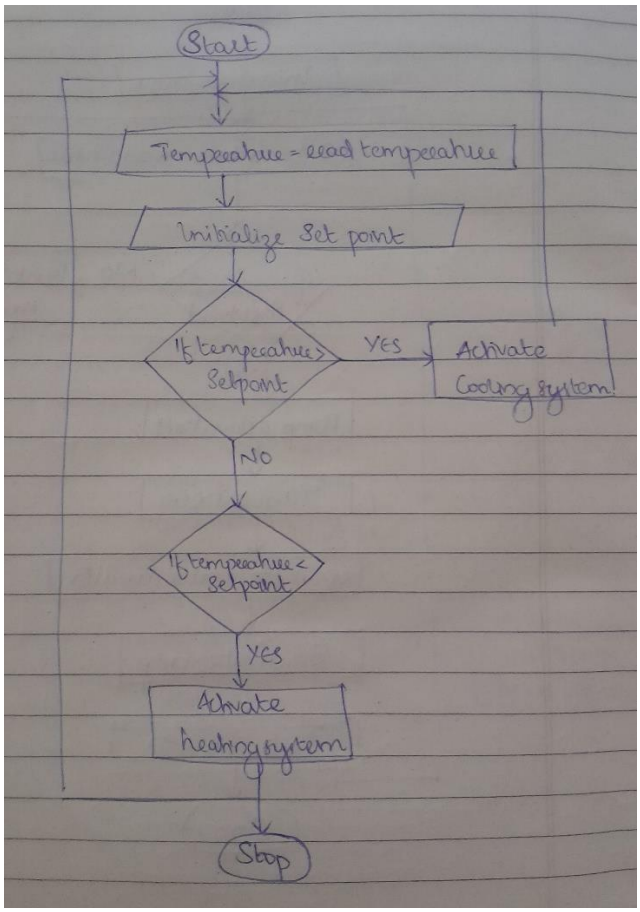
Write the Pseudocode and Flowchart for the problem statements mentioned below:

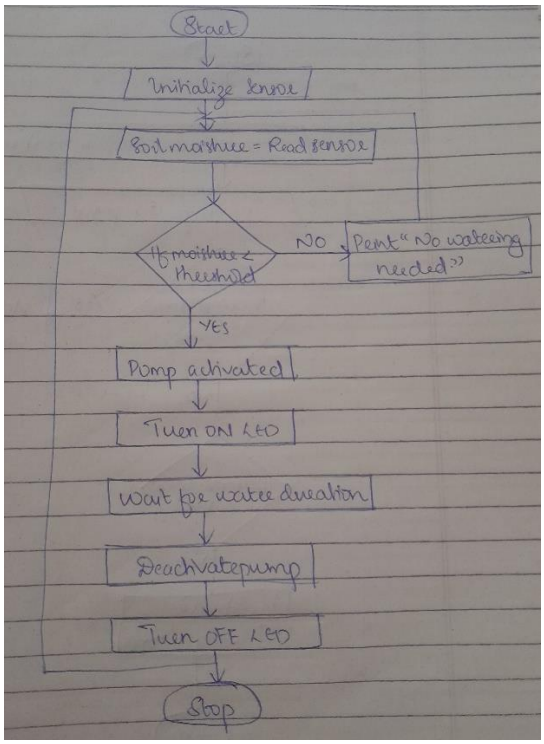
1. Smart Home Temperature Control Problem Statement: Design a temperature control system for a smart home. The system should read the current temperature from a sensor every minute and compare it to a user-defined setpoint.
   Requirements:
   • If the current temperature is above the setpoint, activate the cooling system.
   • If the current temperature is below the setpoint, activate the heating system.
   • Display the current temperature and setpoint on an LCD screen.
   • Include error handling for sensor failures.
   Pseudocode:
   1. Initialize sensor
   2. Initialize timer (set to 1 minute)
   3. Initialize set point
   4. Print current temperature and setpoint
   5. Temperature=read temperature
   6. If Temperature > setpoint, then activate cooling system
   7. Else if Temperature < setpoint, then activate heating system

