USN | | | | | | | | | | |

VTU QP Solution – Dec 2023-Jan 2024

Sub:	Internet of Things and WSN	Sub Code:	18EC744	Branch:	ECE			
Answer any FIVE FULL Questions						MARKS	CO	RBT
1a	List the entities behind the IoT technology and give examples in each case Here's a concise version of the answer: 1. Sensors/Devices - Role: Collect data from the environment. - Examples: Temperature sensors (e.g., Nest Thermostat), wearable devices (e.g., Apple Watch). 2. Connectivity - Role: Transmit data from sensors to central systems. - Examples: Wi-Fi, Bluetooth. 3. Data Processing - Role: Analyze data to generate insights. - Examples: Cloud computing platforms like AWS IoT. 4. User Interface - Role: Allows users to interact with the IoT system. - Examples: Mobile apps, dashboards. 5. Actuators - Role: Execute actions based on processed data. - Examples: Smart locks, automated lights.					10	CO2	L2
b	Represent the following: i)Direct and indirect accesses between CoAP client and CoAP server ii) CoAP Client access for resource directory iii) CoAP client and server access using porxies. When discussing direct and indirect accesses between a CoAP (Constrained Application Protocol) client and a CoAP server, it's important to understand how these two entities communicate in an IoT environment. 1. Direct Access Definition: In direct access, the CoAP client communicates directly with the CoAP server without any intermediaries. The client sends requests (e.g., GET, POST) directly to the server's IP address, and the server responds directly to the client. Example: A smart sensor (CoAP client) sending data directly to a central hub (CoAP server) within a smart home network.					10	CO2	L2

	2. Indirect Access Definition: In indirect access, intermediaries such as proxies, gateways, or brokers are involved in the communication between the CoAP client and the server. These intermediaries can perform functions like load balancing, security enforcement, or protocol translation. Example: A CoAP client in a low-power sensor network sends data to a CoAP server through an MQTT broker, which acts as an intermediary to handle communication across different protocols. 1. CoAP Client Access for Resource Directory Definition: In a CoAP-based IoT system, a Resource Directory (RD) is used to manage resources offered by CoAP servers. The RD allows CoAP clients to discover resources available on various servers. Process: Registration: CoAP servers register their resources with the RD, providing information like resource location, type, and attributes. Discovery: CoAP clients query the RD to discover resources based on certain criteria (e.g., type of sensor, location). Access: Once discovered, the client can directly access the resource on the server. Example: A CoAP client queries the RD to find temperature sensors in a building and retrieves the list of sensors from different servers. 2. CoAP Client and Server Access Using Proxies Definition: Proxies in CoAP networks are intermediaries that can forward requests from a client to a server. Proxies can be used for various purposes, such as load balancing, security, and protocol translation. Types of Proxies: Forward Proxy: Receives requests from the client, forwards them to the server, and then returns the response to the client. This is useful when the client cannot directly communicate with the server. Reverse Proxy: Positioned between the client and multiple servers, it distributes client requests across the servers. This helps in load balancing and hides the server's details from the client.			
2a	Represent the following i) CoAP request or response communication to a machine, IoT device or MT. ii) A computer or machine interface using IP communication to a mobile service A provider. iii) A machine or IOT dev1, ice or MO communication of CoAP request or response communication. i) CoAP Request/Response Communication - A CoAP client, such as a machine or IoT device, communicates with a CoAP server using CoAP protocols. The client sends requests (e.g., GET, POST), and the server	10	Co2	L1

