fri iot tandon

Here's a well-structured and cleanly formatted version of your **Week 3 Notes**:

Week 3 Notes – Connectivity Technologies & Sensor Networks

Connectivity Technologies – Part III

HART & WirelessHART

- **WirelessHART**: An advanced version of the HART Protocol for smart field device networking.

HART (Highway Addressable Remote Transducer) is a communication protocol that allows two-way digital communication between smart field devices and control systems,

- **Physical Layer**:
  - Based on **IEEE 802.15.4**.
  - Uses **time-synchronized channel hopping** with super-frames (10ms timeslots).
  - Implements **channel blacklisting** to avoid interference.
- **Network & Transport Layers**:
  - Enable **mesh networking**.
  - Devices forward packets and maintain a **network graph**.
- **Application Layer**:
  - Command-response messaging, consistent with wired HART.
- **Congestion Control**:
  - Operates in **2.4 GHz ISM band**, excluding channel 26.
- **Network Manager**:
  - Manages routing, timing, node access, and security.
- **WirelessHART vs. ZigBee**:
  - Channel hopping: *per message* (WirelessHART) vs. *per network* (ZigBee).

- MAC Layer: *TDMA* (WirelessHART) vs. *CSMA/CD* (ZigBee).

NFC (Near Field Communication)

- **Derived from**: RFID.
- **Types**: Type A, Type B, **FeliCa** (common in Japan).
- **Device Types**:
  - *Passive*: Can only transmit stored data (e.g., NFC tags).
  - *Active*: Can transmit and receive (e.g., smartphones).
- **Working Principle**: **Magnetic induction**.
- **Specifications**:
  - Frequency: **13.56 MHz**.
  - Data Rates: **106, 212, 424 Kbps**.
  - Range: **< 20 cm**.
  - Storage: **96–512 bytes**.
- **Modes of Operation**:
  - Peer-to-Peer
  - Read/Write
  - Card Emulation
- **Applications**:
  - Mobile payments, parcel tracking, ads, smart toys, home automation.

Connectivity Technologies – Part IV

Bluetooth

- **Purpose**: Short-range wireless cable replacement; secure ad-hoc connections.
- **Device Classes**:
  - Class 3: 1 m

    - Class 2: 10 m (common)
    - Class 1: 100 m
- **Connection Setup**:
    - **Inquiry**, **Paging**, **Connection**
- **Power Saving Modes**:
    - *Sniff*: Intermittent listening
    - *Hold*: Sleep for defined time
    - *Park*: Inactive until reactivated
- **Protocol Stack**:
    - **Baseband**: Manages channel access, error correction, security
    - **L2CAP**: Multiplexing, segmentation
    - **RFCOMM**: Serial port emulation
- **Network**: **Piconet**
    - One *master*, up to *7 slaves*
    - Communication: *Master* ↔ *Slave*, no direct slave-to-slave

---

Connectivity Technologies – Part V

Z-Wave

- **Purpose**: RF-based home automation.
- **Frequencies**: Region-specific (e.g., 908.42 MHz – US, 868.42 MHz – EU).
- **Topology**: **Mesh**, supports up to **232 nodes**.
- **Modulation**: **GFSK**
- **Channel Encoding**: **Manchester**
- **Network Management**:
    - One primary controller.
    - Unique **Home ID** for network, **Node ID** for each device.
- **Self-Healing Mesh**: Routes around obstacles.

- **Z-Wave vs. ZigBee**:
  - Z-Wave: User-friendly, secure, slightly expensive.
  - ZigBee: Ultra low-power, customizable, tech-savvy users.

ISA 100.11A

- **Use Case**: Industrial automation.
- **Topologies**: Star, tree, mesh.
- **Supports**: Ethernet, fieldbuses, radio links.
- **Key Features**:
  - Multi-protocol support, error detection, TDMA, QoS.
  - **Security**: Dual-layer encryption (data link & transport), key distribution via security manager.
- **Usage Classes**: Based on criticality (Safety → Monitoring).

Sensor Networks – Part I

Wireless Sensor Networks (WSNs)

- **Composition**: Many sensor nodes deployed to monitor environmental factors.
- **Data Relay**: Uses **multi-hop paths** to transmit data to a sink.
- **Node Characteristics**:
  - Compact, battery-powered, multifunctional.
  - Short-range, often run **TinyOS**.
- **Constraints**:
  - Small size, low power, autonomy, low cost.
- **Applications**:
  - Monitoring: Temp, humidity, light, pressure, etc.
- **Detection Scenarios**:

    - Single/Multi Source vs. Single/Multi Object
- **Challenges**:
    - Energy management, Security threats (interference, eavesdropping).
- **Sensor Web**:
    - Interfaces to access/control sensors online.
    - Includes: SOS, SPS, SAS, WNS, SensorML, O&M.