   Flowchart:



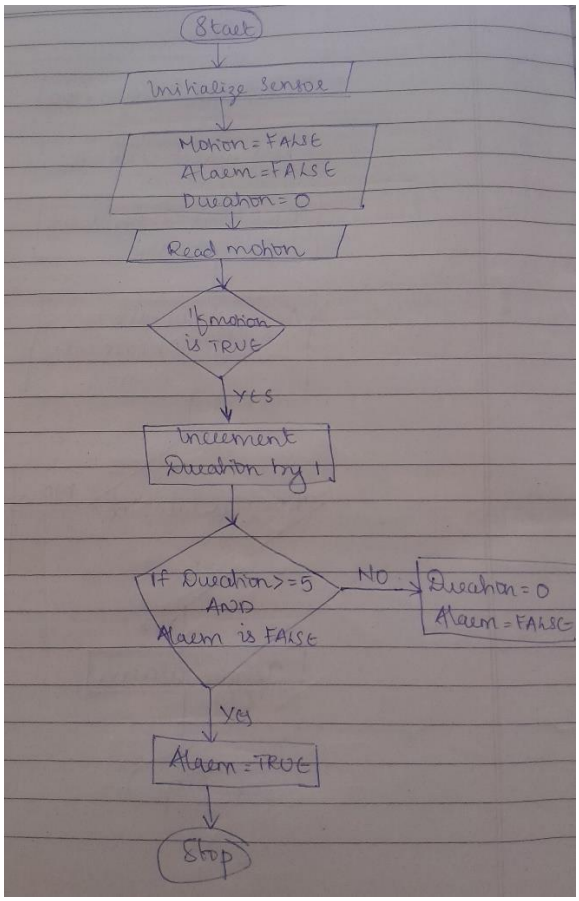
2. Automated Plant Watering System Problem Statement: Create an automated watering system for plants that checks soil moisture levels and waters the plants accordingly.
   Requirements:
   • Read soil moisture level from a sensor every hour.
   • If moisture level is below a defined threshold, activate the water pump for a specified duration.
   • Log the watering events with timestamps to an SD card.
   • Provide feedback through an LED indicator (e.g., LED ON when watering).
   Pseudocode:
   1. Initialize sensor
   2. Initialize timer
   3. Initialize threshold
   4. Soil moisture = Read sensor
   5. If soil moisture < threshold,
      activate water pump for specified duration
      get current time stamp
      write current time stamp
      turn on LED
   6. Else:
      Turn off LED

   Flowchart:



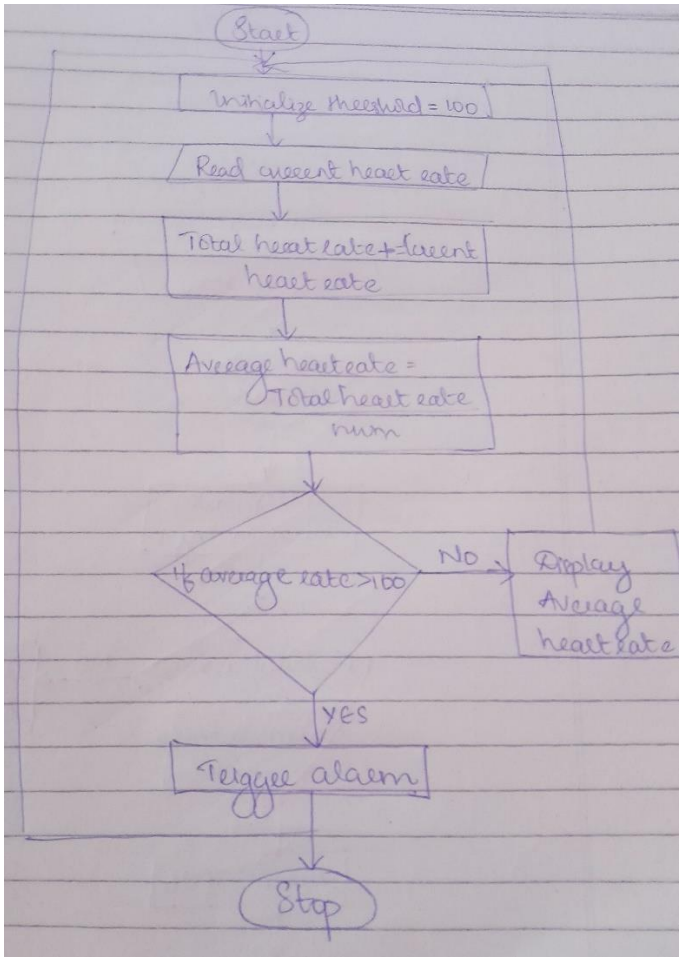
3. Motion Detection Alarm System Problem Statement: Develop a security alarm system that detects motion using a PIR sensor.
   Requirements:
   • Continuously monitor motion detection status.
   • If motion is detected for more than 5 seconds, trigger an alarm (buzzer).
   • Send a notification to a mobile device via UART communication.
   • Include a reset mechanism to deactivate the alarm.
   Pseudocode:
   1. Initialize sensor
   2. Motion=FALSE
   3. Alarm = FALSE
   4. Duration=0
   5. While TRUE do
      a. Read motion from sensor
      b. If Motion is TRUE, then
         i. Increment Duration by 1
      c. IF Duration >= 5 AND Alarm IS FALSE THEN
         i. Set Alarm = TRUE
      d. ELSE, Duration=0 set Alarm = FALSE

