives	At the end of the lesson the student shall be able to: a. Understand the working principle of ultrasonic sensors. b. Demonstrate the interfacing of an ultrasonic sensor (HC-SR04) with Arduino. c. Write a program to measure distance using HC-SR04 and display the output on Serial Monitor.
Teaching Aids (if any)	a. PowerPoint Presentation
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are the methods for measuring distance in electronics? - Have you heard of ultrasonic or sonar-based distance measurement? - Applications: Parking systems, robotics, obstacle detection. - Introduction to HC-SR04: A commonly used ultrasonic distance sensor. - What physical principle does it use? *(Echo and time-of-flight measurement)* - **Development** (30 minutes) - **A. Understanding the HC-SR04 Sensor** - Working Principle - Calculates distance using the formula - Specifications: - Operating Voltage: 5V - Range: 2cm to 400cm - Accuracy: ±3mm - Pins: Vcc, Trig, Echo, GND - - **B. Circuit Diagram and Connections** - Required Components: - Arduino UNO, HC-SR04 Sensor, Breadboard, Jumper Wires - Pin Connections: - Vcc → 5V - GND → GND - Trig → Digital pin (e.g., D9) - Echo → Digital pin (e.g., D10) - Exercise (5 minutes) – Ask students to write a program to read the distance using HC-SR04 and turn ON an LED if the object is detected within 20 cm.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/CFL3E0or2HU 3. Homework Modify the program to sound a buzzer if an object is detected within 10 cm. Use the tone() function in Arduino. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the principle behind ultrasonic distance measurement? 2. Why do we divide the result by 2 when calculating distance using the speed of sound? 3. How does ambient noise affect the accuracy of ultrasonic sensors? 4. What are the advantages and limitations of using HC-SR04 in robotics applications? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 41	**Course Name: Microcontroller and applications** **Topic: Interfacing with IR Sensor**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the working principle of an IR (Infrared) sensor. b. Demonstrate the interfacing of an IR sensor with an Arduino. c. Write a program to detect objects using an IR sensor and trigger appropriate outputs (like LEDs or buzzers).
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - How do automated doors detect motion? - What is an IR beam and where is it used in daily life? - Applications of IR sensors: obstacle detection in robots, automation, object counters. - Difference between **active** and **passive** IR sensors. - Working principle: IR transmitter emits, receiver detects reflection. - **Development** (30 minutes) - **A. Understanding the IR Sensor** - Working Principle - The output is digital: - LOW (0) if object is detected (reflection present) - HIGH (1) if no object is detected - Specifications: - Operating Voltage: 3.3V–5V - Output: Digital - Detection Range: ~2–30 cm - Adjustable sensitivity via onboard potentiometer - **B. Circuit Diagram and Connections** - Required Components: - Arduino UNO, IR Sensor Module, LED (for output), Breadboard, Jumper Wires - **Pin Connections**: - IR Vcc → Arduino 5V - IR GND → Arduino GND - IR OUT → Arduino Digital Pin (e.g., D2) - - Exercise (5 minutes) – - Ask students to write a program to count the number of times an object is detected using an IR sensor and display the count

	on the Serial Monitor. s
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/uku6PpRdt4w?si=fVTBsuFQvCdkIIGO. 3. Homework Write a program to turn ON a buzzer whenever the IR sensor detects an object. Use pin 8 for the buzzer. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What physical principle does the IR sensor use to detect objects? 2. Why is the IR output digital and not analog in this module? 3. What factors affect the range and accuracy of IR sensors? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 42	**Course Name: Microcontroller and applications** **Topic**: **Interfacing with display devices**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the working principle of basic display devices. b. Demonstrate interfacing of LED, 7-segment display, and 16x2 LCD with Arduino. c. Write Arduino programs to control these display devices.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Where do we use displays in electronics and everyday appliances? - Can you differentiate between LED, 7-segment, and LCD displays? - Displays are used in appliances, meters, and consumer electronics for visual feedback. - Common types include LED, 7-segment displays, and LCD modules. - Importance in user interfaces, status indication, and data monitoring. - **Development** (30 minutes) - **A. Interfacing an LED Display** - LEDs are the most basic form of visual output and are used to indicate on/off states. - A single LED is interfaced using one digital output pin of Arduino. - A current-limiting resistor is necessary to prevent damage to the LED. - Applications include indicators, alarms, and test/debugging signals. - **B. Interfacing 7-Segment Display** - A 7-segment display consists of seven LEDs (labeled a to g) arranged to display numbers. - It can be either **common cathode** or **common anode** based on connection type. - Each segment is controlled by a separate Arduino pin. - By turning on specific segments, we can display numbers from 0 to 9 and some alphabets. - Used in digital clocks, counters, and numeric displays. - - **C. Interfacing 16x2 LCD Display** - A 16x2 LCD can display 16 characters per row and has 2 rows.

