Devices

Robosoft

User guide

v0.1.0-r459

Notice

8devices reserves the right to change specifications without prior notice. While the information in this manual has been compiled with great care, it may not be deemed an assurance of product characteristics. 8devices shall be liable only to the degree specified in the terms of sale and delivery. The reproduction and distribution of the documentation and software supplied with this product and the use of its contents are subject to written authorization from 8devices.

Trademarks

8devices logo is a trademark of 8devices. All other registered and unregistered trademarks in this document are the sole property of their respective owners.

Table Of Contents

1. Supported Features

System Core:

- Linux 6.6 kernel
- OpenWrt 24.10 base system
- ath9k driver for QCA4531 radio
- Dual-boot firmware with automatic failsafe
- procd service manager

Radio Capabilities:

- QCA4531 radio: 2.4 GHz operation
- Narrow channel control
- Frequency shift control
- Extended frequency range
- Extended channels numeration
- Frequency spectrum active scan
- TX parameters locaking
- SIFS and slot timing controls

Wi-Fi Operating Modes:

- BSS Access Point (AP) mode
- BSS Station (STA) mode
- NAW mesh networking mode
- WFB long-range broadcast mode
- Monitor mode
- Scanner mode

Management & Monitoring:

- Web management interface
- Single YAML configuration file
- Persistent configuration storage (sysconf)
- Rapid reconfiguration support
- Factory reset support
- SSH service
- Temperature sensor monitoring
- WFB link statistics collection
- Diagnostic data collection
- GPIO FAN control
- GPIO switch UART port control
- GPIO button reboot/reset
- USB device provisioning
- USB serial adapters management

Serial Port (UART):

- TCP/UDP network access to serial ports
- UDP streaming mode for telemetry
- USB serial adapter support (CH341, FTDI, PL2303)

2. Software Releases

2.1 v0.1.0

Product compatibility beta release for usage.

Features:

- Robosoft WEB management interface
- Web troubleshoot collection and retrieval
- Full frequency range spectral scan support
- Wi-Fi scanner mode support for active scanning
- Wi-Fi monitor mode support for envionment debugging
- Wi-Fi bitrate and guard interval locking
- Wi-Fi TX retries override control
- Wi-Fi WMM parameters control
- Wi-Fi timing control (slot time, SIFS)
- Wi-Fi ACK timeout control
- Wi-Fi radio CCA threshold control
- Extended Wi-Fi radio statistics counting
- Diagnostic troubleshooting archive collection
- Developer console UART reuse as general-purpose serial port
- USB flash drive system provisioning
- USB serial adapters auto detection and provisioing
- Robosoft Lima board support

Improvements:

- Optimised boot time to ~28sec (first boot ~37sec)
- Reduced lower band edge frequency to 2202MHz to improved reliability
- Changed default Wi-Fi encryption to WPA2-PSK for broader device compatibility
- Improved Wi-Fi radio channel on-the-fly control
- NAW/WFB auto-channel fallback to channel 6
- Wi-Fi automatic MCS/GI with fallback for WFB mode
- Unified robosoft configuration controls
- Updated push button device reboot/reset triggering

Known Issues:

- DHCP server mode not supported
- LED#2 does not turn on during NAW connection
- Incorrect country displayed for radio wiphy
- Shell access banner shows incorrect CPU usage on boot
- WFB HT40+ mode transmission does not work

2.2 v0.0.2

Product beta release for trials.

Features:

- Extended Wi-Fi channels support
- Extended radio calibration data support
- Fixed channel support in STA mode
- WFB link statistics collection and monitoring
- USB serial adapter support (CH341, FTDI, PL2303)
- Serial port (UART) configuration with TCP/UDP network streaming
- GPIO FAN configuration control

Improvements:

- Improved extended channel frequency handling
- Updated BSS mode extended channels scan timeouts
- Added 40MHz channel width support in WFB mode

Known Issues:

- DHCP server mode not supported
- No support for software GPIO buttons/switches
- LED#2 does not turn on during NAW connection
- Incorrect country displayed for radio wiphy
- Shell access banner shows incorrect CPU usage on boot

2.3 v0.0.1

Initial demo release.

Features:

- Dual-boot firmware update mechanism
- YAML-based configuration system (roboconf)
- Configuration persistence to flash (sysconf)
- Wi-Fi operation modes: AP, STA, NAW, WFB
- Network zones with automatic bridging

Known Issues:

- DHCP server mode not supported
- No support for software GPIO buttons/switches
- LED#2 does not turn on during NAW connection
- WFB wfb-ng link does not send packets when HT40+
- Incorrect country displayed for radio wiphy
- Access banner shows incorrect CPU usage on boot

3. Robosoft Software Introduction

3.1 Overview

Robosoft is an embedded Linux software platform based on OpenWrt, designed for wireless communication systems and unmanned vehicle applications. The platform provides integrated device management, flexible network configuration, multiple wireless connectivity modes, and robust firmware update capabilities.

The MACA2 platform targets system integrators who require a reliable, configurable wireless node that can be deployed as a black-box component in larger systems.

3.2 Main Features

3.2.1 Network Management

The system provides flexible network configuration through network zones. Network zones are logical network segments that can include multiple interfaces (Ethernet, WiFi). The system automatically creates bridges when multiple interfaces belong to the same zone, enabling seamless Layer-2 connectivity.

Network zone features:

- Multiple independent network segments (data, management)
- Automatic bridge creation for multi-interface zones
- Support for static IP, DHCP client, and DHCP server modes
- Interface inheritance of zone network settings

3.2.2 WiFi Connectivity

The platform supports comprehensive WiFi capabilities through three distinct operating modes:

WiFi modes:

- **BSS (Basic Service Set)** - Standard WiFi operation
- **Access Point (AP)** - Broadcast WiFi network for client connections
- **Station (STA)** - Connect to existing WiFi networks
- **NAW (Non-Associated WiFi)** - Mesh networking for 2-node deployments
- **WFB (WiFi Broadcast)** - Long-range packet radio for video/data links
- **SCAN** - Dedicated mode for active spectral scanning
- **Monitor interface** - Passive monitoring, enabled independently of radio mode

WiFi features:

- Configurable channel, bandwidth (20/40 MHz), and TX power
- Fixed MCS and guard interval control with automatic mode
- WMM, and timing parameter control
- On-the-fly channel change for AP and mesh interfaces
- WPA2 security for BSS and NAW modes
- Forward Error Correction (FEC) for WFB mode
- Spectral scanning for RF spectrum analysis
- Real-time statistics and signal monitoring

3.2.3 Configuration System

Configuration is managed through Roboconf, a YAML-based system that provides:

- Simple YAML configuration file with all settings
- Automatic backfilling of missing parameters from defaults
- Dry-run mode for testing changes before applying
- Persistent storage to flash memory via sysconf
- Factory reset capability

3.2.4 Software Updates

The system implements dual-boot firmware with automatic failsafe:

- Two firmware partitions (firmware1, firmware2)
- Updates written to inactive partition
- Automatic boot partition switching after update
- Bootcount-based failsafe with rollback on boot failures
- No interruption to running system during update installation

3.2.5 System Monitoring

Monitoring capabilities include:

- Real-time WiFi statistics (signal strength, noise, bitrate)
- System information (CPU, memory, uptime)
- Temperature monitoring via onboard sensor
- Diagnostic data collection for troubleshooting
- Service status monitoring

3.2.6 System Management

- **Push button** - Short press (<1s) reboots the device, long press (>5s) performs factory reset
- **Slide switch** - Selects UART port operation mode: developer console or general-purpose UART

3.2.7 User Interfaces

Web Interface - Browser-based web management over network (WiFi or Ethernet) for system management and monitoring.

Command Line - SSH or serial console access for configuration and monitoring.

3.2.8 Service Management

The system uses OpenWrt's procd for service initialization and management. Services can be started, stopped, and restarted through init scripts.

3.3 System Architecture Concepts

3.3.1 Dual-Boot Firmware

The system maintains two complete firmware images (firmware1 and firmware2). During updates, the inactive partition is written while the system continues running from the active partition. After successful update, the bootloader switches to the updated partition on next boot.

If the new firmware fails to boot (detected via bootcount mechanism), the bootloader automatically reverts to the previous working firmware.

3.3.2 Configuration Persistence

Configuration follows a layered approach:

- Default configuration is provided by the system
- User configuration is stored and can be modified
- Configuration is persisted to flash storage via sysconf
- On boot, configuration is recovered from flash and applied

3.4 Access Methods

Web Interface - Browser-based web management over network (WiFi or Ethernet).

SSH - Secure shell access over network (WiFi or Ethernet) for command-line management.

Serial Console - Direct UART connection for low-level access, debugging, and recovery. Always available regardless of network configuration.

See: System Access for detailed access procedures.

3.5 Version Information

The system provides version and status information through the `swinfo` command:

- Device model and hostname
- Kernel version
- System uptime
- Memory usage
- CPU utilization

3.6 Configuration File Locations

User Configuration:

- `/etc/roboconf/config.yaml` - Active system configuration

Read-Only System Files:

- `/usr/lib/roboconf/config.yaml` - Factory default configuration
- `/etc/board.conf` - Board production data (serial number, MAC addresses, etc.)

Persistent Storage:

- Configuration is persisted to the `config` flash partition via sysconf

3.7 Supported Hardware

This documentation covers the following platforms:

- **MACA2** - Compact wireless module based on QCA4531 SoC
- **Lima** - Wireless module based on QCA4531 SoC with different peripherals and RF power amplifiers

See: MACA2 Hardware for detailed hardware specifications.

3.8 Next Steps

- System Startup - Boot process and indicators
- System Access - Connecting to the device
- Configuration - Configuration parameters reference
- Software Update - Firmware update procedures

4. MACA2 Board

4.1 Overview

MACA2 is a compact wireless module for embedded applications based on Qualcomm Atheros QCA4531 SoC. This document describes board-level hardware and peripherals.