Flowchart:



4. Heart Rate Monitor Problem Statement: Implement a heart rate monitoring application that reads data from a heart rate sensor. Requirements:
   • Sample heart rate data every second and calculate the average heart rate over one minute.
   • If the heart rate exceeds 100 beats per minute, trigger an alert (buzzer).
   • Display current heart rate and average heart rate on an LCD screen.
   • Log heart rate data to an SD card for later analysis.
   Pseudocode:
   1. Initialize sensor
   2. Initialize HeartRate[50]
   3. Threshold=100
   4. HeartRate[i] = currentHeartRate
   5. totalHeartRate = totalHeartRate + currentHeartRate
   6. averageHeartRate = totalHeartRate / num
   7. If averageHeartRate >100
      a. Trigger alert
   8. Display currentHeartRate and averageHeartRate

Flowchart:



5. LED Control Based on Light Sensor Problem Statement: Create an embedded application that controls an LED based on ambient light levels detected by a light sensor.

   Requirements:

   • Read light intensity from the sensor every minute.

   • If light intensity is below a certain threshold, turn ON the LED; otherwise, turn it OFF.

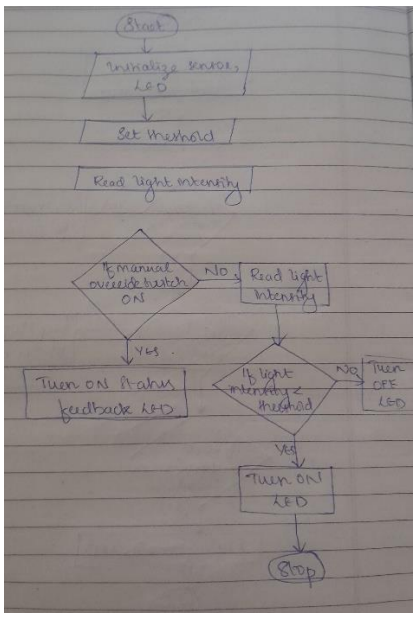
   • Include a manual override switch that allows users to control the LED regardless of sensor input.

   • Provide status feedback through another LED (e.g., blinking when in manual mode).

