LoRaWAN Command Documentation for IO Board

Overview

This document provides details on the commands used to control a LoRaWAN device, specifically for managing PWM (Pulse Width Modulation) settings and relay states. The device supports various commands to set PWM frequency, duty cycles, and the state of relays.

Command Structure

The commands are sent in the form of JSON objects, which are then encoded into byte arrays for transmission over LoRaWAN. Below are the available commands and their corresponding JSON structures.

1. Set PWM Frequency and Duty Cycles

Command: `setPwm`

Description: This command sets the PWM frequency and the duty cycles for two channels.

JSON Structure:

```
{
  "command": "setPwm",
  "frequency": <frequency_value>,      // Frequency in Hz (1 - 100000)
  "c1_dutyCycle": <duty_cycle_1>,      // Duty cycle for Channel 1 (0 - 100)
  "c2_dutyCycle": <duty_cycle_2>       // Duty cycle for Channel 2 (0 - 100)
}
```

2. Set a Single Relay State

Command: `setRelay`

Description: This command sets the state of a specified relay.

JSON Structure:

```
{
  "command": "setRelay",
  "relay": <relay_number>,               // Relay number (0 - 3)
  "state": <state_value>                  // Relay state (0 for off, 1 for on)
}
```

3. Set PWM Frequency Only

Command: `setPwmFrequency`

Description: This command sets the PWM frequency without changing the duty cycles.

JSON Structure:

```
{
  "command": "setPwmFrequency",
  "frequency": <frequency_value>         // Frequency in Hz (1 - 100000)
}
```

4. Set Duty Cycle for a Specific Channel

Command: `setPwmDutyCycle`

Description: This command sets the duty cycle for a specific PWM channel.

JSON Structure:

```
{
  "command": "setPwmDutyCycle",
  "channel": <channel_number>,           // Channel (1 or 2)
  "dutyCycle": <duty_cycle_value>        // Duty cycle (0 - 100)
}
```

Encode Function

The following JavaScript function is used to convert the JSON objects into byte arrays for transmission:

```
function Encode(fPort, obj, variables) {
  var bytes = [];

  // Command to set PWM frequency and duty cycles
  if (obj.command === "setPwm") {
    bytes.push(0x01); // Command for setting PWM frequency and duty cycles
    bytes = bytes.concat(encodeUint32(obj.frequency)); // Frequency
    bytes = bytes.concat(encodeUint32(obj.c1_dutyCycle)); // Channel 1 duty cycle
    bytes = bytes.concat(encodeUint32(obj.c2_dutyCycle)); // Channel 2 duty cycle
  }

  // Command to set a single relay state
  else if (obj.command === "setRelay") {
    bytes.push(0x03); // Command for setting a single relay
    bytes.push(obj.relay); // Relay number (0-3)
    bytes.push(obj.state); // Relay state (0 for off, 1 for on)
  }
```

```
 // Command to set only the PWM frequency
  else if (obj.command === "setPwmFrequency") {
    bytes.push(0x04); // Command for setting PWM frequency
    bytes = bytes.concat(encodeUint32(obj.frequency)); // Frequency
  }

  // Command to set only the duty cycle of a specific channel
  else if (obj.command === "setPwmDutyCycle") {
    bytes.push(0x05); // Command for setting PWM duty cycle
    bytes.push(obj.channel); // Channel (1 or 2)
    bytes = bytes.concat(encodeUint32(obj.dutyCycle)); // Duty cycle
  }

  return bytes;
}

function encodeUint32(value) {
  return [(value & 0xFF), (value >> 8 & 0xFF), (value >> 16 & 0xFF), (value >> 24 &
0xFF)];
}
```

Explanation of the Encode Function:

- The `Encode` function takes three parameters: `fPort`, `obj`, and `variables`.
- It initializes an empty array, `bytes`, to hold the encoded values.
- Depending on the `command` in the JSON object, it adds appropriate command bytes and values to the `bytes` array.
- The function handles the encoding for setting PWM configurations, relay states, and duty cycles, ensuring that values are within the specified ranges.

Conclusion

This document serves as a guide for using the LoRaWAN commands to control the device effectively. It includes the structure of commands, the encoding function, and necessary parameters to ensure correct operation.