	responds with the requested data or action results. Example: A smart thermostat retrieving temperature data from a sensor. ii) IP Communication with a Mobile Service Provider - A computer or machine uses IP-based protocols (TCP/IP, UDP/IP) to interface with a mobile service provider's network, enabling data transmission or accessing mobile services like cloud storage or SMS. Example: A remote monitoring system transmitting data via cellular network to the cloud. iii) Machine-Originated (MO) CoAP Communication - An IoT device initiates CoAP requests (MO communication) to a server, sending data or requesting actions. The server processes and responds, enabling the device to execute further operations. Example: A water level sensor sending updates to a remote server.			
b	Write the equations that represent the simple conceptual framework of IOT and complex conceptual framework for, IOT using cloud platform based processes and services. Simple IoT Conceptual Framework: The simple IoT framework can be represented by: \[\text{Data Collection (Sensors/Devices)} \rightarrow \text{Data Transmission (Connectivity)} \rightarrow \text{Data Processing (Edge/Local)} \rightarrow \text{Action/Response} \] Here, sensors/devices collect data, which is transmitted, processed locally or at the edge, and leads to actions or responses. Complex IoT Conceptual Framework Using Cloud: The complex IoT framework involving a cloud platform can be represented by: \[\text{Data Collection (Sensors/Devices)} \rightarrow \text{Data Transmission (Connectivity)} \rightarrow \text{Cloud Processing (Data Storage + Analytics)} \rightarrow \text{Service/Action (e.g., Machine Learning, Actuation)} \rightarrow \text{User Interface (Dashboards/Apps)} \] In this framework, data is collected, transmitted to the cloud for processing and storage, analyzed using cloud-based services (e.g., ML), and the results are communicated back for user interaction or automated actions.	10	Co2	L1
3a	Illustrate the different classes of internet network and for what reason subnet masks are used? Classes of Internet Networks The Internet Protocol (IP) address system is traditionally divided into five classes, each serving different types of networks: 1. Class A:	10	Co2	L2

	- Range: 1.0.0.0 to 126.0.0.0 - Purpose: Designed for large networks with many hosts. - Subnet Mask: 255.0.0.0 - Example: Large corporations or ISPs. 2. Class B: - Range: 128.0.0.0 to 191.255.0.0 - Purpose: Medium-sized networks. - Subnet Mask: 255.255.0.0 - Example: Universities, large businesses. 3. Class C: - Range: 192.0.0.0 to 223.255.255.0 - Purpose: Small networks. - Subnet Mask: 255.255.255.0 - Example: Small businesses, home networks. 4. Class D (Multicast): - Range: 224.0.0.0 to 239.255.255.255 - Purpose: Used for multicast groups (not for regular IP networks). 5. Class E (Experimental): - Range: 240.0.0.0 to 255.255.255.255 - Purpose: Reserved for experimental purposes (not for general use). Purpose of Subnet Masks Subnet masks are used to divide an IP network into smaller, more manageable sub-networks (subnets). They help in: 1. Identifying Network and Host Portions: Subnet masks differentiate the network part of an IP address from the host part, enabling devices to determine if another IP is within the same network or requires routing. 2. Improving Network Management: By dividing larger networks into subnets, organizations can efficiently manage IP address allocation, enhance security by isolating network segments, and optimize performance by reducing broadcast traffic. 3. Supporting Hierarchical Routing: Subnets allow for hierarchical routing, which reduces the size of routing tables and improves overall network efficiency.			
b	Describe the different classification of cloud service models and give the example in each case. Cloud services are categorized into different models based on the level of control, management, and abstraction they provide. The primary cloud service models are: 1. Infrastructure as a Service (IaaS)	10	Co2	L2

	- Definition: Provides virtualized computing resources over the internet. Users can rent IT infrastructure such as servers, storage, and networking on a pay-as-you-go basis. - Control: Users have control over the operating systems, applications, and data but not the underlying hardware. - Example: Amazon Web Services (AWS) EC2 – Allows users to create and manage virtual servers, storage, and networking resources. 2. Platform as a Service (PaaS) - Definition: Provides a platform allowing customers to develop, run, and manage applications without dealing with the underlying infrastructure. It includes development tools, databases, and middleware. - Control: Users focus on application development and deployment, while the provider manages the underlying infrastructure and platform. - Example: Google App Engine – Offers a platform to build and deploy applications without managing the underlying infrastructure. 3. Software as a Service (SaaS) - Definition: Delivers software applications over the internet on a subscription basis. Users access the software via a web browser, and the provider manages the infrastructure, platform, and application. - Control: Users have minimal control over the infrastructure and platform, focusing primarily on using the application. - Example: Microsoft Office 365 – Provides access to office productivity tools like Word, Excel, and Outlook via a web browser. 4. Function as a Service (FaaS) - Definition: A serverless computing model where users can deploy individual functions or pieces of code that are executed in response to events. Users pay only for the actual compute time used. - Control: Users focus solely on coding functions and managing their triggers, while the provider handles the infrastructure and scaling. - Example: AWS Lambda – Executes code in response to triggers such as changes in data or system events, without requiring server management.			
4a	Show the details IP packets received transmitted at or to network layer. IP packets at the Network Layer (Layer 3) are crucial for routing data across networks. Here's a concise overview: IP Packet Structure - Header: Includes fields such as Version (IPv4/IPv6), Header Length (IPv4) or Payload Length (IPv6), Type of Service (ToS, IPv4 only), Total Length, Identification, Flags, Fragment Offset, Time to Live (TTL), Protocol, Header Checksum, Source IP Address, and Destination IP Address. - Payload: Contains the actual data being transmitted, which could be from protocols like TCP or UDP. Transmission Process	10	Co2	L1