   Pseudocode:

   1. Initialize light sensor
   2. Initialize LED
   3. Set threshold
   4. For every minute
   5. If manual override switch is ON
      a. Turn ON status feedback LED
   6. Else
   7. Read light intensity from sensor
   8. If light intensity < threshold
      a. Turn ON LED
   9. Else
      a. Turn OFF LED

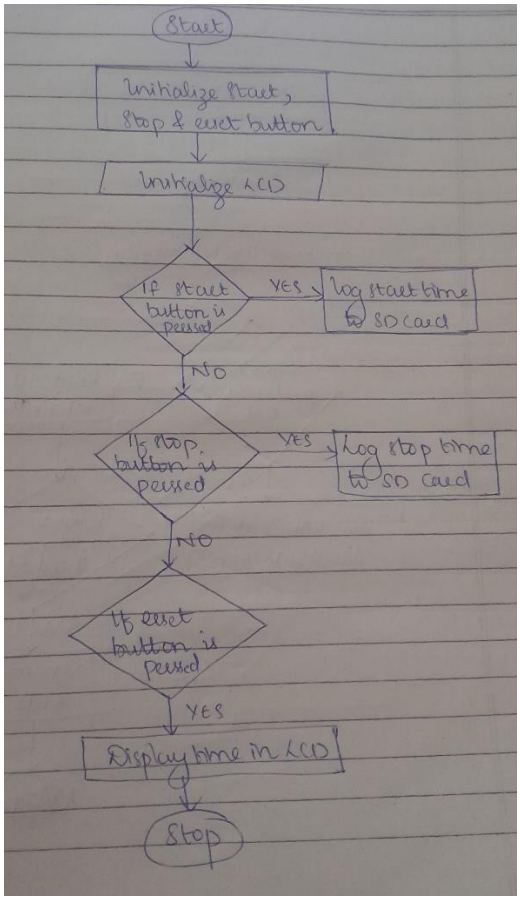
   Flowchart:



6. Digital Stopwatch Problem Statement: Design a digital stopwatch application that can start, stop, and reset using button inputs.
   Requirements:
   • Use buttons for Start, Stop, and Reset functionalities.
   • Display elapsed time on an LCD screen in hours, minutes, and seconds format.
   • Include functionality to pause and resume timing without resetting.
   • Log start and stop times to an SD card when stopped.

   Pseudocode:
   1. Initialize start button, stop button, reset button.
   2. Initialize LCD
   3. If start button is pressed, then
      a. Log start time to SD card
   4. If stop button is pressed, then
      a. Log stop time to SD card
   5. If reset button is pressed, then
      a. Display Time in LCD

   Flowchart:



7. Temperature Logging System Problem Statement: Implement a temperature logging system that records temperature data at regular intervals.

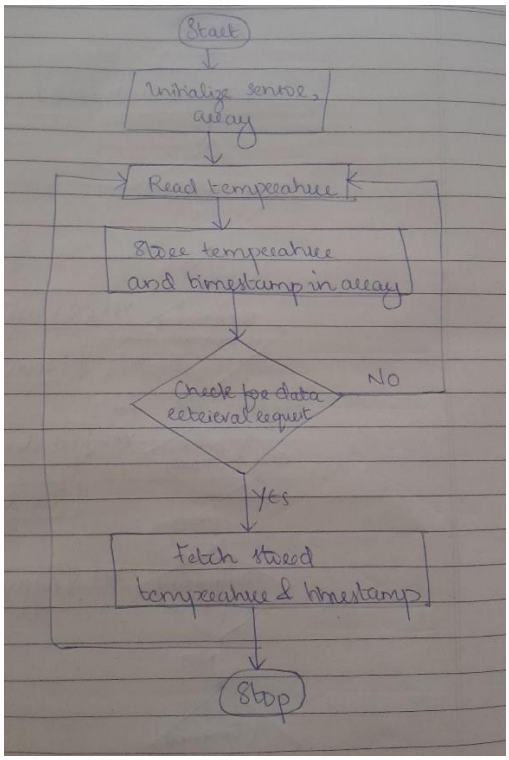
   Requirements:

   • Read temperature from a sensor every 10 minutes.
   • Store each reading along with its timestamp in an array or log file.
   • Provide functionality to retrieve and display historical data upon request.
   • Include error handling for sensor read failures.

