OVERVIEW - SCHEMATIC AND COMMUNICATION PROTOCOL

Wiring Schematic

Items	Model	Quantity	Power Supply
NodeMCU ESP8266	ESP8266-12E	3	4.5 to 9 VDC
Raspberry Pi (RPi)	RPi 5	1	5 to 15 VDC (27W)
Photoresistor		3	With resistors
Light Emitting Diode	RGB LEDs	4	With resistors
LED Bar Graph	10 segments	1	3.3 to 5 VDC
LED Matrix	MAX7219	1	5 VDC
Resistor	220 Ω	10	
Resistor	10k Ω	3	
Push Button	Bread-board Mount	1	

Table 1. Equipment list

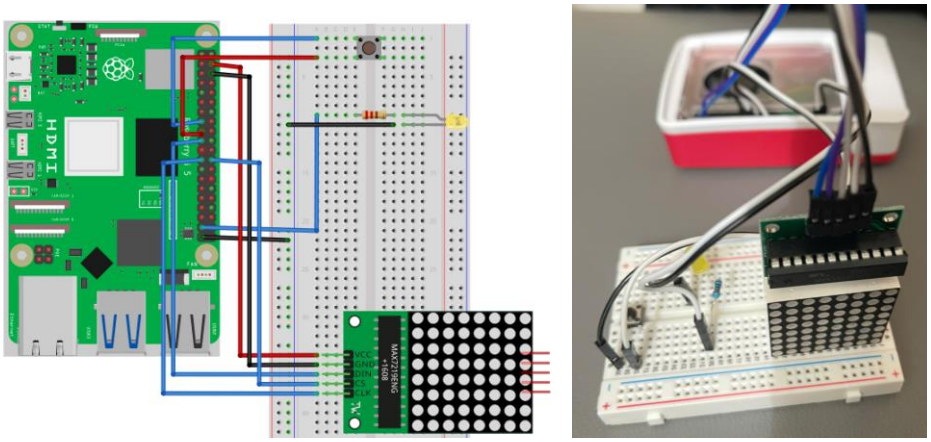


Figure 1. Wiring schematic for RPi5

- Raspberry Pi 5
  - 3.3V → Button (+)
  - GPIO 22 → Button (-)
  - GPIO 26 → Yellow LED (+)
  - 5V → LED Matrix (VCC)
  - GPIO10 (MOSI) → LED Matrix (DIN)
  - GPIO11 (SCLK) → LED Matrix (CLK)
  - GPIO8 (CE0) → LED Matrix (CS)

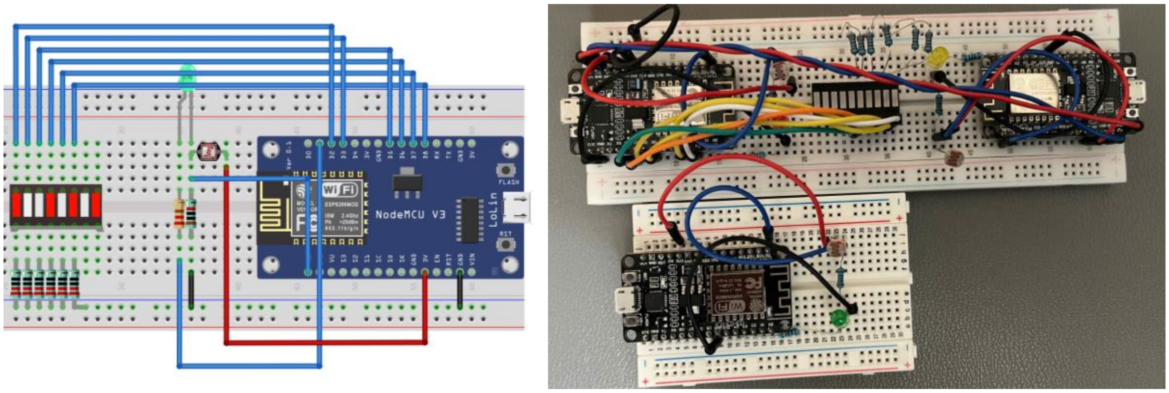


Figure 2. Wiring schematic for ESP8266

- ESP8266:
    - 3.3V → Photoresistor (+)
    - A0 → Photoresistor (+)
    - D1 → RGB Led (+)
    - D2, D3, D5, D6, D7, D8 → LED Bar Graph (1 → 6)

Communication Protocol/Message Description

Protocol Overview

- The system uses broadcast UDP messages over a mobile hotspot network to facilitate communication among multiple ESP8266 devices and a Raspberry Pi. Each message follows a specific format, allowing the devices to identify the type of message, sender, and content.