4.2 Peripherals

- WiFi Radio (HT/11n)
- Ethernet Port #0 (10/100M)
- Ethernet Port #1 (10/100M)
- USB Ports (2x USB 2.0 Host)
- UART Serial Console
- Temperature Sensor
- Status LED
- Reset Button
- Slide Switch

4.3 Board Layout

4.3.1 Top View

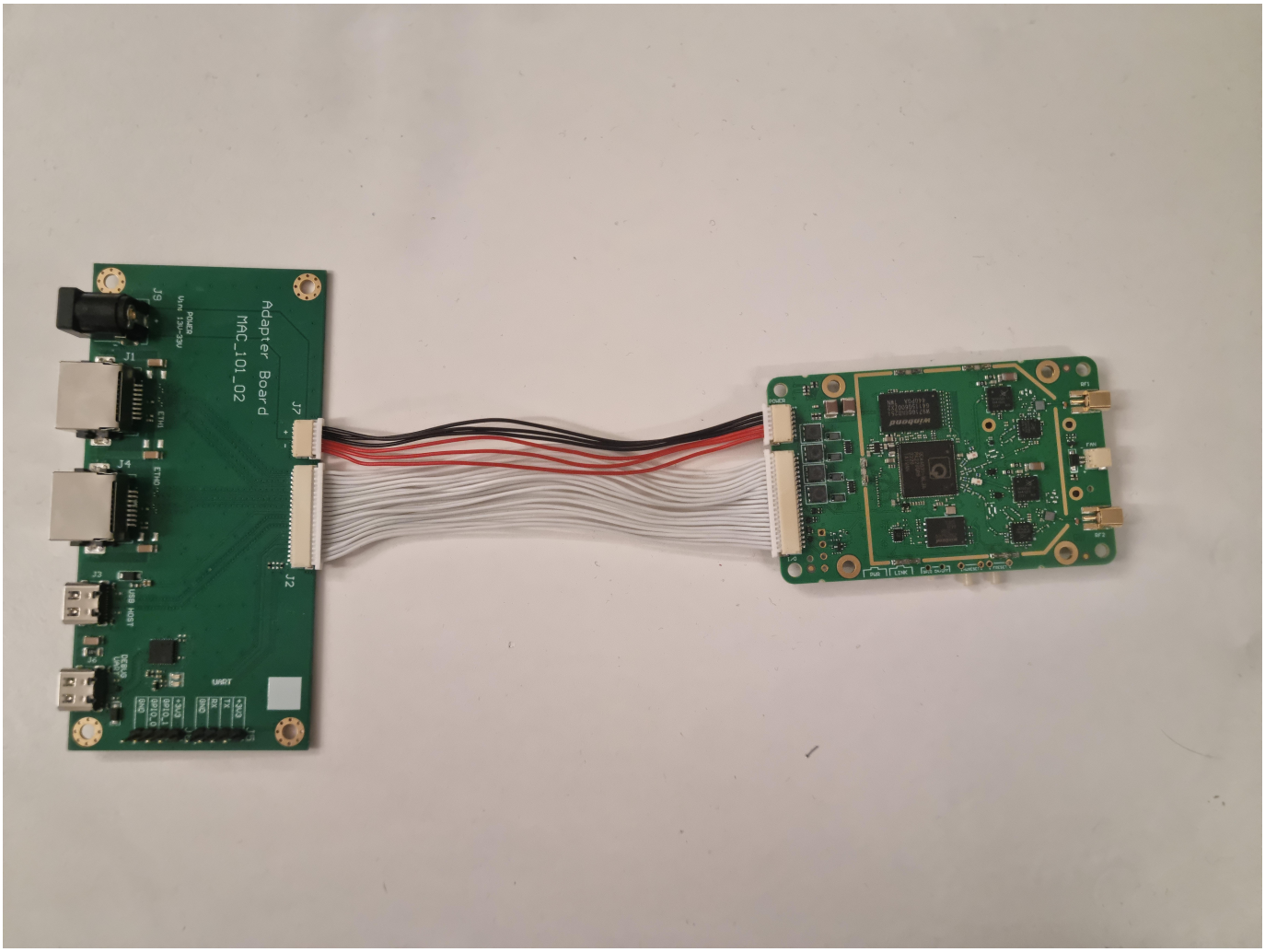

Figure 1: MACA2 board top view

4.3.2 Bottom View

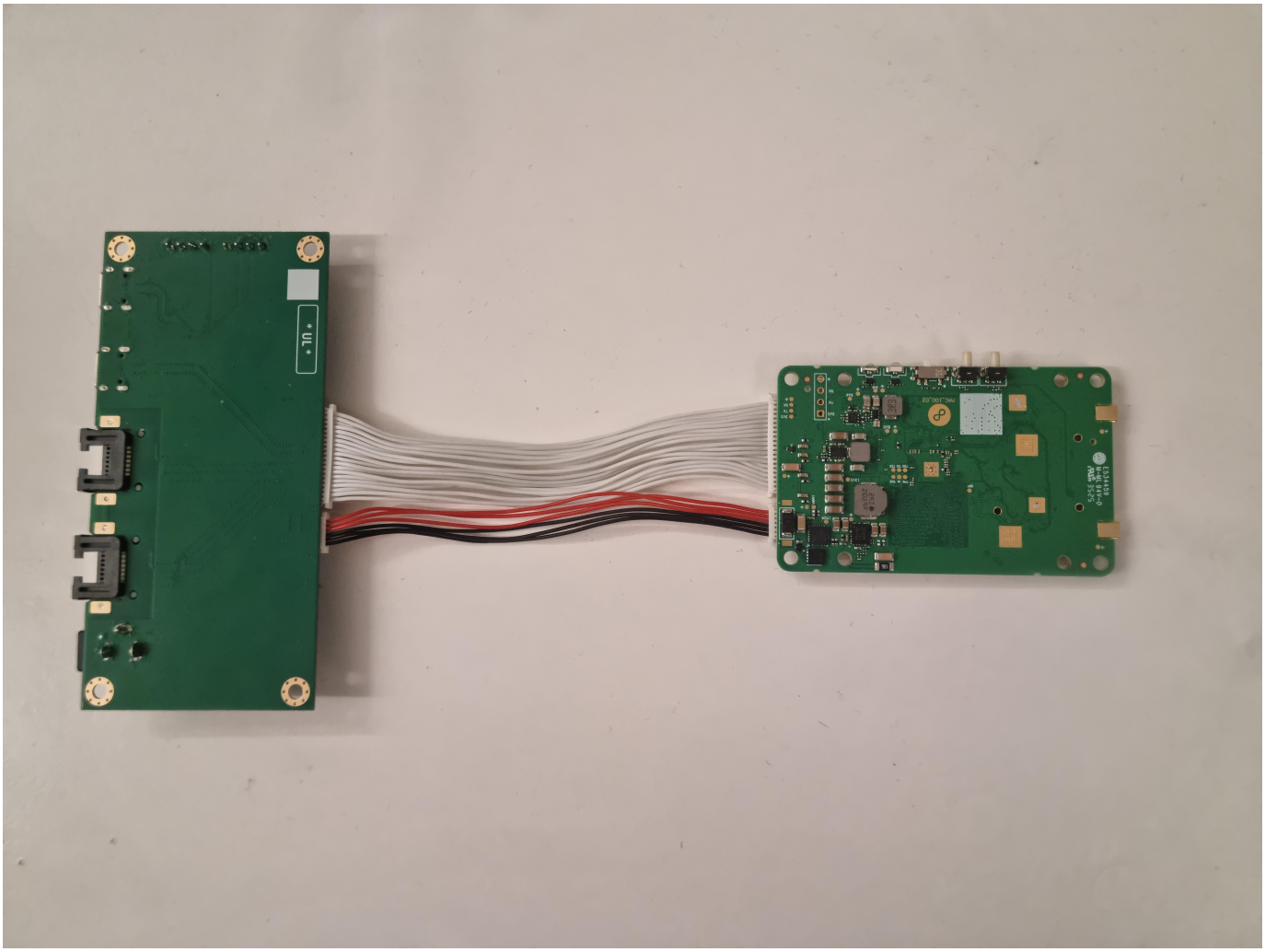

Figure 2: MACA2 board bottom view

4.4 Board Interfaces

4.4.1 WiFi Radio

Parameter	Description
Radio Chip	QCA4531 integrated
WiFi Standard	IEEE 802.11b/g/n
Channel Width	20/40 MHz
Operation Band	2.4 GHz
Operation Mode	2x2 (2T2R)
Antenna Connectors	2x MMCX
Max Conducted Tx Power (aggregate)	39 dBm
Frequency Range	Extended 2.3-2.7 GHz

Integrated WiFi radio supporting 2.4 GHz operation with extended frequency range for specialized applications.

TX Power Control:

The radio TX power control has a 10 dBm offset. When configuring `txpower` in software, the actual RF output is approximately:

Software Setting	RF Output (aggregate)	RF Output (per chain)
3 dBm	~13 dBm	~10 dBm
10 dBm	~20 dBm	~17 dBm
20 dBm	~30 dBm	~20 dBm
29 dBm	~39 dBm (max)	~36 dBm

4.4.2 Ethernet Port #0

Parameter	Description
Speed	10/100 Mbps
Duplex	Half/Full
Interface	eth0

4.4.3 Ethernet Port #1

Parameter	Description
Speed	10/100 Mbps
Duplex	Half/Full
Interface	eth1

4.4.4 USB Ports

Parameter	Description
Ports	2
Standard	USB 2.0
Speed	480 Mbps (HS)
Mode	Host

Host ports for connecting USB peripherals.

4.4.5 Serial Console

Parameter	Description
Interface	UART
Baud Rate	115200
Data Bits	8
Parity	None
Stop Bits	1

Serial console for system access and debugging.

4.4.6 Temperature Sensor

Parameter	Description
Chip	TMP112
Interface	I2C
Resolution	0.0625°C

Onboard temperature sensor for thermal monitoring.

4.4.7 Status LED

GPIO: TCA6408 pin 3

Application controlled LED for system status indication.

State	Behavior
Booting	Blinking
WiFi connecting	Blinking
WiFi connected	Solid ON

4.4.8 Push Button

GPIO: TCA6408 pin 5

Push button for device reboot and system factory reset.

Action	Duration	Function
Short press	<1 second	Device reboot
Long press	>5 seconds	Factory reset

4.4.9 Slide Switch

GPIO: 14

Slide switch for UART port mode selection. Also planned for mission and maintenance mode switching.

Position	Function
OFF	Developer console (serial login and debug)
ON	General-purpose UART (network-accessible serial port)

4.4.10 Fan

GPIO: TCA6408 pin 4

Cooling fan control output.

Initialise FAN control:

```
echo 534 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio534/direction
```

Enable fan:

```
echo 1 > /sys/class/gpio/gpio534/value
```

Disable fan:

```
echo 0 > /sys/class/gpio/gpio534/value
```

4.5 Power

Parameter	Value
Input Voltage	13-33V DC
Input Current	1.5-4A
Power Consumption	Up to 48W

4.6 System Specifications

4.6.1 SoC

Parameter	Value
Model	Qualcomm Atheros QCA4531
CPU	MIPS 24Kc @ 650 MHz
RAM	128 MB DDR2
Flash	32 MB SPI NOR

4.6.2 Flash Layout

Partition	Offset	Size	Purpose
u-boot	0x000000	256 KB	Bootloader
u-boot-env	0x040000	256 KB	Boot environment
art	0x080000	192 KB	Calibration data
eeprom	0x0b0000	64 KB	Production data
firmware1	0x0c0000	14 MB	Primary firmware
firmware2	0xec0000	14 MB	Secondary firmware
config	0x1fc0000	256 KB	Configuration storage

5. MACA2 Setup

This guide covers initial device setup and launch procedures for the MACA2 board.

For board hardware specifications and connector locations, refer to MACA2 Hardware.

5.1 Required Equipment

- Compatible DC power supply
- For console access: Serial console cable (3.3V TTL UART)
- For console access: Serial terminal software (minicom, screen, PuTTY, etc.)
- For network access: Ethernet cable or WiFi client device

5.2 Power Requirements

The MACA2 board requires a DC power supply ranging from **13V to 33V** capable of supplying up to 48W peak power.

Power On:

- Connect power supply to the power connector
- Device automatically starts up
- Status LED begins blinking during boot

Refer to MACA2 Hardware for power connector location on the PCB.

5.3 Serial Console Access

To access the device via serial console:

- Connect 3.3V TTL serial console cable to the UART header
- Configure serial terminal: 115200 baud, 8N1, no flow control
- Power on the device
- Login prompt appears after boot completes

Refer to MACA2 Hardware for serial console connector location on the PCB.

For detailed connection instructions, see System Access.

5.4 Network Access

5.4.1 Via Ethernet

- Connect Ethernet cable to eth0 or eth1 port
- Configure host computer IP in same subnet as device:
- eth0 (data zone): 192.168.1.x
- eth1 (mgmt zone): 192.168.2.x
- SSH to device: `ssh root@192.168.1.1` or `ssh root@192.168.2.1`

5.4.2 Via WiFi

- Device broadcasts default AP: `Robosoft-Default` (open network)
- Connect host computer to this WiFi network
- SSH to device: `ssh root@192.168.1.1`

5.5 First Boot

On first boot:

- Device initializes overlay filesystem (takes longer than normal boot)
- Default configuration is applied
- WiFi AP starts with default SSID
- SSH service is available

Default login: `root` with no password.

5.6 Firmware Recovery

If device fails to boot:

- Automatic failsafe activates after 3 consecutive boot failures
- Device reverts to alternate firmware partition
- If both partitions fail, use serial console to access U-Boot for recovery

For firmware update procedures, see Software Update.

5.7 Related Documentation

- MACA2 Hardware - Board hardware specifications
- System Access - Access methods details
- System Startup - Boot process and indicators
- Configuration - Device configuration

6. System Startup

6.1 Overview

This section describes the system boot process, expected startup behavior, and startup indicators. Understanding the boot sequence helps verify the system is operating correctly and aids in troubleshooting boot-related issues.

6.2 Boot Sequence

6.2.1 Power-On to Login

The system boot process follows these stages:

- **Bootloader Initialization** - U-Boot initializes hardware and selects boot partition
- **Kernel Loading** - Linux kernel loads from active firmware partition
- **Root Filesystem Mount** - Root filesystem mounted with writable overlay
- **Board Data Load** - Production data read from EEPROM to `/etc/board.conf`
- **Configuration Recovery** - User configuration recovered from flash storage
- **System Services Start** - Essential services initialize via procd
- **Network Services Start** - Network zones, bridges, and interfaces configured
- **WiFi Services Start** - WiFi interfaces and modes configured
- **System Ready** - Login prompt available, services operational

6.2.2 Expected Boot Time

Typical boot times from power-on to system ready:

- **Normal boot**: ~28 seconds
- **First boot**: ~37 seconds (additional filesystem initialization)

First boot after flashing new firmware takes longer due to overlay filesystem setup and initial configuration generation.

6.3 Boot Partition Selection

6.3.1 Dual-Boot Architecture

The system maintains two firmware partitions:

Partition	Flash Offset	Size
firmware1	0x0c0000	14 MB
firmware2	0xec0000	14 MB

The bootloader selects the active partition based on:

- `active` environment variable (set during firmware update)
- Boot failure counter (tracks consecutive failed boots)

6.3.2 Active Partition

The bootloader stores the active partition in U-Boot environment:

```
fw_printenv active
```

Output shows `1` for firmware1 or `2` for firmware2.

6.3.3 Automatic Failover

If the system fails to boot successfully:

- Bootloader increments boot failure counter
- After 3 consecutive failures, bootloader switches to alternate partition
- System boots from previous working firmware
- Boot counter resets after successful boot

This automatic failover ensures recovery from failed updates without user intervention.

6.3.4 Bootcount Management

The bootcount is stored in reserved memory and accessible via procfs:

```
cat /proc/bootcount
```

On successful boot, the system resets the bootcount to 0 via the `/etc/init.d/bootcount` service.

6.4 UART Mode Switch

The built-in UART port (ttyS0) has a slide switch that selects between two operating modes:

Switch Position	Mode	Description
OFF	Developer console	UART operates as a serial console for login, debugging, and recovery
ON	General-purpose UART	UART operates as a configurable serial port accessible over the network

In developer console mode (OFF), the UART provides a standard Linux login prompt and boot messages. This is the default mode for initial setup and troubleshooting.

In general-purpose mode (ON), the developer console is disabled and the UART port can be configured for serial device communication via TCP or UDP. See UART Configuration for network access setup.

The switch position is read at boot. Changing the switch position requires a reboot to take effect.

Note: feature requires Caraboot v2.10-RS U-boot version to work.

6.5 Startup Indicators

6.5.1 LED Status

The status LED indicates system state:

State	LED Behavior
Booting	Blinking
WiFi connecting	Blinking
WiFi connected	Solid ON
System error	Off or rapid blink

6.5.2 Serial Console Output

During boot, the serial console displays detailed boot messages:

- Bootloader messages and partition selection
- Kernel initialization and hardware detection
- Service startup messages
- Error messages if services fail to start

Serial console access requires connecting to the UART port. See System Access for connection details.

6.5.3 Network Availability

Network interfaces become available in stages:

- **Ethernet interfaces** - Available shortly after network service starts
- **WiFi Access Points** - Available after hostapd service starts
- **WiFi Station connections** - Available after scanning and authentication
- **Mesh links (NAW)** - Available after mesh peer discovery

6.6 Verifying System Startup

6.6.1 Check System Status

After boot, verify system is operational:

Check system uptime:

```
uptime
```

Check network interfaces:

```
ip addr show
```

Verify expected interfaces are present and have IP addresses assigned.

Check WiFi interfaces:

```
iw dev
```

Verify WiFi interfaces are present and in correct mode.

6.6.2 Check Service Status

Verify key services are running:

List running services:

```
ps | grep -E "(roboconf|hostapd|wpa_supplicant|wfb)"
```

Check specific service:

```
/etc/init.d/roboconf status
/etc/init.d/wfb-ng status
```

6.7 Boot Logs

6.7.1 Viewing Boot Messages

Boot logs are accessible through OpenWrt logging:

View kernel messages:

```
dmesg
```

View system log:

```
logread
```

Follow log in real-time:

```
logread -f
```

6.7.2 Boot Log Analysis

Boot logs help identify:

- Hardware detection issues
- Service startup failures
- Configuration errors
- Network initialization problems

Look for `error` or `failed` messages in boot logs to identify issues.

6.8 First Boot Behavior

6.8.1 Initial Configuration

On first boot with factory defaults:

- Default network zones are configured (data, mgmt)
- Default IP addresses are assigned (192.168.1.1, 192.168.2.1)
- WiFi radio configured in AP mode with default SSID
- SSH service enabled

Default configuration can be customized via `/etc/roboconf/config.yaml`.

6.8.2 Configuration Recovery

On subsequent boots:

- System checks for persisted configuration in flash
- If found, configuration is restored to `/etc/roboconf/config.yaml`
- Roboconf applies configuration to system services
- If no persisted config exists, factory defaults are used

6.9 Troubleshooting Boot Issues

6.9.1 System Not Booting

If system shows no activity:

- Check power supply connections
- Verify power LED status
- Connect to serial console (115200 baud) to view boot messages
- Check for bootloader error messages

If system fails repeatedly, automatic failover activates after 3 consecutive failures.

6.9.2 System Boots But Services Fail

If services fail to start:

- Check system logs:

```
logread | grep -i error
```

- Check specific service logs:

```
logread | grep roboconf
logread | grep hostapd
```

- Verify configuration syntax:

```
roboconf -nnn
```

- Try factory reset if configuration is corrupted:

```
roboconf reset
```

6.9.3 Network Not Available

If network interfaces are not available:

- Check interface status:

```
ip link show
```

- Check bridge configuration:

```
brctl show
```

- Check roboconf applied correctly:

```
uci show network
uci show wireless
```

6.10 Next Steps

- System Access - Connecting to the device
- Configuration - Configuration parameters
- Software Update - Firmware update procedures

7. System Access

This guide describes methods to access and interact with the device. Available access methods:

- **Serial console** - Direct serial connection via UART
- **SSH connection** - Secure shell over network
- **Web interface** - Browser-based management over network

7.1 Serial Console

Serial console provides direct access to the device regardless of network configuration. It is the primary method for initial setup, debugging, and recovery.

Note: The developer console is available when the UART port mode slide switch is in the OFF position. See UART Mode Switch for details.

7.1.1 Connection Setup

Connect a USB-to-UART adapter to the board's UART header. For header location, refer to MACA2 Hardware.

Serial Port Configuration:

Parameter	Value
Port	/dev/ttyUSB# (Linux) or COM# (Windows)
Baud Rate	115200
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None

Connection Wiring:

UART Pin	Signal	USB-UART Adapter
VCC	Power	VCC (optional)
TX	Transmit	RX
RX	Receive	TX
GND	Ground	GND

7.1.2 Terminal Emulators

Use any serial terminal emulator:

Linux:

```
picocom -b 115200 /dev/ttyUSB0

screen /dev/ttyUSB0 115200

minicom -D /dev/ttyUSB0 -b 115200
```

Windows:

Use PuTTY or similar: - Connection type: Serial - Serial line: COM# (check Device Manager) - Speed: 115200

7.1.3 Login

Upon successful connection, a login prompt is displayed:

```
maca2 login: root
```

Default username is `root` with no password.

7.2 SSH Connection

SSH provides secure remote access over the network. Connection requires IP connectivity via WiFi or wired Ethernet.

7.2.1 Network Requirements

The host computer and device must be on the same IP subnet. Default network configuration:

Zone	Default IP	Interface
data	192.168.1.1	eth0
mgmt	192.168.2.1	eth1

For custom network configuration, see Networking.

7.2.2 Shell Access

Connect using SSH client:

```
ssh root@192.168.1.1
```

Or for management interface:

```
ssh root@192.168.2.1
```

Default username is `root` with no password.

First connection:

On first SSH connection, you will be prompted to accept the host key:

```
The authenticity of host '192.168.1.1' can't be established.
ED25519 key fingerprint is SHA256:...
Are you sure you want to continue connecting (yes/no)?
```

Type `yes` to continue.

7.2.3 File Transfer

Transfer files using SCP (legacy mode required):

Upload files to device:

```
scp -O firmware.bin root@192.168.1.1:/tmp/
```

Download files from device:

```
scp -O root@192.168.1.1:/etc/roboconf/config.yaml ./
```

The `-O` option enables legacy SCP protocol mode.

7.3 Web Interface

The web interface provides browser-based system management and monitoring. Connection requires IP connectivity via WiFi or wired Ethernet.

The host computer and device must be on the same IP subnet. Ensure the host computer has an IP address on the same subnet as the device (e.g. `192.168.1.x` for eth0 or `192.168.2.x` for eth1).

For custom network configuration, see Networking.

7.3.1 Web Browser Access

Once device is connected navigate to the device IP address in a web browser.

Default access URLs:

Zone	URL	Interface
data	`http://192.168.1.1`	eth0
mgmt	`http://192.168.2.1`	eth1

7.4 Troubleshooting

7.4.1 Cannot Connect via Serial

- Ensure board connectors are properly seated
- Verify USB-UART adapter is recognized by host (`ls /dev/ttyUSB*` on Linux)
- Check TX/RX wiring is correct (swap if no output)
- Verify baud rate is set to 115200
- Try different USB port or cable

7.4.2 Cannot Connect via SSH

- Ensure board connectors are properly seated
- Verify network connectivity: `ping 192.168.1.1`

- Check IP address configuration on both host and device
- Verify SSH service is running on device
- Check firewall settings on host computer

7.4.3 Connection Drops or Freezes

- For serial: check cable connections and adapter quality
- For SSH: check network stability and interference
- Review system logs: `logread` on device

7.5 Next Steps

- Configuration - Configure device settings
- Networking - Network zone configuration
- Software Update - Update firmware

8. Software Update

8.1 Overview

This section describes the firmware update process and dual-boot architecture. The system uses a robust update mechanism with automatic failsafe to ensure reliable operation.

8.2 Dual-Boot Architecture

The system maintains two firmware partitions (firmware1, firmware2) enabling safe updates with automatic rollback capability. For detailed flash layout, see MACA2 Hardware.

8.2.1 How Dual-Boot Works

- Bootloader reads `active` variable from u-boot-env
- Boots from the active firmware partition (firmware1 or firmware2)
- On successful boot, system resets bootcount to 0
- If boot fails, bootloader increments bootcount
- After 3 consecutive failures, bootloader switches to alternate partition

This ensures the device always boots to a working firmware.

8.3 Firmware Update Process

8.3.1 Update Command

Use the `update` command to install new firmware:

```
update /tmp/firmware.bin
```

The `update` command validates the firmware image, writes it to the inactive partition, switches the active partition, and reboots the device.

The underlying `sysupgrade` utility can also be used directly for firmware updates.

8.3.2 Update Workflow

- **Validation** - Firmware image is validated for compatibility
- **Write** - Image is written to inactive partition

- **Switch** - Active partition pointer is updated
- **Reboot** - System reboots to new firmware

The update writes to the inactive partition, so the running system is not affected during the update process.

8.3.3 Update Methods

Via command line:

Transfer firmware to device using SCP, then run the update command:

```
scp -O firmware.bin root@192.168.1.1:/tmp/
```

```
update /tmp/firmware.bin
```

Via web interface:

Firmware can be uploaded and applied through the web management interface.

Via USB:

Place `firmware.bin` on a USB drive and insert it into the device. The firmware is automatically applied. See Device Provisioning for details.

8.4 Boot Partition Management

8.4.1 Check Active Partition

View current active partition:

```
fw_printenv active
```

Output: `active=1` (firmware1) or `active=2` (firmware2)

8.4.2 Check Bootcount

View current boot failure count:

```
cat /proc/bootcount
```

A value of 0 indicates successful boot. Values 1-2 indicate recent boot failures.

8.4.3 Manual Partition Switch

To manually switch to the alternate partition:

```
# If currently on firmware1, switch to firmware2
fw_setenv active 2

# Reboot to apply
reboot
```

Use with caution - only switch if you know the alternate partition has valid firmware.

8.5 Automatic Failsafe

8.5.1 Failsafe Behavior

The bootloader implements automatic failsafe:

- Each boot attempt increments bootcount
- Successful boot resets bootcount to 0
- If bootcount reaches 3, bootloader switches to alternate partition
- System boots from previous working firmware

8.5.2 Recovery Scenarios

Scenario: New firmware fails to boot

- Bootcount increments on each failed attempt
- After 3 failures, system reverts to previous firmware
- Previous firmware boots successfully
- User can investigate issue and retry update

Scenario: Both partitions have issues

- Connect via serial console
- Access U-Boot bootloader (press key during boot)
- Manually select partition or boot recovery image

8.6 Configuration Preservation

8.6.1 Default Behavior

Any system changes (installed packages, modified files, temporary data) made during run-time are discarded during the update.

System configuration is persisted to flash and automatically recovered after the update. The device boots with the same configuration as before the update.

To reset configuration to factory defaults, see Reset Configuration.

8.7 Troubleshooting

8.7.1 Update Fails to Start

If the update reports errors:

- Verify firmware file integrity
- Check available space in /tmp:

```
df -h /tmp
```

- Verify firmware is for correct device:

```
sysupgrade -T /tmp/firmware.bin
```

8.7.2 Device Fails to Boot After Update

If device doesn't boot after update:

- Wait for automatic failsafe (3 boot attempts)
- System should revert to previous firmware
- Connect via serial console to monitor boot process

8.7.3 Stuck in Boot Loop

If device continuously reboots:

- Connect serial console at 115200 baud
- Observe boot messages for errors
- If bootcount failsafe triggers, wait for alternate partition boot
- If both partitions fail, use U-Boot recovery

8.8 Next Steps

- System Startup - Boot process details
- Configuration CLI - Configuration management
- Monitoring - System diagnostics

9. Configuration Command-Line Tools

9.1 Overview

This section describes command-line utilities for configuration management: `roboconf` for applying configuration and `sysconf` for persistent storage.

9.2 Roboconf

The `roboconf` utility reads YAML configuration and applies it to the system.

9.2.1 Apply Configuration

Command:

```
roboconf apply
```

Process:

- Reads `/etc/roboconf/config.yaml`
- Merges with defaults from `/usr/lib/roboconf/config.yaml`
- Generates UCI configuration for network, wireless, and WFB services
- Persists configuration to flash storage via sysconf
- Reloads affected services

Options:

Option	Description
`-n`	Do not persist config to flash
`-nn`	Do not persist to flash and do not apply settings
`-nnn`	Dry-run debug mode without changing system state

Examples:

```
# Apply and persist configuration
roboconf apply

# Apply without persisting (temporary change)
```

```
roboconf -n

# Test configuration without applying
roboconf -nnn
```

9.2.2 Reset Configuration

Reset configuration to factory defaults.

Command:

```
roboconf reset
```

Process:

- Restores `/etc/roboconf/config.yaml` from factory defaults
- Erases persisted configuration from flash
- Applies default configuration to system

Options:

Option	Description
`-n`	Do not erase config from flash
`-nn`	Do not erase from flash and do not apply settings
`-nnn`	Dry-run debug mode without changing system state

9.2.3 Debug Mode

Enable verbose output for troubleshooting:

```
DEBUG=1 roboconf apply
```

9.3 Sysconf

The `sysconf` utility manages persistent configuration storage in flash memory.

9.3.1 Read Configuration

Recover configuration from flash storage:

```
sysconf -r
```

Reads persisted configuration from flash and writes to `/etc/roboconf/config.yaml`.

9.3.2 Write Configuration

Persist current configuration to flash:

```
sysconf -w
```

Writes `/etc/roboconf/config.yaml` to flash storage.

9.3.3 List Sectors

List all stored configuration sectors:

```
sysconf -l
```

9.3.4 Erase Configuration

Erase persisted configuration:

```
sysconf -e config
```

9.4 Usage Workflows

9.4.1 Modify Configuration

Standard workflow for configuration changes:

```
# 1. Edit configuration
vi /etc/roboconf/config.yaml

# 2. Test changes (dry-run)
roboconf -nnn

# 3. Apply configuration
roboconf apply
```

9.4.2 Temporary Configuration

Apply changes without persisting to flash:

```
# Edit configuration
vi /etc/roboconf/config.yaml

# Apply without persisting
roboconf -n
```

Configuration reverts to persisted version after reboot.

9.4.3 Recover Configuration

If configuration file is missing or corrupted:

```
# Recover from flash
sysconf -r

# Apply recovered configuration
roboconf -nn
```

9.4.4 Factory Reset

Complete reset to factory defaults:

```
roboconf reset
```

9.5 Troubleshooting

9.5.1 Configuration Not Applied

If changes don't take effect:

- Test configuration syntax:

```
roboconf -nnn
```

- Check for errors in output
- Verify services are running:

```
ps | grep -E "(hostapd|wpa_supplicant|wfb)"
```

- Check system logs:

```
logread | grep roboconf
```

9.5.2 Configuration Lost After Reboot

If configuration reverts after reboot:

- Verify configuration was persisted:

```
sysconf -l
```

- Check for "config" sector in output
- Re-apply with persistence:

```
roboconf apply
```

9.6 Next Steps

- Configuration - Configuration parameters reference
- Software Update - Firmware update procedures
- Networking - Network configuration details

10. Configuration Parameters

10.1 Overview

This section describes the Roboconf configuration system and available parameters. Roboconf uses YAML-based configuration with automatic parameter backfilling from defaults.

10.2 Configuration Files

File	Location	Purpose	Editable
Board Information	`/etc/board.conf`	Production data and device identity	No
Configuration File	`/etc/roboconf/config.yaml`	Active system configuration	Yes
Default Configuration	`/usr/lib/roboconf/config.yaml`	Factory defaults	No

10.3 Board Information

The `/etc/board.conf` file contains production data programmed during manufacturing and read from EEPROM at boot.

Board information fields:

Field	Description
`PRODUCT_ID`	Product identifier
`PRODUCT_NAME`	Product name
`SERIAL_NO`	Device serial number
`PCB_NAME`	PCB design name
`PCB_REVISION`	Hardware revision
`PCB_SN`	PCB serial number
`PCB_PROD_DATE`	Production date
`PCB_PROD_LOCATION`	Production facility
`MAC_ADDR_eth0`	Ethernet port 0 MAC address
`MAC_ADDR_eth1`	Ethernet port 1 MAC address
`MAC_ADDR_wlan0`	WiFi interface MAC address

Example:

```
PRODUCT_ID=MACA2
PRODUCT_NAME=8devices Maca2
SERIAL_NO=M2A00001234
PCB_NAME=MACA2-R1
PCB_REVISION=1.0
PCB_SN=PCB00001234
PCB_PROD_DATE=2024-01
PCB_PROD_LOCATION=LT
MAC_ADDR_eth0=00:11:22:33:44:55
MAC_ADDR_eth1=00:11:22:33:44:56
MAC_ADDR_wlan0=00:11:22:33:44:57
```

Board information is read-only and used for device identification and provisioning.

10.4 Configuration Structure

10.4.1 Version Field

Every configuration file must include version field:

```
version: "0.1.0"
```

10.4.2 Network Zones

Network zones define logical network segments. Multiple interfaces can belong to the same zone. Default configuration defines default `data` and `mgmt` network zones however additional network zones may be created as needed.

Zone parameters:

Parameter	Type	Values	Description
`mode`	enum	`static`, `dhcp-client`	IP address assignment method
`address`	IPv4	Valid IPv4 address	IP address for this zone

Addressing modes:

- **`static`** - Fixed IP address assignment
- **`dhcp-client`** - Obtain IP from DHCP server, fallback to configured address

Example:

```
network:
  zones:
    data:
      mode: "dhcp-client"
      address: 192.168.1.1
    mgmt:
      mode: "dhcp-client"
      address: 192.168.2.1
```

10.4.3 Ethernet Interfaces

Ethernet interface configuration assigns interfaces to network zones.

Interface parameters:

Parameter	Type	Description
`zone`	string	Network zone name (must exist in `network.zones`)

Example:

```
ethernet:
  eth0:
    zone: "data"
  eth1:
    zone: "mgmt"
```

Zone inheritance: Interface inherits all settings from assigned zone (IP address, DHCP mode).

Automatic bridging: If multiple interfaces are assigned to the same zone, a bridge is created automatically (`br-<zone_name>`).

10.4.4 WiFi Configuration

WiFi configuration is organized under the `wifi` section with radio-specific settings.

Radio Parameters

Parameter	Type	Range/Values	Description
`mode`	enum	`BSS`, `NAW`, `WFB`, `SCAN`	Operation mode
`country`	string	2 characters	Country code for regulatory compliance
`channel`	int	0-14	WiFi channel (0 = auto)
`frequency`	int	MHz value	Target frequency (0 = use channel default)
`width`	float	1-20, 40	Channel bandwidth in MHz
`txpower`	int	0-30	Transmission power in dBm
`gi`	int/string	400, 800, "auto"	Guard interval in nanoseconds (400=short, 800=long, "auto" = automatic selection)
`mcs`	int/string	0-15, "auto"	Modulation and Coding Scheme ("auto" = automatic selection)
`ack_timeout`	int	0-744	ACK timeout in microseconds (0 = auto)
`cca_threshold`	int	-100 to 0	CCA threshold in dBm

When `gi` or `mcs` is set to `"auto"`, the system uses automatic rate and guard interval selection. WFB mode does not support automatic rates — `"auto"`, values fall back to fixed defaults (MCS 3, long guard interval) for WFB streams.

Channel and Frequency Configuration

The `channel` parameter selects a standard radio-supported channel as the base frequency. The `frequency` parameter fine-tunes the actual operating frequency.

channel	frequency	Result
0	0	Auto channel selection
1-14	0	Fixed standard channel
0	MHz value	Fixed frequency, base channel selected automatically
1-14	MHz value	Fixed frequency, base channel selected manually

When `frequency` is non-zero, the system calculates the frequency shift from the base channel to achieve the target frequency. The difference between `channel=0` and `channel!=0` when using custom frequency is whether the base channel is selected automatically or manually by the user.

The supported extended frequency range depends on the board hardware. Current boards support 2202–2852 MHz. Refer to board specifications for exact limits.

Examples:

```
# Auto channel
channel: 0
frequency: 0

# Fixed channel 6
channel: 6
frequency: 0

# Extended frequency 2350 MHz (auto base channel)
channel: 0
frequency: 2350

# Extended frequency 2350 MHz (manual base channel 6)
channel: 6
frequency: 2350
```

Channel Widths

Standard channel widths are 20 MHz and 40 MHz, providing full data throughput as per 802.11n specification.

Narrow channel widths (1, 1.5, 2, 2.5, up to 20 MHz) reduce data throughput proportionally but extend range and improve signal reliability.

WiFi Modes

BSS Mode (Basic Service Set): Standard WiFi mode supporting Access Point and Station operation.

NAW Mode (Non-Associated WiFi): Mesh based networking mode for 2-node deployments.

WFB Mode (WiFi Broadcast): Long-range packet radio mode using raw WiFi frames.

SCAN Mode: Dedicated mode for active spectral scanning.

Monitor Interface

Monitor mode can be enabled independently of the active radio mode. When enabled, a passive monitor interface is created alongside the primary interface.

Parameter	Type	Range/Values	Description
`mon.enabled`	boolean	True, False	Enable passive monitor interface

```
wifi:
  radio0:
    mode: "BSS"
    mon:
      enabled: True
```

Access Point Configuration

AP parameters (used when `mode: "BSS"`):

Parameter	Type	Range/Values	Description
`enabled`	boolean	True, False	Whether AP is active
`ssid`	string	1-32 characters	Network name
`passphrase`	string	empty or 8-63 chars	WiFi password (empty = open)
`zone`	string	zone name	Network zone assignment

Station Configuration

Station parameters (used when `mode: "BSS"`):

Parameter	Type	Range/Values	Description
`enabled`	boolean	True, False	Whether station is active
`ssid`	string	1-32 characters	Network name to connect to
`passphrase`	string	empty or 8-63 chars	WiFi password
`zone`	string	zone name	Network zone assignment

NAW Configuration

NAW parameters (used when `mode: "NAW"`):

Parameter	Type	Range/Values	Description
`enabled`	boolean	True, False	Whether NAW is active
`link_id`	string	1-32 characters	Mesh link identifier
`passphrase`	string	empty or 8-63 chars	Link password
`zone`	string	zone name	Network zone assignment

Note: NAW mode requires a fixed channel. If auto channel (`channel: 0, frequency: 0`) is configured, the system falls back to channel 6.

10.4.5 WFB Configuration

WFB (WiFi Broadcast) configuration for long-range packet radio links.

WFB streams uses radio configuration to set up Tx and Rx parameters.. Optionally they can be overridden per-stream.

Note: WFB mode requires a fixed channel. If auto channel (`channel: 0, frequency: 0`) is configured, the system falls back to channel 6.

TX Stream Parameters

Parameter	Type	Range/Values	Description
`enabled`	boolean	True, False	Enable TX stream
`mode`	string	"tx"	Stream direction
`listen_port`	int	1-65535	UDP port to receive data
`fec_k`	int	1-128	FEC data packets
`fec_n`	int	1-128	FEC total packets (k + redundancy)
`stbc`	boolean	True, False	Space-Time Block Coding
`ldpc`	boolean	True, False	Low-Density Parity-Check
`qdisc`	boolean	True, False	Queue discipline control
`stream_id`	int	0-255	Stream identifier

RX Stream Parameters

Parameter	Type	Range/Values	Description
`enabled`	boolean	True, False	Enable RX stream
`mode`	string	"rx"	Stream direction
`forward_addr`	IPv4	Valid IPv4	Forward destination address
`forward_port`	int	1-65535	Forward destination port
`stream_id`	int	0-255	Stream identifier to receive

10.4.6 UART Configuration

UART configuration enables serial port access over the network using TCP or UDP streaming.

USB serial adapters (CH341, FTDI, PL2303) are automatically detected when connected. The system creates a default configuration entry for each detected adapter, which can be customized in `config.yaml`.

Operating Modes

Access Mode (TCP/UDP):

In access mode, the serial port listens on a network port and waits for incoming connections. When a client connects, data is bidirectionally forwarded between the network socket and the serial port. This mode is useful for on-demand serial console access or interactive communication with serial devices.

- **TCP access**: Reliable, connection-oriented. Client connects to the configured port, and the connection persists until closed. Suitable for interactive sessions and reliable data transfer.
- **UDP access**: Connectionless. Clients send datagrams to the configured port. Suitable for simple, low-latency data exchange where delivery guarantee is not required.

UDP Streaming Mode:

In UDP streaming mode, the serial port actively pushes received data to a configured remote destination without waiting for incoming connections. This mode is useful for telemetry and data logging scenarios where the device continuously sends serial data to a central server.

When `udp_stream` is enabled, data received on the serial port is immediately forwarded as UDP packets to the address and port specified by `udp_stream_addr` and `udp_stream_port`.

UART Device Parameters

Parameter	Type	Range/Values	Description
`enabled`	boolean	True, False	Enable UART device
`serial_baud`	int	Standard baud rates	Serial port baud rate (default: 115200)
`serial_frame`	string	"N81", "E71", etc.	Frame format: parity, data bits, stop bits
`access_port`	int	1-65535	Network port for serial access
`access_protocol`	enum	`tcp` , `udp`	Network protocol for access mode
`udp_stream`	boolean	True, False	Enable UDP streaming mode
`udp_stream_addr`	IPv4	Valid IPv4	UDP stream destination address
`udp_stream_port`	int	1-65535	UDP stream destination port

Frame format string:

- First character: Parity (N=none, E=even, O=odd)
- Second character: Data bits (7 or 8)
- Third character: Stop bits (1 or 2)

Example - Built-in UART (ttyS0) with TCP access:

The built-in UART port (ttyS0) can be switched from developer console to general-purpose UART mode using the slide switch. When in general-purpose mode, the UART can be accessed over the network.

```
uart:
  ttyS0:
    enabled: True
    serial_baud: 115200
    serial_frame: "N81"
    access_port: 3000
    access_protocol: "tcp"
```

Connect using: `nc <device-ip> 3000`

Example - USB serial adapter with TCP access:

```
uart:
  ttyUSB0:
    enabled: True
    serial_baud: 115200
    serial_frame: "N81"
    access_port: 3001
    access_protocol: "tcp"
```

Example - UDP streaming mode:

```
uart:
  ttyUSB0:
    enabled: True
    serial_baud: 9600
    serial_frame: "N81"
    access_port: 3000
    access_protocol: "udp"
    udp_stream: True
    udp_stream_addr: 192.168.1.100
    udp_stream_port: 5000
```

Serial data is automatically forwarded to 192.168.1.100:5000 as UDP packets.

Supported USB serial adapters:

- CH341
- FTDI
- PL2303

10.5 Configuration Syntax

10.5.1 YAML Format

Configuration uses YAML syntax:

- Use spaces for indentation (not tabs)
- Consistent indentation (2 spaces recommended)
- Quote strings containing special characters
- Boolean values: `True` or `False`
- Integers: unquoted numbers
- Strings: quoted or unquoted

10.5.2 Field Requirements

Case sensitivity: Field names are case-sensitive. Use exact names as documented.

Mode-specific sections:

- Configure `ap`/`sta` when radio `mode: "BSS"`
- Configure `naw` when radio `mode: "NAW"`
- Configure `wfb` section when radio `mode: "WFB"`

10.6 Next Steps

- Configuration CLI - Command-line configuration tools
- WiFi Modes - Detailed WiFi mode descriptions
- Networking - Network configuration details

11. Configuration Examples

This document provides practical configuration examples for common use cases.

11.1 BSS Mode Link (AP/STA)

Establish WiFi link between two devices using Access Point and Station roles.

Device A (Access Point):

```
wifi:
  radio0:
    mode: "BSS"
    channel: 0
    ap:
      enabled: True
      ssid: "LinkNetwork"
      passphrase: "linkpassword"
      zone: "data"
    sta:
      enabled: False
```

Device B (Station):

```
wifi:
  radio0:
    mode: "BSS"
    channel: 0
    ap:
      enabled: False
    sta:
      enabled: True
      ssid: "LinkNetwork"
      passphrase: "linkpassword"
      zone: "data"
```

Requirements: - `ssid` and `passphrase` must match on both devices - When STA uses fixed channel, AP should also use the same fixed channel

Data flow: Traffic received on the data zone network is bridged between AP and STA. Data arriving on Device A ethernet is forwarded over WiFi to Device B, and vice versa.

See Access Point Configuration and Station Configuration for parameter reference, and BSS Mode for detailed description.

11.2 NAW Mode Link

Establish wireless link between two devices using NAW mode.

Device A:

```
wifi:
  radio0:
    mode: "NAW"
    channel: 6
    naw:
      enabled: True
      link_id: "DataLink"
      passphrase: "linkpassword"
      zone: "data"
```

Device B:

```
wifi:
  radio0:
    mode: "NAW"
    channel: 6
    naw:
      enabled: True
      link_id: "DataLink"
      passphrase: "linkpassword"
      zone: "data"
```

Requirements: - `channel`, `link_id`, and `passphrase` must match on both devices - Auto channel not supported, use fixed channel

Data flow: Traffic received on the data zone network is bridged between devices. Data arriving on Device A ethernet is forwarded over WiFi to Device B, and vice versa. Both devices operate as peers with equal roles.

See NAW Configuration for parameter reference and NAW Mode for detailed description.

11.3 WFB Mode Link

Establish long-range broadcast link between two devices.

Device A (Transmitter):

```
wifi:
  radio0:
```

```
    mode: "WFB"
    channel: 6

wfb:
  streams:
    video:
      enabled: True
      mode: "tx"
      listen_port: 5600
      fec_k: 8
      fec_n: 12
      mcs: 1
      stream_id: 1
```

Device B (Receiver):

```
wifi:
  radio0:
    mode: "WFB"
    channel: 6

wfb:
  streams:
    video:
      enabled: True
      mode: "rx"
      forward_addr: 192.168.1.100
      forward_port: 5600
      stream_id: 1
```

Requirements: - `channel` and `stream_id` must match on both devices - Auto channel not supported, use fixed channel - Encryption keys must match (see WFB Keys)

Data flow: Transmitter listens for UDP packets on `listen_port`. Received packets are encoded with FEC and broadcast over WiFi. Receiver captures WiFi frames, decodes FEC, and forwards recovered data via UDP to `forward_addr forward_port`.

Use case: Video streaming from drone/robot to ground station, telemetry links, or any application requiring long-range one-way data transmission.

See WFB Configuration for parameter reference and WFB Mode for detailed description.

11.4 Multiple WFB Streams

Configure multiple independent streams on same link.

Transmitter:

```
wfb:
  streams:
    video:
      enabled: True
      mode: "tx"
      listen_port: 5600
      stream_id: 1
    telemetry:
      enabled: True
      mode: "tx"
      listen_port: 14550
      stream_id: 2
```

Receiver:

```
wfb:
  streams:
    video:
      enabled: True
      mode: "rx"
      forward_addr: 192.168.1.100
      forward_port: 5600
      stream_id: 1
    telemetry:
      enabled: True
      mode: "rx"
      forward_addr: 192.168.1.100
      forward_port: 14550
      stream_id: 2
```

Requirements: - Each stream needs unique name and `stream_id` - Ports must be unique on each device - Stream names are arbitrary identifiers

11.5 Frequency Fine-Tuning

To operate on extended frequency range, specify the `frequency` parameter:

```
wifi:
  radio0:
    mode: "NAW"
    channel: 6
    frequency: 2350
```

The `frequency` parameter defines the final transmission frequency in MHz. The `channel` parameter defines the base frequency, providing more fine-grained frequency domain control, but does not change the final outcome. Alternatively, `channel` can be set to 0 and only the `frequency` parameter used. This applies to all WiFi modes.

See Channel and Frequency Configuration for details.

11.6 Narrow Channel Operation

To extend range using narrow channels, specify the `width` parameter:

```
wifi:
  radio0:
    mode: "WFB"
    channel: 6
    width: 10
```

Supported narrow channel widths: 1, 1.5, 2, 2.5, up to 20 MHz. Standard channel widths: 20 MHz and 40 MHz. This applies to all WiFi modes.

See Channel Widths for details.

11.7 Separate Management Network

Configure separate network zones for data and management traffic.

```
network:
  zones:
    data:
      mode: "static"
      address: 192.168.1.1
    mgmt:
      mode: "static"
      address: 192.168.2.1

ethernet:
  eth0:
    zone: "data"
  eth1:
    zone: "mgmt"
```

Key points: - Each zone has its own IP subnet - Management access via eth1 on 192.168.2.x network - Data traffic via eth0 on 192.168.1.x network

WiFi zone assignment:

WiFi interfaces in BSS or NAW mode can be assigned to network zones, providing L2 bridging between wired and wireless interfaces in the same zone:

```
wifi:
  radio0:
    mode: "BSS"
    ap:
      enabled: True
      zone: "data"
```

See Network Zones for details.

11.8 Next Steps

- Configuration - Full parameter reference
- Configuration CLI - Apply and manage configuration
- WiFi Modes - Detailed WiFi mode descriptions

12. Device Provisioning

12.1 Overview

The device supports automated provisioning methods for firmware updates, configuration import, and diagnostic data collection without requiring network or shell access.

12.2 USB Provisioning

When a USB drive is inserted, the system automatically detects and processes provisioning files present on the drive.

12.2.1 Supported Operations

File on USB	Operation
`firmware.bin`	Firmware update
`config.yaml`	Configuration import
`tx.key`, `rx.key`	WFB encryption key import

Additionally, the device exports a `troubleshoot.zip` diagnostic bundle to the USB drive.

12.2.2 Firmware Update

When `firmware.bin` is present on the USB drive, the device performs a firmware update automatically.

The device serial number is recorded on the USB drive after a successful upgrade to avoid duplicate upgrades when the same drive is used across multiple devices.

12.2.3 Configuration Import

When `config.yaml` is present on the USB drive, the configuration is imported and applied automatically. The following files are recognized:

- `config.yaml` - Copied to `/etc/roboconf/config.yaml`
- `tx.key` - Copied to `/etc/wfb-ng/tx.key`
- `rx.key` - Copied to `/etc/wfb-ng/rx.key`

Configuration is applied via `roboconf apply` after import.

12.2.4 Troubleshoot Export

A diagnostic data bundle is automatically saved to the USB drive as `troubleshoot.zip`. This file can be used for offline troubleshooting and support.

12.2.5 Usage

- Prepare a USB drive with the desired provisioning files
- Insert the USB drive into the device
- Operations are performed automatically
- Check log output on the USB drive for results

12.3 Next Steps

- Configuration Parameters - Configuration file reference
- Software Update - Firmware update details
- Monitoring - Diagnostic data collection

13. Networking

13.1 Overview

This section describes network configuration concepts including zones, interfaces, and IP addressing.