   Pseudocode:

   1. Initialize sensor
   2. Initialize array to store temperature and timestamp
   3. For every 10 minutes
      a. Read temperature from sensor
      b. Store temperature and timestamp in array.
      c. Check for data retrieval request
      d. If YES
         i. Fetch stored temperature and timestamp
         ii. Display the historical data
      e. Else
         i. Continue reading temperature

   Flowchart:



8. Bluetooth Controlled Robot Problem Statement: Create an embedded application for controlling a robot via Bluetooth commands.

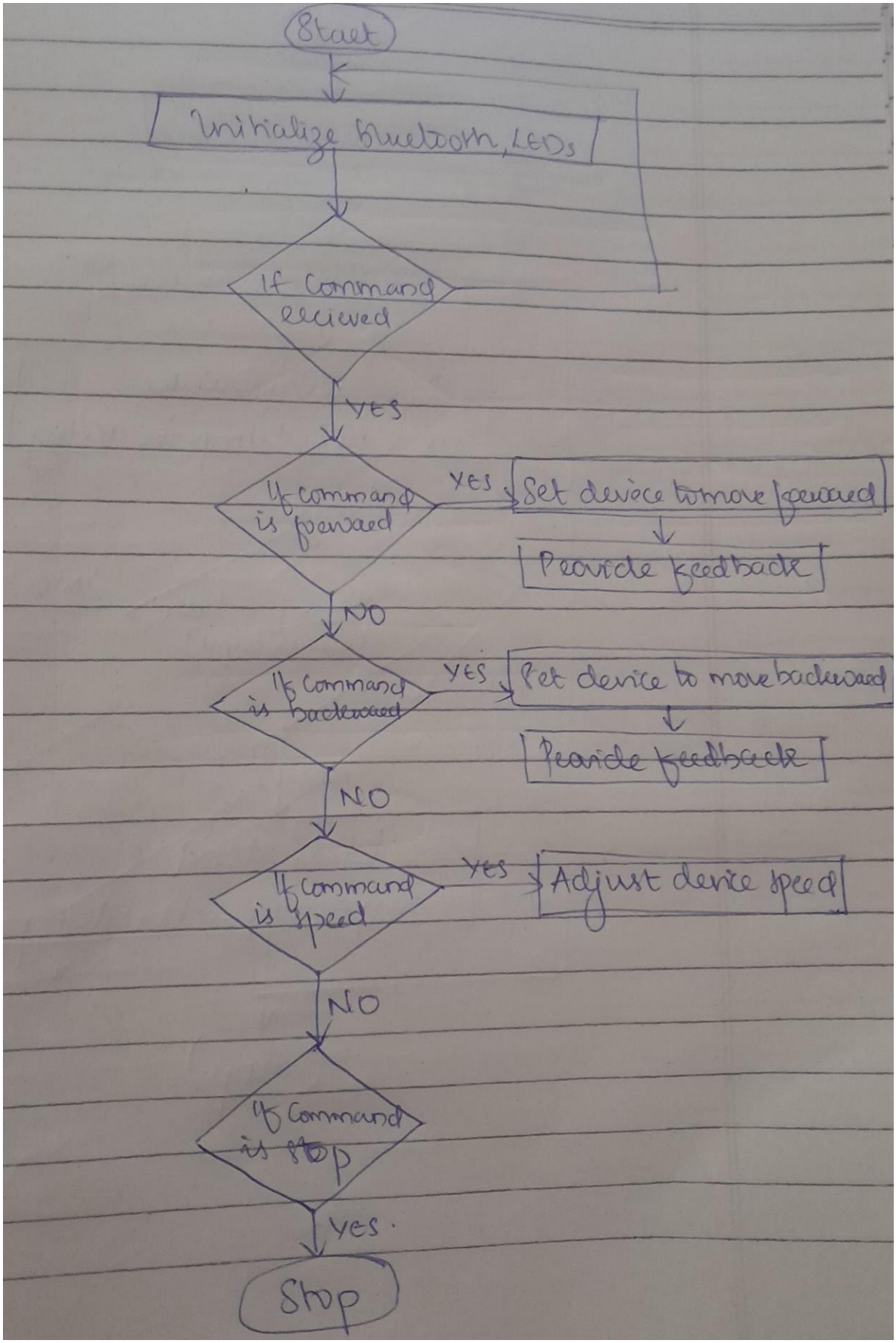
   Requirements:

   • Establish Bluetooth communication with a mobile device.
   • Implement commands for moving forward, backward, left, and right.
   • Include speed control functionality based on received commands.
   • Provide feedback through LEDs indicating the current state (e.g., moving or stopped).