	- Sending: The IP packet is created, routed based on the destination IP, and encapsulated in a frame for transmission. - Receiving: The packet is received, decapsulated, and processed by checking the header and routing it to the correct destination. The payload is then delivered to the appropriate transport layer for further handling. This ensures efficient data transfer across different network segments.			
b	Name two open source platform used in the IOT based data collection, computing. Give the reasons for using these platform. Two popular open-source platforms used in IoT-based data collection and computing are Home Assistant and Apache Kafka. 1. Home Assistant - Purpose: Used for home automation, integrating various IoT devices and sensors. - Reasons for Use: - Flexibility: Supports a wide range of devices and integrations. - Customizability: Users can create customized automation rules and dashboards. - Community Support: Extensive documentation and community contributions enhance functionality and ease of use. 2. Apache Kafka - Purpose: A distributed streaming platform used for real-time data ingestion and processing. - Reasons for Use: - Scalability: Handles large volumes of data with high throughput and low latency. - Reliability: Provides durable message storage and fault tolerance. - Integration: Easily integrates with various data processing frameworks and storage solutions. Both platforms offer robust, scalable solutions for managing IoT data, making them valuable for diverse applications.	10	Co2	L1

5a	Write the program for Arduino controlled traffic light control system in which green light is ON towards East-West the starting. Three traffic lights-Red, Yellow and Green needs to be controlled on each of the north, east, south and west in clockwise pathways. Assume delays 5s each between, successive states of LEDS and steady state for 70s for a pair of pathways.
	
	cpp
	// Define pin numbers for LEDs
	const int redEast = 2;
	const int yellowEast = 3;
	const int greenEast = 4;
	const int redWest = 5;
	const int yellowWest = 6;
	const int greenWest = 7;
	const int redNorth = 8;
	const int yellowNorth = 9;
	const int greenNorth = 10;
		10	Co2	L1

```
const int redSouth = 11;
const int yellowSouth = 12;
const int greenSouth = 13;

void setup() {
  // Set all LED pins as OUTPUT
  pinMode(redEast, OUTPUT);
  pinMode(yellowEast, OUTPUT);
  pinMode(greenEast, OUTPUT);
  pinMode(redWest, OUTPUT);
  pinMode(yellowWest, OUTPUT);
  pinMode(greenWest, OUTPUT);
  pinMode(redNorth, OUTPUT);
  pinMode(yellowNorth, OUTPUT);
  pinMode(greenNorth, OUTPUT);
  pinMode(redSouth, OUTPUT);
  pinMode(yellowSouth, OUTPUT);
  pinMode(greenSouth, OUTPUT);
}

void loop() {
  // East-West Green
  digitalWrite(greenEast, HIGH);
  digitalWrite(redEast, LOW);
  digitalWrite(yellowEast, LOW);
  digitalWrite(greenWest, HIGH);
  digitalWrite(redWest, LOW);
  digitalWrite(yellowWest, LOW);
  delay(70000); // Steady state for East-West

  // East-West Yellow
  digitalWrite(greenEast, LOW);
  digitalWrite(yellowEast, HIGH);
  digitalWrite(greenWest, LOW);
  digitalWrite(yellowWest, HIGH);
  delay(5000); // Yellow delay

  // North-South Green
  digitalWrite(yellowEast, LOW);
  digitalWrite(yellowWest, LOW);
  digitalWrite(greenNorth, HIGH);
  digitalWrite(redNorth, LOW);
  digitalWrite(yellowNorth, LOW);
  digitalWrite(greenSouth, HIGH);
  digitalWrite(redSouth, LOW);
  digitalWrite(yellowSouth, LOW);
  delay(70000); // Steady state for North-South

  // North-South Yellow
```

	digitalWrite(greenNorth, LOW);
	digitalWrite(greenSouth, LOW);
	digitalWrite(yellowNorth, HIGH);
	digitalWrite(yellowSouth, HIGH);
	delay(5000); // Yellow delay
	
	// Cycle repeats
	}
	
	Explanation:
	- Setup: Initializes the LED pins as outputs.
	- Loop: Controls the traffic light states with specified delays:
	- Green: For East-West or North-South direction.
	- Yellow: After green for both East-West and North-South directions.
	- Delays: 70 seconds for steady green light and 5 seconds for yellow.
	
	This code ensures a cyclic traffic light pattern with the specified timing.			

b	Illustrate the layered attacker model and possible attacks using IETF six layer modified model for IOT/M2N1. The IETF six-layer modified model for IoT/M2M (Machine-to-Machine) networks enhances the traditional network architecture by adding layers specific to IoT scenarios. Here's an illustration of the layered attacker model and possible attacks: IETF Six-Layer Model: 1. Physical Layer: Involves hardware components (e.g., sensors, actuators). - Possible Attacks: Physical tampering, hardware attacks. 2. Data Link Layer: Manages node-to-node communication. - Possible Attacks: MAC address spoofing, denial-of-service (DoS) attacks. 3. Network Layer: Handles routing and forwarding of packets. - Possible Attacks: IP spoofing, routing attacks. 4. Transport Layer: Manages end-to-end communication and data integrity. - Possible Attacks: Session hijacking, data interception. 5. Application Layer: Provides interfaces and services for applications. - Possible Attacks: Injection attacks, application vulnerabilities. 6. Management Layer: Oversees network management and configuration. - Possible Attacks: Unauthorized access, configuration manipulation. These layers highlight where attacks can occur and underscore the need for robust security measures across all layers to protect IoT/M2M systems.	10	Co2	L1
6a	Write the programming of Arduino for usages of RFID Serial data reading using UART port. Use port 2 and 5 for 11erial RX and serial TX.	10	Co2	L1

	#include <SoftwareSerial.h>
	
	// Define RX and TX pins
	const int rxPin = 2; // Serial RX
	const int txPin = 5; // Serial TX
	
	// Create a SoftwareSerial instance
	SoftwareSerial rfidSerial(rxPin, txPin);
	
	void setup() {
	// Start hardware serial communication for debugging
	Serial.begin(9600);
	
	// Start software serial communication with RFID module
	rfidSerial.begin(9600); // Adjust baud rate if necessary
	
	Serial.println("RFID Serial Data Reading Initialized");
	}
	
	void loop() {
	// Check if data is available from the RFID module
	if (rfidSerial.available()) {
	// Read and print the data
	String rfidData = "";
	while (rfidSerial.available()) {
	char c = rfidSerial.read();
	rfidData += c;
	}
	Serial.print("RFID Data: ");
	Serial.println(rfidData);
	}
	}			

b	Mention the five levels of software development for' application and services for IOT/M2M (no need to draw the block diagram and explanation). The five levels of software development for applications and services in IoT/M2M systems are: 1. Device Layer Software: Includes firmware and low-level drivers that run directly on IoT devices, managing hardware interactions and basic functionalities. 2. Communication Layer Software: Handles data transmission and communication protocols between devices and the network, ensuring data integrity and efficient transmission. 3. Data Management Software: Manages the collection, storage, and retrieval of data from IoT devices. This includes databases and data handling systems that ensure data consistency and accessibility.	10	Co2	L1

	4. Application Layer Software: Provides specific functionalities and services for end-users, including applications that interact with the data and devices to deliver value, such as dashboards and control interfaces. 5. Service Layer Software: Involves backend services that provide additional features like analytics, machine learning, and integration with other systems, enhancing the overall functionality and usability of IoT applications.			
7a	Explain any five required mechanisms in case of wireless sensor network. In a wireless sensor network (WSN), several mechanisms are essential to ensure efficient operation and data integrity. Here are five crucial mechanisms: 1. Energy Management: - Purpose: To optimize the energy consumption of sensor nodes, as these nodes are typically battery-powered. - Mechanism: Techniques such as sleep/wake cycles, energy-efficient routing algorithms, and low-power communication protocols are employed to extend the network's lifetime. 2. Data Aggregation: - Purpose: To minimize data redundancy and reduce communication overhead. - Mechanism: Aggregation techniques combine data from multiple sensors before transmission, which helps in reducing the amount of data sent over the network and conserving energy. 3. Routing Protocols: - Purpose: To efficiently transmit data from sensor nodes to a sink or base station. - Mechanism: Protocols like LEACH (Low-Energy Adaptive Clustering Hierarchy) and RPL (Routing Protocol for Low-Power and Lossy Networks) are designed to optimize route selection and manage data flow. 4. Localization: - Purpose: To determine the physical location of sensor nodes within the network. - Mechanism: Techniques such as GPS-based localization or range-based algorithms (e.g., trilateration) provide the position information required for context-aware applications and efficient data routing. 5. Fault Tolerance: - Purpose: To ensure network reliability and data integrity despite node failures or environmental disturbances. - Mechanism: Methods like redundancy, self-healing algorithms, and fault detection mechanisms help maintain network functionality and recover from node or link failures.	10	Co2	L1
b	Illustrate the energy consumption in microcontrollers used in WSN. Energy consumption in microcontrollers used in Wireless Sensor Networks (WSNs) is a critical factor, as these microcontrollers are often powered by batteries and need	10	Co2	L1

	to operate efficiently to extend the network's lifetime. Here's an illustration of the main components contributing to energy consumption: Energy Consumption Components: 1. CPU Processing: - Description: The energy used by the microcontroller during computation and execution of tasks. - Impact: Higher processing demands (e.g., complex algorithms, data processing) increase energy consumption. 2. Radio Transmission: - Description: Energy consumed during data transmission over the wireless communication module. - Impact: Transmitting data typically consumes more energy than receiving. Power levels and distance affect the consumption rate. 3. Radio Reception: - Description: Energy used by the microcontroller when receiving data from other nodes. - Impact: Although less than transmission, reception still consumes a significant amount of energy, especially with frequent or high-volume data. 4. Sleep Mode: - Description: Energy consumption when the microcontroller is in a low-power sleep state. - Impact: Various sleep modes (e.g., idle, deep sleep) have different energy consumption levels, with deeper sleep modes conserving more energy. 5. Sensor Operation: - Description: Energy required to operate attached sensors for data collection. - Impact: The power consumption depends on the type of sensor and its operational state (e.g., active sensing vs. idle). Energy Consumption Management Strategies: - Duty Cycling: Alternating between active and sleep modes to reduce overall power usage. - Efficient Protocols: Using energy-efficient communication and routing protocols. - Low-Power Components: Employing low-power microcontrollers and sensors. Managing these components effectively helps in optimizing the energy consumption of microcontrollers in WSNs, thereby enhancing the network's longevity and efficiency.			
8a	State the difference between Microcontrollers versus microprocessor, FPGAS and ASICS. Microcontrollers (MCUs):	10	Co2	L2