Sensor Networks – Part II

Node Behavior in WSNs

- **Types**:
    - *Normal*
    - *Misbehaving*:
        - Unintentional: Failed/Badly failed
        - Intentional: Selfish/Malicious
- **Restoration Techniques**:
    - **CoRD**, **CoRAD**

Topology & Management

- **Event-Aware Topology**:
    - Dynamic response to events (location, area, duration).
- **INTSEM (Information-Theoretic Self Management)**:
    - Controls transmission via sleep cycles for energy efficiency.

Social Sensing & Rare Events

- **Challenge**: Monitoring rare events with standard methods (e.g., SMAC, PW-MAC).

- **Solution: PDC (Probabilistic Duty Cycle)** using social media insights to adjust node activity.

Applications

Mines:

- Fire Monitoring and Alarms (FMA) using real-time sensor feedback.

Healthcare:

- **WBANs (Wireless Body Area Networks)**:
  - Continuous monitoring of vitals, elderly care.
  - **Cloud-Assisted WBANs**:
    - Post-disaster fairness in data aggregation using social choice.
    - Uses pseudo-clusters and mobile aggregation centers.
  - **Payload Tuning**:
    - Adjusts for patient-specific traits using **FIS** and **MDP**.
  - **PATS (Priority-based Time-slot Allocation Scheme)**:
    - Evolutionary game theory approach to prioritize medical emergencies.

Let me know if you'd like this in a downloadable PDF, formatted as flashcards, or turned into a revision summary!

Sure! Here's a well-organized, formatted version of your **Week 4, Week 5, and Week 6 Notes** for easy reading and review:

Week 4 Notes

Sensor Networks – Part III

- **Target Tracking**:

- Push-based, poll-based, and guided formulations.
- Cluster, tree, and face structures respectively.

- **WSNs in Agriculture**:
  - Used for agricultural intrusion detection.
  - Distributed topology management (coverage, connectivity, lifetime).
  - Coalition Formation Games for WMSNs.

AID (Agriculture Information Device), a set of sensor nodes are deployed over an agricultural field to:

- **3D Localization in UWSNs**:
  - Silent, energy-efficient localization using 3 surface anchor nodes.
  - Iterative, mobility-aware approach.
- **Self-Organizing Virtual Architecture (Tic-tac-toe-arch)**:
  - Calculates node connectivity durations.
  - Dynamic virtual topology formation.

Sensor Networks – Part IV

The objective of coverage in WSN is to use minimum number of sensors and maximize the network lifetime.

- **WSN Coverage**:
  - Monitors area of interest, ensures connectivity.
  - Event-driven (e.g., forest fire) or on-demand (e.g., inventory) reporting.
  - Goal: Fewer sensors, longer network lifetime.
- **Coverage Algorithms**:
  - Centralized (global map), distributed (neighbor-based), localized (subset of nodes).
  - Sensor deployment: Deterministic or random.
- **Coverage Types in Static WSNs**:
  - **Area Coverage**: Random and connected random.
  - **Point Coverage**
  - **Barrier Coverage**: Weak and strong types.
- **Coverage Maintenance**:
  - Ensure all 'crossings' (intersections of sensing areas) are covered.

- **Optimal Geographical Density Control (OGDC) Algorithm:**
  - Nodes compute deviations (distance, angle) from desired config.
  - Optimal node remains active; others sleep to conserve energy.

Sensor Networks – Part V

If there is a failure in the stationary sensor network then it is likely that the point of failure can partition the network into two or more fragments.

- **Stationary WSNs**:
  - Static, easy deployment and optimized placement.
  - Risk of partitioning on node failure.
- **Mobile WSNs (MWSNs)**:
  - Like MANETs with self-CHOP properties (Configure, Heal, Optimize, Protect).
  - Components: Mobile nodes, sinks, data mules.
  - Applications: Marine life monitoring, water quality.
- **Participatory Sensing**:
  - Human-carried devices for distributed sensing.
  - Enables data sharing and authenticity verification.

Human Centric Sensing

Energy of devices (battery life) and participant selection (choosing reliable human users/devices) are major challenges in Human-Centric Sensing systems.

- **Flying Ad Hoc Networks (FANETs)**:
  - Gateway selection by stable node in sub-area.
  - Not ideal for time-critical relaying.
- **Machine-to-Machine (M2M) Communication:**
  - Numerous low-cost, low-energy nodes.
  - Automatic communication, minimal human interaction.
  - **Low-end Nodes**: Cheap, static, limited capabilities, no IP, dense deployment.

Week 5 Notes

Device Interoperability

User Interoperability: This refers to the interoperability problem between a user and a device.

Device Interoperability: This refers to the interoperability problem between two different devices

- Describes actions during new device connection and control/monitoring phases.

the ability of computer systems or software to exchange and make use of information.

