Busy Tag
AT Command Manual

SIA BUSY TAG

Firmware Version 3.0 (Wi-Fi)
July 7, 2026

Smart LED Status Display with Wi-Fi

USB CDC AT command interface

This document describes the complete USB/serial AT command set exposed by Busy Tag firmware 3.0 (Wi-Fi hardware, HW 2.0).

Contents

1 Introduction **4**
1.1 Key Features . . . 4
1.2 Hardware Specifications . . . 4

2 Getting Started **5**
2.1 USB Connection . . . 5
2.2 Testing the Connection . . . 5
2.3 Command Format . . . 5
2.4 Response Format . . . 5

3 Device Information **5**

4 LED Control **6**
4.1 Solid Color . . . 6
4.2 Custom Patterns . . . 7
4.3 Built-in Patterns . . . 7

5 Display Control **9**
5.1 Display Brightness . . . 9
5.2 Idle Backlight Dimming . . . 9
5.3 Showing Images . . . 9
5.4 On-screen Status Messages . . . 10
5.5 Status Toolbar . . . 10
5.6 Setup QR Code . . . 10

6 Time & Power **12**
6.1 Timezone . . . 12
6.2 Power Save . . . 12

7 Wi-Fi **12**
7.1 Wi-Fi Mode . . . 12
7.2 Station (Client) Connection . . . 12
7.3 Wi-Fi Scanning . . . 13
7.4 Access Point . . . 13

8 Cloud Connectivity **15**
8.1 WebSocket Status & URLs . . . 15
8.2 Network File Transfer . . . 15
8.3 Local HTTP Server . . . 16

9 File Management **17**
9.1 Storage Information . . . 17
9.2 File Operations . . . 17

10 System & Reset Commands **17**
10.1 Reset Reason Codes . . . 18

11 Asynchronous Events **18**

12 Error Codes **19**

13 Complete Command Reference **20**

14 Programming Example **22**
14.1 Python 22
14.2 C# 22
14.3 Swift 23

1 Introduction

Busy Tag is a smart status display. It is driven over a USB CDC serial link using an `AT` command protocol. Firmware 3.0 runs on the Wi-Fi hardware revision (HW 2.0) and adds Wi-Fi station/access-point networking, a cloud WebSocket connection, HTTP(S) file download/upload, a local HTTP server, an on-screen setup QR code, and a configurable status toolbar on top of the 2.x feature set.

1.1 Key Features

- **RGB LED bar** – solid colors, custom multi-step patterns, and 25 built-in animations.
- **LCD display** – shows stored PNG and GIF images, solid color fills, and a configurable status toolbar (Wi-Fi, temperature, clock).
- **Wi-Fi** – station, access-point, and combined STA+AP modes with scanning and diagnostics.
- **Cloud connectivity** – WebSocket client with Ed25519 device authentication, remote commands, and event reporting.
- **Network file transfer** – direct HTTP(S) downloads to storage (including OTA firmware) and uploads of stored files to the cloud.
- **Local HTTP server** – REST access to the same command set on the local network (used by mobile apps in AP mode).
- **USB CDC interface** – complete AT command set for control and file transfer.
- **Internal storage** – FAT file system with wear leveling for images and data files.
- **Power management** – dynamic frequency scaling, Wi-Fi modem sleep, and idle display dimming.
- **Temperature & diagnostics** – internal die temperature, reset reasons, chip, bootloader and USB line state info.

1.2 Hardware Specifications

Component	Specification
MCU	Dual-core Xtensa LX7, 240 MHz, Wi-Fi 802.11 b/g/n
Framework	ESP-IDF v5.x
Storage	Internal flash, FAT with wear leveling
Display	LCD, PNG + animated GIF support
LED	RGB LED bar (7 addressable positions)
Interface	USB CDC, Wi-Fi (STA / AP / STA+AP), local HTTP server
Hardware version	2.0 (reported by `AT+GHV`)

Table 1: Hardware overview

2 Getting Started

2.1 USB Connection

Connect Busy Tag to a host with a USB cable. The device enumerates as a USB CDC serial port. Open it with any serial terminal.