13.2 Network Zones

Network zones are logical network segments. Each zone has its own IP configuration and can contain multiple interfaces.

13.2.1 Zone Concept

- Zones group interfaces that share the same network
- Interfaces in a zone inherit zone IP settings
- Multiple interfaces in one zone are automatically bridged

13.2.2 Default Zones

Zone	Default IP	Purpose
data	192.168.1.1	Primary data network
mgmt	192.168.2.1	Management network

13.2.3 Zone Configuration

```
network:
  zones:
    data:
      mode: "static"
      address: 192.168.1.1
    mgmt:
      mode: "dhcp-client"
      address: 192.168.2.1
```

13.2.4 Addressing Modes

static - Fixed IP address: - Device uses configured address - Address does not change

dhcp-client - Dynamic IP from DHCP server: - Device requests IP from DHCP server - Falls back to configured address if no DHCP response

13.3 Interface Assignment

13.3.1 Ethernet Interfaces

Assign Ethernet ports to zones:

```
ethernet:
  eth0:
    zone: "data"
  eth1:
    zone: "mgmt"
```

13.3.2 WiFi Interfaces

WiFi interfaces are assigned to zones in WiFi configuration:

```
wifi:
  radio0:
    mode: "BSS"
    ap:
      enabled: True
      zone: "data"
    sta:
      enabled: False
      zone: "mgmt"
```

13.4 Automatic Bridging

When multiple interfaces belong to the same zone, a bridge is created automatically.

Example: If eth0 and WiFi AP both use "data" zone: - Bridge `br-data` is created - eth0 and wlan0 are added to bridge - Bridge receives zone IP address - Traffic flows between interfaces at Layer 2

13.5 Network Verification

13.5.1 Check IP Addresses

```
ip addr show
```

13.5.2 Check Bridge Status

```
brctl show
```

13.5.3 Check Routing

```
ip route show
```

13.5.4 Test Connectivity

```
ping 192.168.1.100
```

13.6 Common Configurations

13.6.1 Single Zone (All Interfaces Bridged)

All interfaces on same network:

```
network:
  zones:
    data:
      mode: "static"
      address: 192.168.1.1

ethernet:
  eth0:
    zone: "data"
  eth1:
    zone: "data"

wifi:
  radio0:
    mode: "BSS"
    ap:
      enabled: True
      zone: "data"
```

13.6.2 Separate Data and Management

Data on WiFi, management on Ethernet:

```
network:
  zones:
    data:
      mode: "static"
      address: 192.168.1.1
    mgmt:
      mode: "static"
      address: 192.168.2.1
```

```
ethernet:
  eth0:
    zone: "mgmt"

wifi:
  radio0:
    mode: "BSS"
    ap:
      enabled: True
      zone: "data"
```

13.6.3 DHCP Client Uplink

Connect to existing network via Ethernet:

```
network:
  zones:
    data:
      mode: "static"
      address: 192.168.1.1
    uplink:
      mode: "dhcp-client"
      address: 0.0.0.0

ethernet:
  eth0:
    zone: "data"
  eth1:
    zone: "uplink"
```

13.7 Troubleshooting

13.7.1 No IP Address

If interface has no IP:

- Check zone assignment in configuration
- Verify roboconf applied:

```
uci show network
```

- Check interface status:

```
ip link show
```

13.7.2 Cannot Reach Device

If device is unreachable:

- Verify IP configuration:

```
ip addr show
```

- Check you're on same subnet
- Verify interface is up:

```
ip link show eth0
```

13.7.3 Bridge Issues

If bridged interfaces don't communicate:

- Check bridge exists:

```
brctl show
```

- Verify interfaces are in bridge
- Check STP status if applicable

13.8 Next Steps

- Configuration - Full parameter reference
- WiFi Modes - WiFi configuration
- Monitoring - Network diagnostics

14. Monitoring

14.1 Overview

This section describes system monitoring and diagnostic tools for health checks, statistics retrieval, and troubleshooting.

14.2 System Information

14.2.1 swinfo

Display system information:

```
swinfo
```

Shows: - Device model and hostname - Kernel version - System uptime - Memory usage - CPU utilization

14.2.2 Uptime

```
uptime
```

Shows system uptime and load averages.

14.2.3 Memory Usage

```
free -m
```

Shows RAM usage in megabytes.

14.2.4 Storage Usage

```
df -h
```

Shows filesystem usage.

14.3 Temperature Monitoring

The MACA2 includes an onboard temperature sensor (TMP112).

14.3.1 Read Temperature

```
cat /sys/class/hwmon/hwmon*/temp1_input
```

Value is in millidegrees Celsius (divide by 1000 for degrees).

Example:

```
# Read and convert to degrees
awk '{print $1/1000"°C"}' /sys/class/hwmon/hwmon*/temp1_input
```

14.4 Network Statistics

14.4.1 Interface Statistics

```
ip -s link show eth0
```

Shows TX/RX packets, bytes, errors, and drops.

14.4.2 WiFi Statistics

```
iw dev wlan0 station dump
```

Shows per-station statistics including signal strength and bitrates.

See WiFi Controls for detailed WiFi monitoring.

14.5 WFB Statistics

WFB link statistics are collected to `/var/stats/` for monitoring link quality and performance.

14.5.1 Live Monitoring

Watch TX statistics in real-time:

```
watch -n 1 cat /var/stats/wfb-stats-tx.json
```

Watch RX statistics in real-time:

```
watch -n 1 cat /var/stats/wfb-stats-rx.json
```

Follow text log output:

```
tail -f /var/stats/wfb-stats-rx.txt
```

14.5.2 Quick Link Health Check

```
# Check current signal strength and packet loss
cat /var/stats/wfb-stats-rx.json
```

Key indicators to monitor:

- **rssi/snr** - Signal quality (higher is better)
- **lost** vs **good** - Packet loss ratio
- **recovered** - FEC corrections (high values indicate marginal link)

See WiFi Controls for statistics field descriptions.

14.6 GPIO Button and Switch State

```
cat /proc/gpio-buttons
```

Shows all button and switch states in `name:value` format (1 = active, 0 = inactive).

14.7 Diagnostic Data Collection

The `trouble` utility collects comprehensive diagnostic data including system configuration, logs, network state, and interface statistics.

```
trouble
```

Generates a diagnostic zip file in `/tmp/`.

Custom output path:

```
trouble /tmp/custom_diag.zip
```

Diagnostic data can also be downloaded through the web interface.

14.8 System Logs

14.8.1 View System Log

```
logread
```

Shows system messages from all services.

14.8.2 Follow Log in Real-Time

```
logread -f
```

Streams new log entries as they occur.

14.8.3 Filter Logs

```
logread | grep roboconf
logread | grep hostapd
logread | grep error
```

14.8.4 Kernel Messages

```
dmesg
```

Shows kernel ring buffer messages.

14.9 Process Monitoring

14.9.1 Running Processes

```
ps
```

Lists all running processes.

14.9.2 Process Resource Usage

```
top
```

Interactive process monitor. Press 'q' to exit.

14.9.3 Check Specific Service

```
ps | grep hostapd
ps | grep wfb
```

14.10 Boot Information

14.10.1 Boot Count

```
cat /proc/bootcount
```

Shows number of boot attempts since last successful boot. Value of 0 indicates healthy boot.

14.10.2 Active Firmware Partition

```
fw_printenv active
```

Shows which firmware partition (1 or 2) is currently active.

14.11 Health Checks

14.11.1 Quick Health Check

Run these commands to verify system health:

```
# System info
swinfo

# Memory
free -m

# Storage
df -h

# Temperature
cat /sys/class/hwmon/hwmon*/temp1_input

# Network interfaces
ip addr show

# WiFi status
iw dev
```

14.11.2 Service Status

Verify key services are running:

```
ps | grep -E "(hostapd|wpa_supplicant|wfb)"
```

14.12 Troubleshooting

14.12.1 High Memory Usage

If memory is low:

- Check process memory:

  ```
  top
  ```

- Look for memory-heavy processes
- Consider restarting services or device

14.12.2 High Temperature

If temperature is elevated:

- Check current temperature
- Ensure adequate ventilation
- Reduce TX power if applicable
- Check for runaway processes

14.12.3 Service Not Running

If expected service is not running:

- Check if process exists:

```
ps | grep <service>
```

- Check service logs:

```
logread | grep <service>
```

- Try restarting:

```
/etc/init.d/<service> restart
```

14.12.4 Network Issues

For network problems:

- Check interface status:

```
ip addr show
```

- Check connectivity:

```
ping <target>
```

- Check routing:

```
ip route show
```

- Check logs:

```
logread | grep -i network
```

14.13 Next Steps

- WiFi Controls - WiFi-specific monitoring
- System Startup - Boot troubleshooting
- Configuration CLI - Configuration troubleshooting

15. WiFi Operating Modes

15.1 Overview

The system supports the following WiFi operating modes, each designed for different use cases:

- **BSS** - Standard WiFi for client connectivity
- **NAW** - Mesh networking for multi-node deployments
- **WFB** - Long-range broadcast for video/data links
- **SCAN** - Dedicated mode for active spectral scanning

Only one mode can be active at a time on the radio.

Additionally, a passive **monitor interface** can be enabled independently of the active radio mode for diagnostics and spectrum analysis.

15.2 BSS Mode (Basic Service Set)

Standard WiFi operation supporting Access Point (AP) and Station (STA) roles.

Use cases: - Provide WiFi connectivity to client devices - Connect to existing WiFi infrastructure - Create wireless bridge/repeater

Configuration:

Both AP and STA sections must be configured together. Enable the roles you need:

```
wifi:
  radio0:
    mode: "BSS"
    channel: 6
    ap:
      ssid: "LocalNetwork"
      passphrase: "password123"
      enabled: True
      zone: "data"
    sta:
      ssid: "UplinkNetwork"
      passphrase: "uplinkpass"
      enabled: False
      zone: "mgmt"
```

Roles: - **AP enabled**: Device broadcasts SSID, clients can connect - **STA enabled**: Device connects to existing network as client - **Both enabled**: Wireless bridge/repeater configuration

15.3 NAW Mode (Non-Associated WiFi)

Mesh networking mode for creating self-organizing wireless networks.

Use cases: - Two-node deployments with ACK-enabled transmission - Extended coverage without fixed infrastructure - Multi-node deployments requiring dynamic routing - Redundant wireless links

Configuration:

```
wifi:
  radio0:
    mode: "NAW"
    channel: 6
    naw:
      link_id: "MeshNetwork1"
      passphrase: "meshpassword"
      enabled: True
      zone: "data"
```

Behavior: - Nodes with same link_id discover each other - Mesh network forms automatically - Traffic routes through available paths - Network self-heals if nodes disconnect

Requirements: - All mesh nodes must use same link_id - All mesh nodes must use same channel - Passphrases must match for authentication

15.4 WFB Mode (WiFi Broadcast)

Long-range packet radio using raw WiFi frames with forward error correction.

Use cases: - Video transmission for drones/robots - Long-range telemetry links - One-way or bidirectional data streams

15.4.1 WFB Concepts

TX Stream: Transmitter listens on UDP port, encodes data with FEC, broadcasts via WiFi.

RX Stream: Receiver captures WiFi frames, decodes FEC, forwards data via UDP.

Stream ID: Identifies which stream to send/receive. TX and RX must use matching stream IDs.

FEC (Forward Error Correction): Adds redundancy to survive packet loss. Configured via fec_k and fec_n parameters: - `fec_k`: Number of data packets - `fec_n`: Total packets (data + redundancy) - Example: fec_k=8, fec_n=12 means 8 data packets + 4 redundancy packets

15.4.2 WFB Configuration

Each WFB stream must define `mode` as `tx` or `rx`.

Transmitter:

```
wifi:
  radio0:
    mode: "WFB"
    gi: 800
    mcs: 1

wfb:
  streams:
    tx:
      enabled: True
      mode: "tx"
      listen_port: 5600
      fec_k: 8
      fec_n: 12
      stream_id: 1
```

Receiver:

```
wifi:
  radio0:
    mode: "WFB"

wfb:
  streams:
    rx:
      enabled: True
      mode: "rx"
      forward_addr: 192.168.1.100
      forward_port: 5600
      stream_id: 1
```

15.4.3 WFB Keys

WFB uses encryption keys for secure communication. TX and RX devices must have matching keys.

Key files:

File	Purpose
`/etc/wfb-ng/tx.key`	Transmitter key
`/etc/wfb-ng/rx.key`	Receiver key

Generate new keys:

```
wfb_keygen
```

This generates a new keypair and writes to `/etc/wfb-ng/tx.key` and `/etc/wfb-ng/rx.key`.

Key distribution: Copy the generated keys to all devices in the WFB link. TX device needs `tx.key`, RX device needs `rx.key`.

Important: After generating or copying new keys, restart WFB service manually:

```
/etc/init.d/wfb-ng restart
```

15.4.4 WFB Parameters

MCS (Modulation and Coding Scheme):

MCS	Data Rate	Range	Reliability
0-1, 8-9	Low	Long	Best
2-3, 10-11	Medium	Long	Good
4-7, 12-15	High	Short	Lower

Lower MCS provides better range and reliability at lower data rates.

Guard Interval (gi): - `800` : Long guard interval — better multipath tolerance, slightly lower throughput - `400` : Short guard interval — higher throughput, less multipath tolerance

FEC Tuning:

Scenario	fec_k	fec_n	Overhead	Recovery
Low loss	8	12	50%	Good
Medium loss	4	8	100%	Better
High loss	2	6	200%	Best

Higher redundancy improves reliability but increases bandwidth usage.

15.4.5 WFB Usage

Video streaming example:

On transmitter (drone):

```
wifi:
  radio0:
    mode: "WFB"

wfb:
  streams:
    tx:
      enabled: True
      listen_port: 5600
      fec_k: 8
      fec_n: 12
      stream_id: 1
```

Send video to UDP port 5600 on the device. WFB transmits it over WiFi.

On receiver (ground station):

```
wifi:
  radio0:
    mode: "WFB"

wfb:
  streams:
    rx:
      enabled: True
      forward_addr: 127.0.0.1
      forward_port: 5600
      stream_id: 1
```

Received video is forwarded to localhost:5600 for display.

15.5 SCAN Mode

Dedicated mode for active spectral scanning. In this mode the radio is configured exclusively for spectrum analysis.

The scan frequency range is limited to regulatory channels defined by the configured country code. To scan the full extended frequency range, set `country: "CT"` (compliance test).

Use cases: - RF spectrum analysis and interference detection - Channel quality assessment

Configuration:

```
wifi:
  radio0:
    mode: "SCAN"
    channel: 6
```

15.6 Monitor Interface

A passive monitor interface can be enabled independently of the active radio mode. This allows packet capture and diagnostics alongside normal operation.

Configuration:

```
wifi:
  radio0:
    mode: "BSS"
    mon:
      enabled: True
```

15.7 Mode Comparison

Feature	BSS	NAW	WFB	SCAN
Standard WiFi	Yes	No	No	No
Client devices	Yes	No	No	No
Mesh networking	No	Yes	No	No
Long range	No	No	Yes	No
FEC support	No	No	Yes	No
Bidirectional	Yes	Yes	Configurable	No
Encryption	WPA2	WPA2	Separate keys	N/A
Spectral scan	Yes	Yes	No	Yes

15.8 Mode Selection Guidelines

Choose BSS when: - Connecting standard WiFi devices - Creating local management network - Bridging to existing WiFi infrastructure

Choose NAW when: - Deploying multiple nodes - Need dynamic routing between nodes - Want self-healing network topology

Choose WFB when: - Transmitting video over long distances - Need maximum range - Can tolerate one-way or limited bidirectional communication

15.9 Next Steps

- WiFi Controls - Shell commands and statistics
- Configuration - Full parameter reference
- Networking - Network zone configuration

16. WiFi Controls

16.1 Overview

This section describes shell commands for WiFi interface management and statistics retrieval.

16.2 Interface Information

16.2.1 List WiFi Interfaces

```
iw dev
```

Shows all wireless interfaces with their type, channel, and MAC address.

16.2.2 Interface Details

```
iw dev wlan0 info
```

Shows detailed information about specific interface.

16.2.3 List Connected Clients (AP Mode)

```
iw dev wlan0 station dump
```

Shows connected stations with signal strength and statistics.

16.3 Radio Information

16.3.1 Show Radio Capabilities

```
iw phy phy0 info
```

Shows supported frequencies, bandwidths, and features.

16.3.2 Current Channel

```
iw dev wlan0 info | grep channel
```

16.4 Statistics

16.4.1 Interface Statistics

```
iw dev wlan0 station dump
```

Output includes: - Signal strength (dBm) - TX/RX bytes and packets - TX/RX bitrate - Connection time

Example output:

```
Station 00:11:22:33:44:55 (on wlan0)
        signal:         -45 dBm
        tx bytes:       1234567
        rx bytes:       7654321
        tx bitrate:     72.2 MBit/s
        rx bitrate:     65.0 MBit/s
```

16.4.2 Link Quality (STA Mode)

```
iw dev wlan0 link
```

Shows connection status when in station mode: - Connected SSID - Signal strength - TX bitrate

16.5 Service Management

16.5.1 WiFi Services

hostapd - Access Point daemon:

```
/etc/init.d/hostapd status
/etc/init.d/hostapd restart
```

wpa_supplicant - Station mode daemon:

```
/etc/init.d/wpa_supplicant status
/etc/init.d/wpa_supplicant restart
```

wfb-ng - WiFi Broadcast service:

```
/etc/init.d/wfb-ng status
/etc/init.d/wfb-ng restart
```

16.5.2 Reload WiFi Configuration

After changing `/etc/roboconf/config.yaml`:

```
roboconf apply
```

Or reload wireless subsystem directly:

```
wifi reload
```

16.6 Spectral Scanning

The `spectral_scan` utility performs a single-shot spectral snapshot capture. Each invocation captures one set of spectral samples across the specified frequencies.

The scan frequency range is limited to regulatory channels defined by the configured country code. To scan the full extended frequency range, set the country to `CT` (compliance test) in the configuration.

```
# Scan all available frequencies, JSON output
spectral_scan -F all -j

# Scan specific frequency
spectral_scan -F 2412

# Scan frequency range
spectral_scan -F 2412-2462

# Scan current channel only
spectral_scan

# Save results to file
spectral_scan -F all -o /tmp/scan_result.json
```

Spectral scan data is also available through the web interface.

16.7 WFB Statistics

16.7.1 Check WFB Status

```
ps | grep wfb
```

Shows running WFB TX/RX processes.

16.7.2 Statistics Files

WFB statistics are collected to `/var/stats/` :

File	Description
`wfb-stats-tx.json`	TX statistics in JSON format
`wfb-stats-tx.txt`	TX statistics in text format
`wfb-stats-rx.json`	RX statistics in JSON format
`wfb-stats-rx.txt`	RX statistics in text format

View current statistics:

```
cat /var/stats/wfb-stats-tx.json
cat /var/stats/wfb-stats-rx.json
```

TX statistics include:

- Transmitted packets and bytes
- Dropped and truncated packets
- Input packets and bytes
- Timeout counters

RX statistics include:

- Received packets and bytes
- Good, recovered, and lost packets
- Invalid and corrupted packets
- RSSI (min/avg/max)
- SNR (min/avg/max)
- Output packets and bytes

16.7.3 WFB Process Output

WFB processes log statistics to system log:

```
logread | grep wfb
```

16.8 Troubleshooting

16.8.1 Interface Not Present

If `iw dev` shows no interfaces:

- Check radio is enabled:

```
iw phy
```

- Check for driver errors:

```
dmesg | grep ath
```

- Verify configuration:

```
uci show wireless
```

16.8.2 Cannot Connect (STA Mode)

If station fails to connect:

- Check wpa_supplicant status:

```
logread | grep wpa_supplicant
```

- Verify credentials in configuration

16.8.3 Clients Cannot Connect (AP Mode)

If clients cannot connect to AP:

- Verify AP is running:

```
iw dev wlan0 info
```

  Type should be "AP"

- Check hostapd status:

```
logread | grep hostapd
```

- Verify channel and security settings

16.8.4 Poor Signal Quality

If signal is weak or unstable:

- Check current signal level:

```
iw dev wlan0 station dump | grep signal
```

- Consider changing channel or increasing TX power in configuration

16.9 Next Steps

- WiFi Modes - Mode descriptions and configuration
- Configuration - Full parameter reference
- Monitoring - System monitoring