Message Format

- Each message contains:
    - Start Bit: Identifies the beginning of a message.
    - Payload: Contains specific information (e.g., sensor readings, commands).
    - End Bit: Marks the end of a message.
- General Message Structure:
    - Start Bit: "+++" | "~~~" → indicates the start of the message.
    - End Bit: "***" | "---" → indicates the end of the message.
    - Payload: Content varies depending on the message type (sensor data or control command).
- For example: +++"<PAYLOAD>"***

Communication Flow

- Communication Between ESP8266 Modules:

    - Each ESP8266 device broadcasts its Swarm ID and analog sensor reading periodically over UDP.
    - All ESP8266 devices listen to these broadcasts to compare their sensor readings and determine if they should be the Master based on the highest reading.
- Sensor Data Broadcast from ESP8266 to Raspberry Pi:
    - Each ESP8266 sends its Swarm ID (self-assigned from the last digit of its IP address) and analog sensor reading (from a light sensor) to the Raspberry Pi.
    - The RPi5 listens to incoming messages and processes each one by logging the data and triggering LED feedback based on the analog reading.
- System Reset Broadcast from RPi5 to ESP8266:
    - When the RPi5's button is pressed, it broadcasts a reset command "RESET_REQUESTED" to all ESP8266 devices.
    - ESP8266 devices, upon receiving this command, reset their internal state as required (not explicitly coded here, but implied for the reset functionality).

Detailed Message Descriptions

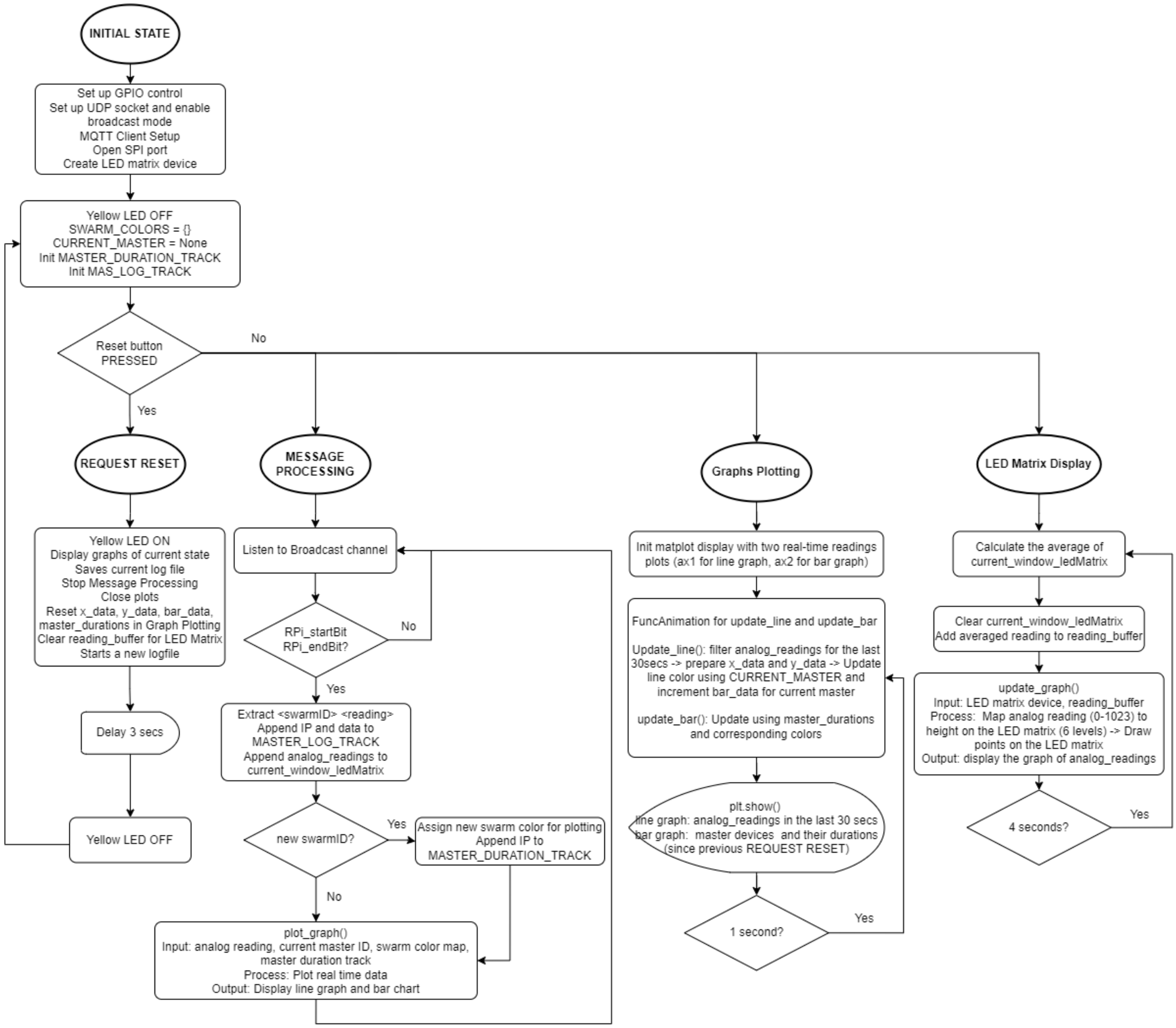
- Sensor Data Message (ESP8266 to ESP8266):
    - Start Bit: "~~~"
    - Payload: "<Swarm ID>,<Analog Reading>"
        - Swarm ID: Unique identifier for each ESP8266 (based on the last digit of its IP address).
        - Analog Reading: Light sensor reading as an integer, used to compare against other modules' readings.
    - End Bit: "---"
    - Example: If an ESP8266 with Swarm ID 4 has an analog reading of 750, the message would look like: "~~~4,750---"
- Sensor Data Message (from ESP8266 Master to RPi5):
    - Start Bit: "+++"
    - Payload: "<Swarm ID>,<Analog Reading>"
    - End Bit: "***"
- Reset Request Message (from RPi5 to all ESP8266): When the button connected to the RPi5 is pressed, it broadcasts a reset request message to all ESP8266 devices. This message has the following structure:
    - Start Bit: "+++"

- Payload: “RESET_REQUESTED”
- End Bit: "***"

Message Type	Direction	Start Bit	Payload	End Bit
Sensor Data	ESP <> ESP	“~~~”	“<SwarmID>,<AnalogReading>”	“---"
Master Data	ESP → RPi	“+++”	“<SwarmID>,<AnalogReading>”	“***”
Reset Request	RPi → ESP	“+++”	“RESET_REQUESTED”	“***”

PART 1 - RASPBERRY PI WIFI SETUP AND PACKET DELIVERY

Flowchart


INITIAL STATE
Set up GPIO control
Set up UDP socket and enable broadcast mode
MQTT Client Setup
Open SPI port
Create LED matrix device
Yellow LED OFF
SWARM_COLORS = {}
CURRENT_MASTER = None
Init MASTER_DURATION_TRACK
Init MAS_LOG_TRACK
Reset button PRESSED
No
Yes
REQUEST RESET
Yellow LED ON
Display graphs of current state
Saves current log file
Stop Message Processing
Close plots
Reset x_data, y_data, bar_data, master_durations in Graph Plotting
Clear reading_buffer for LED Matrix
Starts a new logfile
Delay 3 secs
Yellow LED OFF
MESSAGE PROCESSING
Listen to Broadcast channel
RPi_startBit RPi_endBit?
No
Yes
Extract <swarmID> <reading>
Append IP and data to MASTER_LOG_TRACK
Append analog_readings to current_window_ledMatrix
new swarmID?
Yes
Assign new swarm color for plotting
Append IP to MASTER_DURATION_TRACK
No
plot_graph()
Input: analog reading, current master ID, swarm color map, master duration track
Process: Plot real time data
Output: Display line graph and bar chart
Graphs Plotting
Init matplot display with two real-time readings plots (ax1 for line graph, ax2 for bar graph)
FuncAnimation for update_line and update_bar
Update_line(): filter analog_readings for the last 30secs -> prepare x_data and y_data -> Update line color using CURRENT_MASTER and increment bar_data for current master
update_bar(): Update using master_durations and corresponding colors
plt.show()
line graph: analog_readings in the last 30 secs
bar graph: master devices and their durations (since previous REQUEST RESET)
1 second?
Yes
LED Matrix Display
Calculate the average of current_window_ledMatrix
Clear current_window_ledMatrix
Add averaged reading to reading_buffer
update_graph()
Input: LED matrix device, reading_buffer
Process: Map analog reading (0-1023) to height on the LED matrix (6 levels) -> Draw points on the LED matrix
Output: display the graph of analog_readings
4 seconds?
Yes

Main Functionality of Raspberry Pi Code

i. Thread Descriptions

- listen_for_messages:
  - Listens for incoming UDP messages, processes analog readings from swarm devices, and logs the data.
  - Updates master tracking and publishes readings to the MQTT broker at regular intervals.

        - Filters outdated readings and assign colors to swarm IDs for plotting.

    - ledMatrix_display:
        - Displays a real-time bar graph of averaged analog readings on an LED matrix.
        - Updates every 4 seconds, managing a buffer of recent readings to maintain a consistent display.
    - monitor_button:
        - Continuously monitors the state of a physical button.
        - Triggers a reset of the system, clears logs, and creates new log files when the button is pressed.
    - plot_graph:
        - Generates and updates real-time plots of analog readings and master durations using Matplotlib.
        - Visualizes the last 30 seconds of data for analog readings and cumulative durations for master devices.

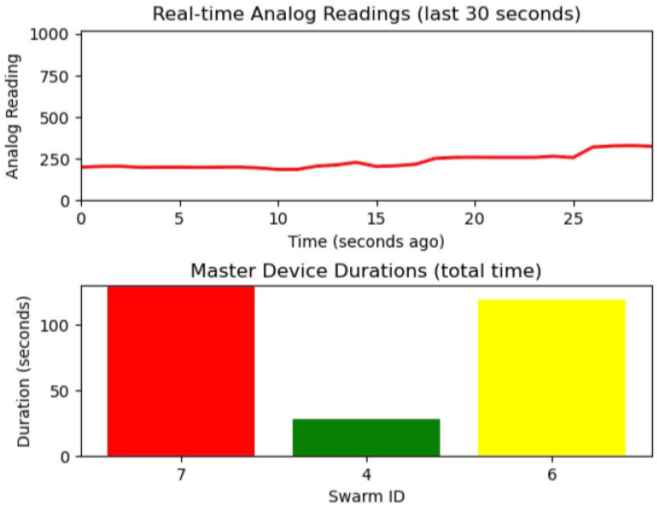


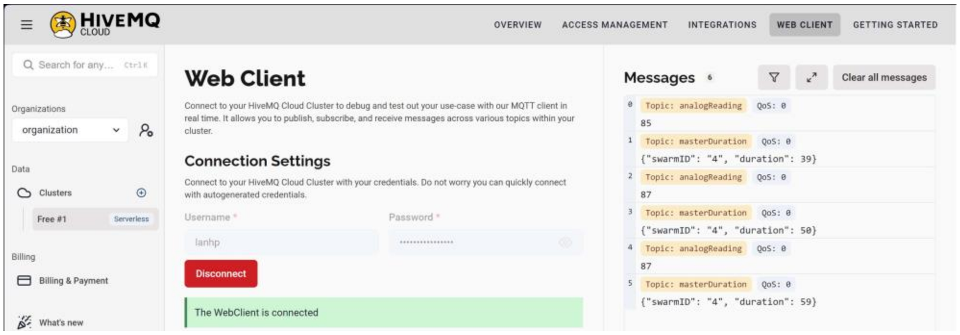
Figure 2. Real-time graphs

ii. Functionality

The RPi code facilitates the coordination of a swarm of ESP8266 devices and visualizes data.

- Input:
    - Receives UDP messages from ESP8266 devices containing sensor readings and device statuses.
    - Reads button states and accepts reset commands.
- Process:

- Logs, filters, and analyzes sensor data, determining the master device based on the highest reading.
- Maintains and displays a time-based graph on an LED matrix.
- Publishes sensor data and reset statuses to an MQTT broker for external monitoring on a NodeRED dashboard.



Web Client

Connect to your HiveMQ Cloud Cluster to debug and test out your use-case with our MQTT client in real time. It allows you to publish, subscribe, and receive messages across various topics within your cluster.

Connection Settings

Connect to your HiveMQ Cloud Cluster with your credentials. Do not worry you can quickly connect with autogenerated credentials.

Username *: lanhp

Password *: ••••••••••••••••

The WebClient is connected

Messages 6

```
0 Topic: analogReading  QoS: 0
  85
1 Topic: masterDuration  QoS: 0
  {"swarmID": "4", "duration": 39}
2 Topic: analogReading  QoS: 0
  87
3 Topic: masterDuration  QoS: 0
  {"swarmID": "4", "duration": 50}
4 Topic: analogReading  QoS: 0
  87
5 Topic: masterDuration  QoS: 0
  {"swarmID": "4", "duration": 59}
```

Figure 3. Sample data sent to MQTT broker HiveMQ

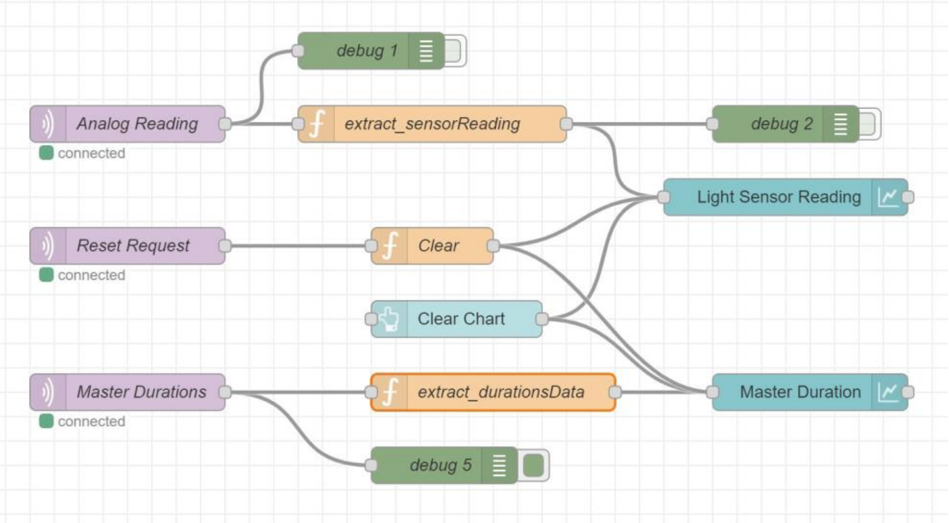


Figure 4. Flows in NodeRED

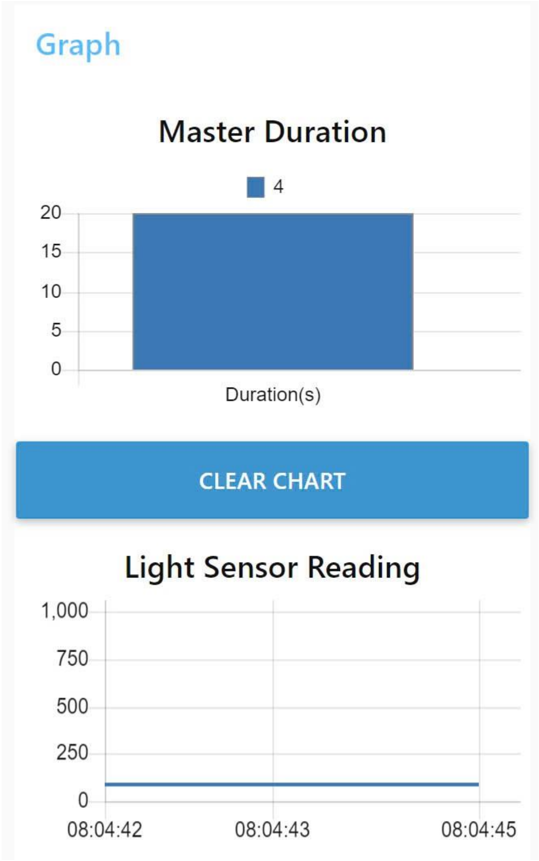
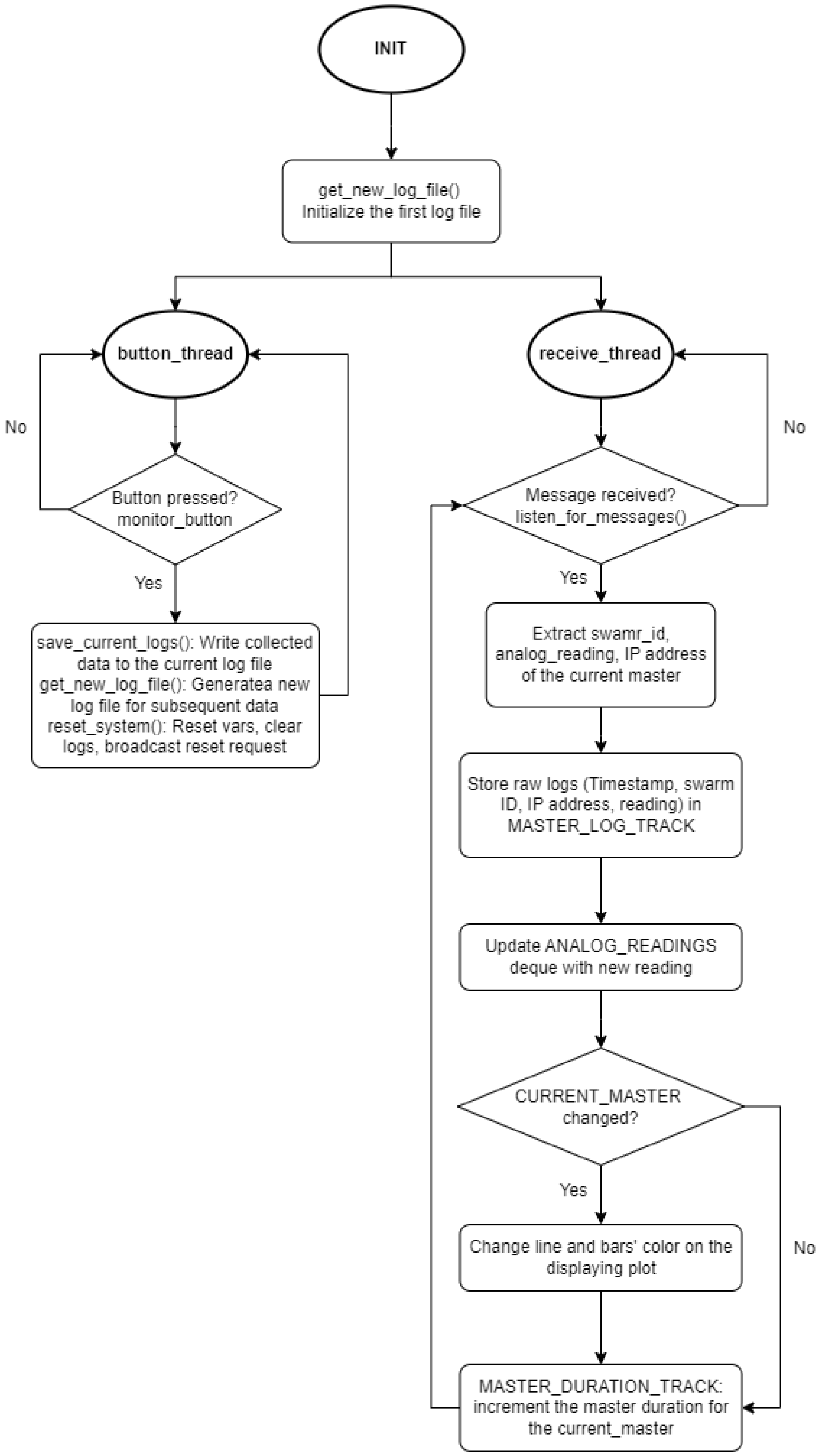


Figure 5. NodeRED dashboard

- Output:
  - Displays real-time data graphs on an LED matrix.
  - Saves system logs to files and publishes updates to MQTT topics.
  - Indicates reset events through a yellow LED.

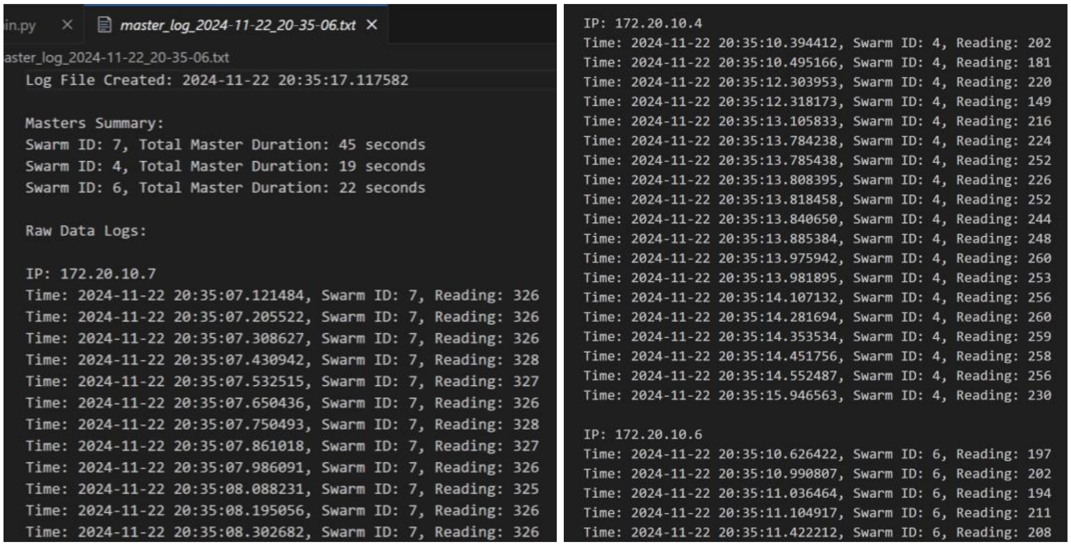
Data Structures/Log Files


INIT
get_new_log_file()
Initialize the first log file
button_thread
receive_thread
No
Button pressed?
monitor_button
Yes
save_current_logs(): Write collected data to the current log file
get_new_log_file(): Generatea new log file for subsequent data
reset_system(): Reset vars, clear logs, broadcast reset request
No
Message received?
listen_for_messages()
Yes
Extract swamr_id, analog_reading, IP address of the current master
Store raw logs (Timestamp, swarm ID, IP address, reading) in MASTER_LOG_TRACK
Update ANALOG_READINGS deque with new reading
CURRENT_MASTER changed?
Yes
No
Change line and bars' color on the displaying plot
MASTER_DURATION_TRACK: increment the master duration for the current_master

- Key Data Structures and Fields:
    - MASTER_LOG_TRACK (dict):
        - Stores logs of data received from each device (IP-based).
        - Fields: IP address, log entry (timestamp, swarm_id, analog reading).
    - MASTER_DURATION_TRACK (defaultdict):

        - Tracks the duration (in seconds) each Swarm ID has been the master.
        - Fields: swarm_id, total duration (seconds).
    - SWARM_COLORS (dict):
        - Assigns a color to each Swarm ID.
        - Fields: swarm_id, assigned color (red, green, yellow).
    - ANALOG_READINGS (deque):
        - Stores the last 30 analog readings from the devices.
        - Field: analog reading value.

- Log File:
    - LOG_FILE (str): The current log file name, dynamically generated based on the timestamp of file creation.
    - Log Content: Includes a summary of all devices that became masters (IP addresses), how long each device was a master (from the beginning), and raw data from each master.


in.py
master_log_2024-11-22_20-35-06.txt
aster_log_2024-11-22_20-35-06.txt

```
Log File Created: 2024-11-22 20:35:17.117582

Masters Summary:
Swarm ID: 7, Total Master Duration: 45 seconds
Swarm ID: 4, Total Master Duration: 19 seconds
Swarm ID: 6, Total Master Duration: 22 seconds

Raw Data Logs:

IP: 172.20.10.7
Time: 2024-11-22 20:35:07.121484, Swarm ID: 7, Reading: 326
Time: 2024-11-22 20:35:07.205522, Swarm ID: 7, Reading: 326
Time: 2024-11-22 20:35:07.308627, Swarm ID: 7, Reading: 326
Time: 2024-11-22 20:35:07.430942, Swarm ID: 7, Reading: 328
Time: 2024-11-22 20:35:07.532515, Swarm ID: 7, Reading: 327
Time: 2024-11-22 20:35:07.650436, Swarm ID: 7, Reading: 326
Time: 2024-11-22 20:35:07.750493, Swarm ID: 7, Reading: 328
Time: 2024-11-22 20:35:07.861018, Swarm ID: 7, Reading: 327
Time: 2024-11-22 20:35:07.986091, Swarm ID: 7, Reading: 326
Time: 2024-11-22 20:35:08.088231, Swarm ID: 7, Reading: 325
Time: 2024-11-22 20:35:08.195056, Swarm ID: 7, Reading: 326
Time: 2024-11-22 20:35:08.302682, Swarm ID: 7, Reading: 326
```

```
IP: 172.20.10.4
Time: 2024-11-22 20:35:10.394412, Swarm ID: 4, Reading: 202
Time: 2024-11-22 20:35:10.495166, Swarm ID: 4, Reading: 181
Time: 2024-11-22 20:35:12.303953, Swarm ID: 4, Reading: 220
Time: 2024-11-22 20:35:12.318173, Swarm ID: 4, Reading: 149
Time: 2024-11-22 20:35:13.105833, Swarm ID: 4, Reading: 216
Time: 2024-11-22 20:35:13.784238, Swarm ID: 4, Reading: 224
Time: 2024-11-22 20:35:13.785438, Swarm ID: 4, Reading: 252
Time: 2024-11-22 20:35:13.808395, Swarm ID: 4, Reading: 226
Time: 2024-11-22 20:35:13.818458, Swarm ID: 4, Reading: 252
Time: 2024-11-22 20:35:13.840650, Swarm ID: 4, Reading: 244
Time: 2024-11-22 20:35:13.885384, Swarm ID: 4, Reading: 248
Time: 2024-11-22 20:35:13.975942, Swarm ID: 4, Reading: 260
Time: 2024-11-22 20:35:13.981895, Swarm ID: 4, Reading: 253
Time: 2024-11-22 20:35:14.107132, Swarm ID: 4, Reading: 256
Time: 2024-11-22 20:35:14.281694, Swarm ID: 4, Reading: 260
Time: 2024-11-22 20:35:14.353534, Swarm ID: 4, Reading: 259
Time: 2024-11-22 20:35:14.451756, Swarm ID: 4, Reading: 258
Time: 2024-11-22 20:35:14.552487, Swarm ID: 4, Reading: 256
Time: 2024-11-22 20:35:15.946563, Swarm ID: 4, Reading: 230

IP: 172.20.10.6
Time: 2024-11-22 20:35:10.626422, Swarm ID: 6, Reading: 197
Time: 2024-11-22 20:35:10.990807, Swarm ID: 6, Reading: 202
Time: 2024-11-22 20:35:11.036464, Swarm ID: 6, Reading: 194
Time: 2024-11-22 20:35:11.104917, Swarm ID: 6, Reading: 211
Time: 2024-11-22 20:35:11.422212, Swarm ID: 6, Reading: 208
```

Figure 6. Log content example

- Data Flow:
    - Input: Messages received from devices (IP address, Swarm ID, analog reading).
    - Process: Log the data, assign a color to Swarm IDs, track the master device, update readings.
    - Output:
        - Log file containing master durations and raw data logs.
        - Real-time plots with updated readings and master durations.