Introduction to Arduino Programming – Part I

- **Arduino Overview**:
  - Open-source hardware (board) + software (IDE).
  - Accepts analog/digital input, gives output without extra loaders.
- **Arduino Boards**:
  - Based on ATMEGA328, ATMEGA32u4, ATMEGA2560, AT91SAM3X8E.
- **Arduino IDE**:
  - C/C++-based.
  - Key operations: New sketch, open, examples, verify, upload, save, serial monitor.
- **Sketch Structure**:
  - `setup()` – initialization.
  - `loop()` – continuous execution.
- **Data Types**:
  - Includes `int`, `char`, `boolean`, `byte`, `float`, `double`, `String`, etc.
- **Function Libraries**:
  - `pinMode()`, `digitalWrite()`, `analogRead()`
  - `delay(ms)`, character functions like `isdigit()`, `isalpha()`
- **Example – Blink**:
  - LED blinks using `digitalWrite()` and `delay()`.

Introduction to Arduino Programming – Part II

- **Operators**: Arithmetic, comparison, boolean, bitwise, compound.
- **Control Statements**: `if`, `else`, `switch`, ternary operator.
- **Loops**: `for`, `while`, `do...while`, nested/infinite loops.
- **Arrays**: Declared with/without size/values, multi-dimensional supported.
- **String Handling**:

    - Char array or String object.
    - Methods: `replace()`, `toUpperCase()`, `length()`
- **Math Library**: `min()`, `max()`, etc.
- **Random Numbers**:
    - `randomSeed()`, `random(min, max)`
- **Interrupts**:
    - External events.
    - Functions: `digitalPinToInterrupt()`, `attachInterrupt()`
- **Example – Traffic Control**:
    - LEDs simulate traffic lights using `digitalWrite()` and `delay()`.

Use of different programming languages such as JavaScript, Python, JAVA, and others is an example of heterogeneity in IoT. This brings in the need for interoperability

---

Integration of Sensors & Actuators with Arduino

- **Part I – DHT Sensor**:
    - Reads temperature/humidity.
    - Uses `DHT.h`, `DHT.read11()`
- **Part II – Actuators**:
    - Convert energy to motion.
    - Types: Servo, stepper, solenoid, relay, AC motors.
- **Servo Motor Interfacing**:
    - Control pin to digital pin.
    - Use `Servo` library: `attach()`, `write()`, `read()`, etc.

Week 6 Notes

Introduction to Python Programming – Part I

- **Why Python?**
    - Simple, readable, versatile, supports hardware interfacing, open-source.

- **IDEs**: PyCharm, Spyder.
- **Basics**:
  - `print()` function.
  - Indentation defines blocks.
- **Data Types**: Numbers, Strings, Lists, Tuples, Dictionaries.
- **Control Statements**: `if`, `elif`, `else`, `while`, `for`, `break`, `continue`.
- **Functions**:
  - Use `def`, can return values or multiple values.
- **Variable Scope**:
  - Global vs Local.
- **Modules**:
  - Use `import`, `from ... import ...`.
  - Example: `random` module to generate integers.
- **Prime Check Program**: Demonstrates user input, loops, and conditions.

Introduction to Python Programming – Part II

- **File Operations**:
  - Open: `open(filename, mode)` with `r`, `w`, `a`, `r+`.
  - Read: `read()`, Write: `write()`, Close: `close()`
  - `with open(...) as file:` for best practice.
  - CSV Handling: `csv.reader()`, `csv.writer()`
- **Image Operations**:
  - Use `Pillow (PIL)` – `pip install pillow`.

Introduction to Raspberry Pi – Part I

- **Architecture**: CPU/GPU, RAM, USB, Ethernet, GPIO.

- **Setup Requirements**:
  - HDMI, monitor, keyboard, mouse, 5V adapter, microSD card with OS.
- **GPIO**:
  - Digital input/output functionality.
  - Configurable via diagrams.
- **Initial Setup**:
  - `sudo raspi-config` for filesystem and settings.
- **Languages Supported**:
  - Python, C/C++, Java, Scratch, Ruby.
- **Applications**:
  - Media center, home automation, bots, VPNs, lightweight servers.

Raspberry Pi – Part II + Sensor & Actuator Integration

- **Blinking LED Example**:
  - Use `RPi.GPIO` and `time.sleep()` to control LED state.
- **Temperature Dependent Auto Cooling System**:
  - **DHT Sensor**: Connected to GPIO, read using `Adafruit_DHT`.
  - **Relay Control**: Activates mini-fan when temperature exceeds threshold.

In a temperature-controlled fan system using a relay, the fand should turn on when the surrounding temperature exceeds a predefined threshold.

Let me know if you want these compiled into a PDF or formatted for print!