GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Bachelor of Engineering

Level: UG

Branch: Biomedical Engineering

Subject Code : BE04003031

Subject Name : Microcontroller Programming & Interfacing

w. e. f. Academic Year:	2024-25
Semester:	4
Category of the Course:	PCC

Prerequisite:	Fundamentals of Digital Electronics, Basic Electronics, C Programming
Rationale:	This course equips students with essential knowledge in embedded systems, balancing theory with practical applications. It prepares them for industry demands, fosters interdisciplinary skills, and encourages innovation. Including topics like architecture, programming, and advanced interfaces ensures relevance to IoT, robotics, and automation. Project-based learning and lab sessions enhance engagement, bridging the gap between academic concepts and real-world problem-solving, readying students for future technological challenges.

Course Outcome:

After Completion of the Course, Student will be able to:

No	Course Outcomes	RBT Level
01	Define and explain the fundamental concepts of microprocessors, microcontrollers, architectures, and programming environments.	R,U
02	Illustrate the architecture, features, and memory organization of AVR microcontrollers and interpret their pin functions and fuse bits.	R,U,A
03	Develop and execute Embedded C programs for basic I/O operations such as LED blinking, switch interfacing, and seven-segment display control.	R,U,A
04	Implement timer, interrupt, and serial communication-based programs to achieve event-driven and data transfer operations in embedded systems.	R,U,A,N
05	Design and integrate embedded system applications by interfacing peripherals and controlling motors using ATmega32.	R,U,A,N,E,C

**Revised Bloom's Taxonomy (RBT)*

Teaching and Examination Scheme:

Teaching-Learning Scheme (in Hours per Semester)					Total Credits = TH/30	Assessment Pattern and Marks					Total Marks
						Theory		Tutorial/Practical			
L	T	P	PBL*	TH		ESE (E)	PA (M)	PA (I)	PBL (I)	ESE (V)	
60	0	30	30	120	04	70	30	20	30	50	200

Course Content:

Unit No.	Content	No. of Hours	% of Weightage
1.	Definition of Microprocessor and Microcontroller, Difference between Microprocessor and Microcontroller, Criteria for Choosing Microcontroller, Von Neumann vs Harvard architecture, RISC vs CISC, Programming Language - Machine, Low and High-Level Language, Compiler, Assembler, Debugger, IDE	6	10
2.	Overview of AVR family, Block diagram of AVR Microcontroller, Features of ATmega32 MCU, Pin Diagram of ATmega32, Architecture of ATMega32, Memory Organization of ATmega32, Oscillator and Reset Circuit, AVR Fuse Bits	12	20
3.	Embedded C, Difference between C and Embedded C, Key Characteristics of Embedded C, Structure of Embedded C Program, Standard Embedded C Data Types, Types of Operators in C, Control statements decision making, Embedded C Programming - I/O Programming, LED, Switch and Seven Segment Programming	10	10
4.	Timer Programming in C, Interrupt Programming in C, Parallel vs. serial Data Transfer, Asynchronous vs Synchronous Data transfer, Serial Communication Protocols – SPI, I^2C, USART/UART, Serial Programming	12	20
5.	LCD Interfacing, ADC Interfacing, Temperature Sensor Interfacing, Relay and Optoisolator Interfacing, Keyboard Interfacing	12	20
6.	Working of Electric DC Motor - Brushed DC Motor, Brushless DC Motor, Stepper Motor, Servo Motor, H- Bridge, Programming – DC, Stepper and Servo Motor	8	20
	Total	**60**	**100**

Suggested Specification Table with Marks (Theory):

Distribution of Theory Marks (in %)					
R Level	**U Level**	**A Level**	**N Level**	**E Level**	**C Level**
10%	20%	30%	20%	15%	5%

Where R: Remember; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create (as per Revised Bloom's Taxonomy)

References/Suggested Learning Resources:

(a) Books:

1. AVR Microcontroller and Embedded Systems: Using Assembly and C by Muhammad Ali Mazidi, Sarmad Naimi, and Sepehr Naimi
2. Arduino Robotics by John David warren, josh Adams, Harald Molle
3. Programming and Customizing the AVR Microcontroller, By Dhananjay Gadre, McGraw Hill Education
4. Embedded C Programming and the Atmel AVR by Richard H. Barnett, Larry D. O'Cull, and Sarah Cox

(b) Open-source software and website:

1. AVR Studio Tutorials: Official Microchip website for AVR Studio and ATmega32 resources.
2. Tutorials Point: Offers tutorials on C programming and embedded systems.
3. NPTEL Lecture series - Microprocessors and Microcontrollers by Prof. Santanu Chattopadhyay (IIT Kharagpur)

Suggested Course Practical List:

1. **LED Interface:** Write a program to toggle an LED using an ATmega32 microcontroller.
2. **Switch Interface**: Detect switch presses and execute different operations (e.g., toggle LEDs or control buzzer).
3. **Seven-Segment Display**: Display numbers (0–9) on a seven-segment display by programming ATmega32.
4. **Timer to Create Delay**: Use Timer0 to generate a 1-second delay using normal mode.
5. **PWM Generation**: Generate a PWM signal to control the brightness of an LED or motor speed.
6. **Interrupt Handling**: Program an ATmega32 to handle external interrupts triggered by switches.
7. **Serial Communication**: Implement UART communication between an ATmega32 and a PC (e.g., sending data and receiving feedback).
8. **LCD Interfacing**: Display a custom message on an LCD using ATmega32.
9. **ADC with Temperature Sensor**: Interface a temperature sensor (e.g., LM35) to ATmega32 and display the converted temperature on an LCD.
10. **DC Motor Control**: Use PWM to control the speed of a DC motor connected to ATmega32.

11. **Stepper Motor Control**: Implement a program to rotate a stepper motor at specific angles or sequences.
12. **Relay Interface**: Write a program to control electrical appliances using relays connected to ATmega32.
13. **Keyboard Interfacing**: Detect key presses from a 4x4 matrix keyboard and display the corresponding character on an LCD.

List of Laboratory/Learning Resources Required:

Suggested Components:

1. ATmega32 Development Board: Essential for programming and interfacing peripherals.
2. Power Supply Unit (5V DC): Reliable power source for the development board and peripherals.
3. LEDs and Resistors: For basic LED control and interfacing experiments.
4. Push Buttons/Switches: For switch interfacing and interrupt handling.
5. Seven-Segment Display (Common Cathode/Anode): To display numbers and test segment driving programs.
6. 16x2 LCD Module: For displaying messages, temperature readings, and keyboard inputs.
7. Sensors (e.g., Temperature Sensor): For ADC interfacing and sensor data visualization.
8. DC Motor and Driver IC (e.g., L293D): To control the speed of a DC motor using PWM.
9. Stepper Motor and Driver: For precise movement control experiments.
10. Relay Module: To control high-voltage electrical appliances.
11. Matrix Keypad (4x4): For interfacing and detecting key presses.
12. USB to UART Converter Module: For serial communication with a PC.
13. Oscilloscope/Logic Analyzer: To visualize and debug PWM signals and serial communication.
14. Prototyping Tools: Breadboard, jumper wires, and connectors.

Suggested Software/resources:

1. Datasheets: Datasheets for ATmega32, L293D, sensors, and other components.
2. AVR Studio/Atmel Studio: IDE for writing, compiling, and debugging programs.
3. Programmer (e.g., USBasp or AVRISP): For burning programs into the ATmega32.
4. Proteus Simulation Software: To simulate hardware circuits before implementation.
5. Serial Monitor Tool: To monitor UART communication.

Suggested Project List:

1. **Digital Thermometer**: Interface an LM35 temperature sensor with ATmega32 to display temperature readings on an LCD.
2. **Home Automation System**: Control electrical appliances using relays and implement UART communication for remote control via a PC.
3. **Electronic Voting Machine**: Use a 4x4 matrix keypad for input and an LCD to display the result, with additional security features like a reset switch.
4. **Smart Traffic Light Controller**: Program ATmega32 to simulate a traffic light system using LEDs and timers.
5. **DC Motor Speed Controller**: Use PWM signals to control the speed of a DC motor, with feedback displayed on an LCD.

6. **Obstacle Avoiding Robot**: Use ultrasonic sensors to detect obstacles and program ATmega32 to control a motor for navigation.
7. **Temperature-Controlled Fan**: Automatically control the speed of a fan based on temperature readings from the LM35 sensor using PWM.

List of Problem Based Suggested Activities for Students:

Sr. No.	Name of the Activity	No. of Hours	Evaluation Criteria
1	Mini-Project	10 hours	Working/Summary report / Viva
2	Hands-On Coding Challenges/ Peer-to-Peer Learning	5 hours	Simulation/ Preparation and presentation skills.
3	Practical Simulation Exercises	5 hours	Simulation screenshots and output log submission
4	Online Technical Quizzes/ Group Discussions and Presentations	5 hours	Based on quiz scores/Preparation and presentation skills.
5	Troubleshooting Assignments	5 hours	Based on the depth of the solution submitted.

Note:

1. All the suggested activities should be related to the subject.
2. Subject coordinator shall identify activities from above list as per the subject needs, they also declare list of activities wise hours, evaluation scheme and rubrics to students at the start of semester.
3. The number of hours is suggestive. Faculty can sub-divide the number of hours based on the activity. However, total number of hours is fixed.
4. All records pertaining to the evaluation and assessment of self-learning activities must be properly maintained and preserved at the institute level. These records should be made available to the university upon request.
5. Institutes are encouraged to utilize digital platforms, such as Microsoft Teams, for effective record-keeping and to ensure transparency in the evaluation and assessment of self-learning activities.

* * * * * * *