	- It uses an alphanumeric character display to show text and numbers. - Operates in 4-bit or 8-bit mode; commonly 4-bit mode is preferred to save pins. - Requires connections to RS (Register Select), EN (Enable), and four data pins (D4 to D7). - A potentiometer is connected to adjust screen contrast. - The LCD is initialized using specific commands and can show sensor readings, messages, etc. - Very useful in embedded projects for real-time data display and system feedback. - - Exercise (5 minutes) – - Ask students to draw the circuit diagram and write logic steps (not code) to display their name on a 16x2 LCD while blinking an LED on another pin.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/uku6PpRdt4w?si=fVTBsuFQvCdkIIGO. 3. Homework Design a circuit and outline the steps to display a 4-digit counter (from 0000 to 9999) using either a 4-digit 7-segment display or simulate the values on a 16x2 LCD. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the primary difference between a 7-segment display and a 16x2 LCD in terms of functionality and data display? 2. Why is a resistor necessary when using LEDs in display circuits? 3. What are the advantages of using a 16x2 LCD in Arduino projects? 4. How would you interface a sensor with an LCD to display real-time values? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 43	**Course Name: Microcontroller and applications** **Topic**: **Interfacing with actuators**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Articulate the working principle of actuators. b. Classify different types of actuators used in embedded systems. c. Demonstrate how to interface actuators (e.g., DC motor, servo motor, relay) with Arduino. d. Design and describe the basic logic flow for actuator control using Arduino.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Have you seen an automatic door open or a robot arm move? What makes them move? - Ask: What is the difference between a sensor and an actuator? - Actuators convert electrical signals into physical motion or force. - Common types include motors, servos, relays, solenoids. - Used in automation, robotics, home appliances, automotive systems. - **Development** (30 minutes) - **A. Definition of Actuators** - Types of motion: Rotational (motors) and linear (solenoids, linear actuators). - Examples: DC motors, Servo motors, Stepper motors, Relays, Solenoids. - **B. DC Motor Interfacing** - A DC motor converts electrical energy into continuous rotational motion. - Cannot be directly connected to Arduino due to current limitations. - Requires a driver (e.g., L293D, L298N) to control direction and speed. - Applications: Fans, wheeled robots, motorized arms. - **C. Servo Motor Interfacing (7 minutes)** - Servo motors rotate to specific angles, controlled via PWM (Pulse Width Modulation). - Commonly used in robotic arms, camera systems, and model aircraft. - Operates on control signal, power, and ground connections.