	- Definition: Integrated circuits that include a CPU, memory (RAM and ROM), and peripherals on a single chip. - Use Case: Ideal for embedded systems and IoT applications requiring control, sensing, and communication. - Performance: Generally lower processing power compared to microprocessors but sufficient for control tasks. - Power Consumption: Low, suitable for battery-operated devices. - Flexibility: Limited; functionality is fixed based on design. Microprocessors (MPUs): - Definition: CPUs that focus on processing power and typically require external memory and peripherals. - Use Case: Used in general-purpose computing, including desktops, laptops, and high-performance applications. - Performance: High processing power and capability to handle complex tasks. - Power Consumption: Higher compared to microcontrollers, not typically used in low-power applications. - Flexibility: High; can run complex operating systems and applications. FPGAs (Field-Programmable Gate Arrays): - Definition: Integrated circuits that can be programmed after manufacturing to implement custom hardware functions. - Use Case: Suitable for applications needing customizable hardware for specific tasks, such as signal processing or complex logic. - Performance: Can be optimized for parallel processing and custom hardware implementations. - Power Consumption: Variable; depends on the design and application. - Flexibility: High; can be reprogrammed to change functionality even after deployment. ASICs (Application-Specific Integrated Circuits): - Definition: Custom-designed chips created for a specific application or function, with fixed hardware. - Use Case: Ideal for high-volume production where cost and performance can be optimized for specific tasks. - Performance: High; optimized for specific functions, offering superior performance and efficiency. - Power Consumption: Typically lower for the specific application, compared to general-purpose solutions. - Flexibility: Low; functionality is fixed once designed and manufactured.			
b	What are the 3 types of mobility WSN scenario and illustrate. In Wireless Sensor Networks (WSNs), mobility can significantly impact network design and performance. The three types of mobility scenarios in WSNs are: 1. Node Mobility	10	Co2	L2

	- Description: Sensor nodes themselves are mobile, meaning they can move within the network area. - Example: In a military surveillance scenario, sensor nodes are placed on moving vehicles or drones. These nodes need to communicate and adapt their routing as they move, requiring dynamic routing protocols that can handle changing network topologies. 2. User Mobility - Description: The users or operators of the sensor network are mobile, affecting how they interact with the network. - Example: In a smart city setup, users might use mobile devices (smartphones, tablets) to interact with various sensors (e.g., traffic cameras, environmental sensors). As users move, the network must handle interactions from different locations and possibly adjust data collection or processing based on user location. 3. Sink Mobility - Description: The sink or base station, which collects data from sensor nodes, is mobile. - Example: In agricultural monitoring, a mobile sink (e.g., a vehicle equipped with communication equipment) moves through a field to gather data from stationary sensor nodes. This mobility requires the network to dynamically adapt to data collection strategies as the sink moves to different locations. Illustration: - Node Mobility: A fleet of drones equipped with sensors flying over a disaster area to monitor environmental conditions. As the drones move, they must continuously update their data routing paths. - User Mobility: An app on a smartphone allows users to check real-time air quality from various sensors scattered around a city. As users move, the app must pull relevant data from the nearest sensors. - Sink Mobility: A mobile laboratory vehicle equipped with sensors collects data from various fixed sensors placed in a forest for wildlife monitoring. The vehicle's movement requires the network to adapt to changing data collection points. Each type of mobility scenario requires specific strategies to manage network connectivity, data routing, and energy consumption effectively.			

9a	Compare different modulation schemes available.1 in the design of physical layer and transceiver in WSNs. In the design of the physical layer and transceiver for Wireless Sensor Networks (WSNs), different modulation schemes play a crucial role in determining the efficiency and performance of communication. 1. Amplitude Shift Keying (ASK) - Description: Modulates the amplitude of the carrier signal to represent data. - Advantages: - Simple implementation. - Low power consumption. - Disadvantages: - Susceptible to noise and signal degradation. - Lower data rates compared to other schemes. 2. Frequency Shift Keying (FSK) - Description: Modulates the frequency of the carrier signal to represent data. - Advantages: - Better noise immunity than ASK. - Suitable for low to moderate data rates. - Disadvantages: - More complex than ASK. - Higher power consumption compared to ASK. 3. Phase Shift Keying (PSK) - Description: Modulates the phase of the carrier signal to represent data. - Types: Binary Phase Shift Keying (BPSK), Quadrature Phase Shift Keying (QPSK). - Advantages: - High noise immunity. - Efficient in terms of power. - Disadvantages: - Requires more complex demodulation. - Bandwidth requirements can be higher than ASK and FSK. 4. Quadrature Amplitude Modulation (QAM) - Description: Combines amplitude and phase modulation to increase data rates. - Advantages: - Supports high data rates. - Efficient use of bandwidth. - Disadvantages: - More complex transceiver design. - Higher power consumption and susceptibility to noise compared to PSK. 5. Orthogonal Frequency Division Multiplexing (OFDM)	10	Co2	L2