   Pseudocode:

   1. Initialize blutooth module
   2. Initialize LEDs
   3. If command received
      a. If command is “Forward”
         i. Set the device to move forward
         ii. Provide feedback through LED
      b. Else if command is “Backward”
         i. Set device to move backward
         ii. Provide feedback through LED
      c. Else if command is “Left”
         i. Set device to turn Left
         ii. Provide feedback through LED
      d. Else if command is “Right”
         i. Set device to turn right
         ii. Provide feedback through LED
      e. Else if command is “Speed”
         i. Adjust device speed
      f. Else if command is “Stop”
         i. Stop the device

   Flowchart:


Start
Initialize Bluetooth, LEDs
If Command recieved
YES
If command is forward
YES
Set device to move forward
Provide feedback
NO
If Command is backward
YES
Set device to move backward
Provide feedback
NO
If Command is speed
YES
Adjust device speed
NO
If Command is stop
Yes.
Stop

9. Battery Monitoring System Problem Statement: Develop a battery monitoring system that checks battery voltage levels periodically and alerts if voltage drops below a safe threshold.

    Requirements:

    • Measure battery voltage every minute using an ADC (Analog-to-Digital Converter).

    • If voltage falls below 11V, trigger an alert (buzzer) and log the event to memory.

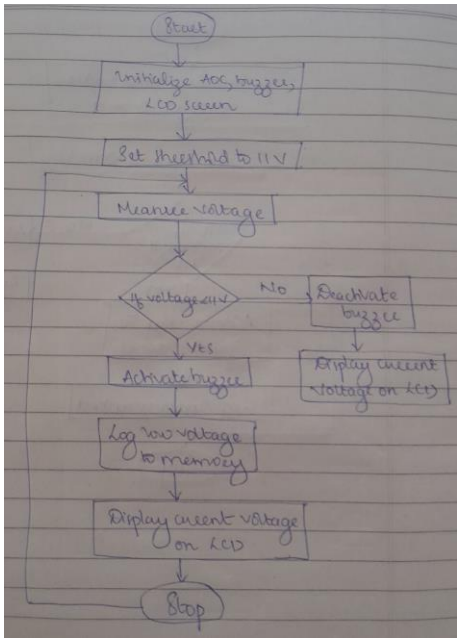
    • Display current voltage on an LCD screen continuously.

    • Implement power-saving features to reduce energy consumption during idle periods.

    Pseudocode:

    1. Initialize ADC, Buzzer, LCD screen
    2. Set voltage threshold to 11V
    3. For every minute
        a. Measure voltage using ADC
        b. Display current voltage on LCD screen
    4. If voltage < 11V, then
        a. Activate buzzer
        b. Log low voltage alert with timestamp to memory
        c. Display current voltage on LCD screen
    5. Else
        a. Deactivate buzzer
        b. Display current voltage on LCD screen

    Flowchart:



10. RFID-Based Access Control System Problem Statement: Design an access control system using RFID technology to grant or deny access based on scanned RFID tags. Requirements:
    • Continuously monitor for RFID tag scans using an RFID reader.
    • Compare scanned tags against an authorized list stored in memory.
    • Grant access by activating a relay if the tag is authorized; otherwise, deny access with an alert (buzzer).
    • Log access attempts (successful and unsuccessful) with timestamps to an SD card.
    Pseudocode:
    1. Initialize RFID reader
    2. Initialize buzzer
    3. Initialize SD card
    4. If RFID tag scanned, then
        a. Get scanned RFID tag
        b. If scanned RFID tag is in authorised list
            i. Activate relay to grant access
            ii. Log access granted with timestamp to SD card
        c. Else
            i. Activate buzzer to deny access
            ii. Log access denied with timestamp to SD card

Flowchart: