

Power Configuration

- [Meshtastic Power Settings Reference](#)
- [Power Configuration Settings Reference](#)

Meshtastic Power Settings Reference

The Power configuration section in Meshtastic controls sleep modes, charge management, and power-related behavior. These settings are critical for battery and solar-powered deployments.

Accessing Power Settings

In the app: **Radio Config** → **Power**

Via CLI: `meshtastic --get power`

Is Power Saving Enabled

```
meshtastic --set power.is_power_saving true
```

When enabled, the node reduces power consumption by putting the CPU into light sleep between receive windows. In light sleep the CPU is suspended but the LoRa radio stays on, so the node can still hear and respond to traffic. This is appropriate for CLIENT role nodes that send and receive messages but want longer battery life.

For the **ROUTER** role, `power.is_power_saving` is force-enabled automatically (on ESP32) and cannot be turned off — you do not set it manually. Even with power saving on, the LoRa radio remains in standby and wakes on incoming packets, so the node continues to relay; light sleep (radio standby) does not prevent relaying. **REPEATER is deprecated as of firmware 2.7.11**; for infrastructure prefer **ROUTER** (or ROUTER_LATE).

On Battery Discharge Values

These settings affect how the node reports battery state-of-charge from measured voltage. The defaults work for LiPo batteries; the ADC multiplier may need adjustment for LiFePO4 or for boards whose voltage divider reads inaccurately:

Setting	Default	Notes
---------	---------	-------

adc_multiplier_override	0 (auto)	May need tuning per board for accurate voltage reading, including LiFePO4 packs
-------------------------	----------	---

Meshtastic does **not** provide a generic “shut down after N seconds at a critical-voltage threshold” key. (The firmware option `power.on_battery_shutdown_after_secs` shuts the device down after *external power is lost* for N seconds — it is a power-loss timer, not a low-voltage cutoff — and defaults to `0` = disabled.) For low-battery protection rely on the firmware’s built-in low-battery handling rather than a fabricated voltage-cutoff timer.

Sleep Configuration

For ESP32-based nodes, Meshtastic uses light sleep between activity to reduce power consumption. Note that the older super-deep-sleep keys (`power.sds_secs` and `power.mesh_sds_timeout_secs`) have been removed/deprecated and are no longer part of the current, documented power config — do not rely on them.

```
# Enable power saving (CLIENT nodes; ROUTER force-enables it automatically)
meshtastic --set power.is_power_saving true

# Light sleep interval. A value of 0 means the firmware default (5 minutes).
meshtastic --set power.ls_secs 0

# Minimum wake time - stay awake at least this long after waking
meshtastic --set power.min_wake_secs 10
```

Screen and BT Power

```
# Screen timeout (0 = always off)
meshtastic --set display.screen_on_secs 30

# Bluetooth enable/disable
meshtastic --set bluetooth.enabled true

# BLE pairing mode (RANDOM_PIN, FIXED_PIN, or NO_PIN)
meshtastic --set bluetooth.mode RANDOM_PIN
```

Use `RANDOM_PIN` (the secure default) unless you have a specific reason not to. **Setting** `NO_PIN` **disables BLE pairing security** and lets any nearby device connect without a PIN — only use it on physically secured nodes. (These Bluetooth options live under `bluetooth.*`, not the Power config, but are included here because they affect power draw.)

Power Consumption by Role

Meshtastic does not publish per-board current-draw figures; actual milliamp draw is highly board-, display-, and firmware-dependent. The official device-role table describes power qualitatively (Regular / Low / High):

Role	Power saving	Relative power level
CLIENT	Optional (user-set)	Regular
CLIENT_MUTE	Optional (user-set)	Low
ROUTER	Force-enabled on ESP32 (LoRa radio in standby, wakes on packets)	High
REPEATER (deprecated)	Yes	High

Peripherals add to whatever the role draws: an OLED/TFT display and WiFi both materially increase ESP32 power (WiFi is ESP32-only and is not available on nRF52). For exact numbers, measure your specific board or consult its datasheet rather than relying on generic figures.

Practical Power Optimization Checklist

- Set role correctly: CLIENT for personal nodes (enable power saving for longer battery life); ROUTER for fixed infrastructure (power saving is automatic)
- Set screen timeout to 30 seconds or disable (`screen_on_secs 0`)
- On ESP32, note that enabling WiFi disables Bluetooth (the WiFi setting takes precedence)
- Set position broadcast to 30+ minutes for fixed nodes
- Set telemetry broadcast interval up (the device telemetry default is 1800 s / 30 min)
- Reduce TX power to the minimum needed for coverage. For the legal transmit-power ceiling (US/Canada: 1 W / 30 dBm conducted, with EIRP limits for high-gain antennas), see the LoRa Settings Reference.
- Use nRF52840 hardware over ESP32 for substantially better battery life on the same battery

Power Configuration Settings Reference

Meshtastic's power management settings control how your node balances battery life against responsiveness. Understanding these settings is essential for field deployments and battery-powered infrastructure.

Key Power Settings

Setting	Default	Description
<code>power.is_power_saving</code>	false	Enable aggressive power saving (see below)
<code>power.adc_multiplier_override</code>	0	Override ADC calibration for voltage reading (float, typically ~2-6; 0 = use firmware default)
<code>power.wait_bluetooth_secs</code>	60	Keep BLE active for N seconds after last connection
<code>power.ls_secs</code>	0	Light sleep interval (ESP32 only). A default of 0 means the firmware uses its built-in 5-minute light-sleep interval; it does NOT mean "no light sleep." No effect on nRF52/RP2040.
<code>power.min_wake_secs</code>	10	Minimum time to stay awake after waking from sleep

Note: the older `power.sds_secs` (super deep sleep) and `power.mesh_sds_timeout_secs` keys, and `power.on_battery_shutdown_after_secs`, are no longer part of the current power config and should not be set. Verify any power key against the current firmware config protobuf before using it.

`is_power_saving` Mode

When enabled, the node aggressively reduces power consumption:

- Turns off WiFi when not actively connecting. (Caution: on a WiFi/MQTT gateway this can disrupt the connection — leave power saving off for always-on gateways.)

- Turns off BLE after `wait_bluetooth_secs` with no activity
- Reduces CPU clock speed when idle
- Turns off the display backlight more aggressively

Recommended for: Battery-operated client nodes, portable nodes used intermittently

NOT recommended for: Infrastructure repeaters, always-on gateway nodes

Light Sleep vs Deep Sleep

Meshtastic supports power-saving sleep behavior on ESP32 hardware:

- **Light sleep (`ls_secs`)** - Node sleeps briefly between LoRa receive windows. LoRa radio stays partially active and can wake on a received packet. Moderate power reduction at the cost of slight receive latency. (ESP32 only.)
- **Deep sleep** - On ESP32 the node can drop into a much deeper low-power state during long idle periods, drawing very little power but missing incoming messages until it wakes. This is most useful for sensor-style nodes that transmit periodically and don't need to receive continuously.

Battery Voltage Monitoring

Meshtastic reports battery voltage and calculated charge percentage via the telemetry system. The voltage reading depends on the hardware's ADC (Analog-to-Digital Converter) and a calibration factor:

```
# If battery percentage reads incorrectly, adjust ADC multiplier:
# First, measure actual battery voltage with a multimeter
# Then compare to what meshtastic reports:
meshtastic --info    # reported battery voltage appears under the node's deviceMetrics

# Adjust if needed (value is a float multiplier, typically ~2-6):
meshtastic --set power.adc_multiplier_override 1.05
```

ESP32 vs nRF52 Power Comparison

The figures below are **approximate community measurements** intended for rough comparison only. Actual current draw varies with board, firmware, peripherals (GPS, display), and LoRa preset.

For authoritative numbers consult the relevant chip datasheets (Espressif ESP32/ESP32-S3, Nordic nRF52840) and the Semtech SX126x receive-current spec.

Chip	Idle Current	LoRa RX	LoRa TX @22 dBm	WiFi Active
ESP32 (T-Beam)	~80 mA	~50 mA	~120 mA	+100-200 mA
ESP32-S3 (T-Beam Supreme)	~30 mA	~25 mA	~90 mA	+80-150 mA
nRF52840 (RAK4631)	~0.5 mA	~4 mA	~80 mA	N/A (no WiFi)

Note: 22 dBm here is a typical board hardware output level used for this benchmark, which is distinct from the 30 dBm (1 W) conducted FCC legal maximum for 902-928 MHz.

The nRF52840 platform has dramatically lower idle and receive current, which is why RAK4631-based nodes generally achieve significantly longer battery life (often several times) than ESP32 nodes in typical deployment scenarios.

Recommended Configurations by Use Case

```
# Portable client node (maximize battery life):
meshtastic --set power.is_power_saving true
meshtastic --set power.wait_bluetooth_secs 30
meshtastic --set position.position_broadcast_secs 300 # 5 min GPS broadcast

# Fixed indoor node (always accessible):
meshtastic --set power.is_power_saving false
meshtastic --set power.wait_bluetooth_secs 0 # always keep BLE on

# Solar outdoor node (balance receive and power):
meshtastic --set power.is_power_saving true
meshtastic --set power.ls_secs 30 # light sleep 30s intervals (ESP32)
meshtastic --set power.min_wake_secs 15
```