	- Angle control is precise (0° to 180° typical range). - - **D. Relay Module Interfacing** - Used to turn on/off AC appliances like lights and fans. - Contains a coil, switch, and opto-isolator in modules for Arduino safety. - Ensures electrical isolation between Arduino and high-power circuits. - - **E. Safety and Best Practices** - Use protective diodes with inductive loads (motors). - Use external power supplies when needed. - Always check voltage/current ratings of actuators. - Implement safeguards in software to avoid continuous load/stall conditions. - - Exercise (5 minutes) – - Ask students to write a program to interface with an actuator and control its operation.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/uku6PpRdt4w?si=fVTBsuFQvCdkIIGO. 3. Homework Students will identify which actuator (DC motor, servo, or relay) is suitable for the following applications and briefly describe the interfacing approach: Automatic fan control Robotic arm movement Home automation light control Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. **What** is the main functional difference between a DC motor and a servo motor in terms of control and motion? 2. **Why** can't actuators like DC motors be directly powered by Arduino? 3. **What** precautions must be taken while using a relay module to control high-voltage appliances? 4. **How** does the role of an actuator complement that of a sensor in an embedded system? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 44	**Course Name: Microcontroller and applications** **Topic: Serial Communication**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the concept and purpose of serial communication. b. Explain the working of UART-based serial communication in Arduino. c. Demonstrate the setup and use of serial communication between development boards and other devices.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - Overview of serial communication. - Importance of serial communication in embedded systems. What is serial communication? Can you name some common protocols used for serial communication? Why is serial communication important in embedded systems? - **Development** (30 minutes) - Introduction to Serial Communication Setup - Explanation of hardware connections and configurations. - Overview of serial communication libraries in popular development environments (e.g., Arduino IDE). - UART Communication Setup and Programming - Detailed steps to set up UART communication. - Code example for sending and receiving data using UART. - SPI Communication Setup and Programming - Detailed steps to set up SPI communication. - Code example for sending and receiving data using SPI. - I2C Communication Setup and Programming - Detailed steps to set up I2C communication. - Code example for sending and receiving data using I2C. - - Exercise (5 minutes) – - Ask students to write a program to set up serial communication and exchange data between two devices.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/8GX5brSZ_1E?si=Dib8YS5xkt4o8DYD 3. Homework Design a system where a user sends a command (‘1’, ‘2’, or ‘3’)

	from the Serial Monitor, and Arduino responds with a specific message or status for each command. Write the flow of logic or block diagram. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the role of baud rate in serial communication, and what happens if the rates don’t match? 2. Why is serial communication preferred for debugging Arduino-based systems? 3. How can serial communication be used for both sending and receiving data? 4. What are the potential limitations of using serial communication in real-time systems? Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 45	**Course Name: Microcontroller and applications** **Topic**: **Wireless Communication using RF Module**	**Course No.: ECE-401**

Objectives	At the end of the lesson the student shall be able to: a. Understand the principles of wireless communication using RF modules. b. Describe the working of RF transmitter and receiver modules. c. Demonstrate the interfacing of RF modules with Arduino.
Teaching Aids (if any)	a. ICT Usage
Teaching Development	- **Introduction** (5 minutes) - Ask questions. - What are RF modules? - Can you name some applications of RF modules? - Overview of wireless communication using RF (Radio Frequency) modules. - Importance of RF modules in remote control, telemetry, and sensor networks. - Why is wireless communication important in modern electronics? - Basics of RF Communication - Definition and key concepts (transmitter, receiver, frequency, modulation). - Types of RF modules (transmitter, receiver, transceiver). - - **Development** (30 minutes) - Introduction to RF Module Setup - Explanation of hardware connections for RF modules. - Overview of setup procedures in popular development environments. - Transmitter Setup and Programming - Detailed steps to set up a transmitter RF module. - Code example for transmitting data wirelessly using RF. - Receiver Setup and Programming - Detailed steps to set up a receiver RF module. - Code example for receiving data wirelessly using RF. - Wireless Communication Setup - Integration of transmitter and receiver modules for bidirectional communication. - Testing and troubleshooting tips. - - Exercise (5 minutes) – - Ask students to write a program to establish wireless communication between two devices using RF modules.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Video https://youtu.be/s_G3S0t9Iyo?si=uuVQpuwA9drUITyV 3. Homework Ask students to write down the logical flow (block diagram or step-by-step process) of sending a signal wirelessly from one Arduino using an RF transmitter to another using an RF receiver to turn on an LED. Spend 5 minutes to wrap up and consolidate the learnings
Evaluation	Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 1. What is the primary advantage of using RF modules over wired communication in embedded systems? 2. How does interference from other RF devices affect the reliability of communication? 3. What are some use-cases where RF is preferred over other wireless methods like Bluetooth or Wi-Fi? Spend 5 minutes to evaluate student assimilation of the lesson contents