	- Description: Divides the channel into multiple sub-carriers, each modulated using schemes like QAM or PSK. - Advantages: - Excellent for handling multipath fading and interference. - High data rates and efficient spectrum usage. - Disadvantages: - Complex implementation. - High power consumption due to multiple sub-carriers. Comparison Summary: - ASK is simple but less robust against noise. - FSK offers better noise immunity but is less efficient than PSK. - PSK provides good performance with moderate complexity and power consumption. - QAM supports high data rates but requires more power and complex design. - OFDM is suitable for high-speed data and challenging environments but has high complexity and power demands. Selecting the appropriate modulation scheme depends on factors such as the required data rate, power consumption, noise tolerance, and system complexity. Each scheme has trade-offs that must be considered based on the specific needs of the WSN application.			
b	Explain Perkins model of assignment of MAC address in WSN. The Perkins model for MAC address assignment in Wireless Sensor Networks (WSNs) addresses dynamic address management challenges. It involves: 1. Dynamic Address Allocation: Nodes obtain addresses from a centralized or distributed system rather than using static addresses, adapting to network changes. 2. Address Management: A central authority or distributed algorithm assigns unique addresses and manages them, preventing conflicts and ensuring proper allocation. 3. Address Reclamation: Addresses of inactive or failed nodes are reclaimed and recycled for new nodes, optimizing address usage and preventing exhaustion. 4. Scalability and Flexibility: The model supports network growth and changes in topology by dynamically managing address assignments. 5. Conflict Resolution: Mechanisms detect and resolve address conflicts, ensuring unique address assignment. Overall, the Perkins model enhances network scalability, flexibility, and efficient address management in dynamic WSN environments.	10	Co2	L2
10a	Explain the general concept of geographic routing. Geographic routing is a technique used in wireless networks where routing decisions are based on the geographic locations of nodes rather than traditional routing tables. Nodes use their own location and possibly the locations of nearby nodes, obtained via GPS or other positioning systems, to forward packets. The	10	Co2	L2

	primary method, **greedy forwarding**, involves sending packets to the neighboring node that is closest to the destination, thereby progressively moving the packet towards its target. This approach simplifies routing by using location-based addressing, which reduces the need for complex routing tables and minimizes routing overhead. However, geographic routing depends on accurate location information and may face challenges like local optima, where packets get stuck in areas with no further progress. Overall, geographic routing offers scalability and efficiency, making it suitable for large and dynamic networks.			
b	Illustrate the S-MAC principle used in the low duty cycle protocols and wakeup concept. S-MAC (Sensor-MAC) is a low-duty-cycle protocol designed for Wireless Sensor Networks (WSNs) to save energy and extend node lifespan. S-MAC Principle: 1. Low Duty Cycle Operation: - Description: Nodes alternate between active and sleep states. During the active period, nodes listen for and transmit data. In sleep mode, they conserve energy by powering down their radio and other components. - Mechanism: Nodes use synchronized sleep schedules to minimize energy consumption while ensuring data transmission reliability. 2. Sleep Scheduling: - Description: Nodes coordinate their sleep schedules to avoid collisions and ensure that neighboring nodes are awake when communication is required. - Mechanism: Nodes use periodic synchronization messages to maintain consistent wakeup times across the network. 3. Collision Avoidance: - Description: To prevent data collisions during active periods, S-MAC employs a request-to-send (RTS) and clear-to-send (CTS) handshake, ensuring that nodes only transmit when the channel is clear. Wakeup Concept: - Description: Nodes wake up at predetermined intervals to check for incoming messages and then return to sleep if no communication is needed. This periodic wakeup reduces energy consumption by minimizing the time the radio is active. S-MAC efficiently balances energy conservation with network communication needs, making it well-suited for energy-constrained WSN environments.	10	Co2	L2