Serial Connection Parameters

- **Baudrate:** 460,800 bps
- **Data Bits:** 8 **Stop Bits:** 1 **Parity:** None
- **Flow Control:** None

2.2 Testing the Connection

```
> AT
OK
```

2.3 Command Format

- **Read / query:** `AT+<CMD>?` or `AT+<CMD>`
- **Write / set:** `AT+<CMD>=<parameters>`
- **Action:** `AT+<CMD>`

Every command line must terminate with `\r\n` (CR+LF). Commands are case-insensitive.

2.4 Response Format

- **Data + success:** a `+<PREFIX>:<data>` line followed by `OK`.
- **Simple success:** `OK`
- **Error:** `ERROR:<code>` – see Section 12.
- **Prompt:** `>` – the device is waiting for a data payload (file or pattern body).
- **Event:** `+evn:<type>[,<data>]` – asynchronous notification, see Section 11.
- **Log:** `+log:<text>` – diagnostic log line, informational only.

3 Device Information

```
> AT+GDN
+DN:busytag-5F83C0
OK

> AT+GFV
+FV:3.0
OK

> AT+GHV
+HV:2.0
OK
```

Command	Response	Description
AT+GDN	+DN:<name>	Get device name
AT+GMN	+MN:<name>	Get manufacturer name (SIA BUSY TAG)
AT+GID	+ID:<id>	Get unique device ID
AT+GFV	+FV:<version>	Get firmware version (3.0)
AT+GHV	+HV:<version>	Get hardware version (2.0)
AT+GESPV	+ESPV:<version>	Get ESP-IDF version
AT+GESPCT	+ESPCT:<date> <time>	Get firmware compile date/time
AT+GTEMP	+TEMP:<celsius>	Get internal die temperature
AT+GCHI	+CHI:<model>,<rev>,<cores>,<features>	Get chip info
AT+GUSBS	+USBS:<source>,<dtr>,<rts>	Get USB source and line state
AT+GBLV	+BLV:<version>	Get bootloader version
AT+GBLCD	+BLCD:<date> <time>	Get bootloader compile date/time
AT+GBLIDF	+BLIDF:<version>	Get bootloader ESP-IDF version

Table 2: Device information commands

```
> AT+GTEMP
+TEMP:42.5
OK

> AT+GCHI
+CHI:ESP32-S3,0,2,WiFi,BT,BLE,ExtPSRAM
OK

> AT+GUSBS
+USBS:HOST_APP,1,1
OK
```

- AT+GCHI <features> is a comma list drawn from WiFi, BT, BLE, EmbFlash, EmbPSRAM, ExtPSRAM. It reports chip silicon capabilities, not enabled firmware features.
- AT+GUSBS <source> is one of POWER (power-only supply), HOST (data host, port closed), HOST_APP (host with the serial port open), or SUSPENDED. <dtr>/<rts> are the current line states (0/1).

4 LED Control

The front RGB LED bar has 7 addressable positions, selected with a 7-bit bitmask (0–127). 127 (binary 1111111) selects all LEDs. Colors are given as RRGGBB hex.

4.1 Solid Color

Command	Description
AT+SC=<pins>,<RRGGBB>	Set solid color on selected LEDs
AT+SC?	Get current solid color: +SC:<pins>,<RRGGBB>

Table 3: Solid color commands

```
> AT+SC=127,FF0000          # all LEDs red
OK
```

```
# The pins value is a 7-bit mask, one bit per LED (LED1=1, LED2=2, LED3=4, ...)
    .
> AT+SC=7,00FF00          # first three LEDs (1+2+4) green
OK

> AT+SC?
+SC:127,FF0000
OK
```

4.2 Custom Patterns

A custom pattern is a sequence of steps, each: `<pins>,<RRGGBB>,<speed>,<delay>` where `speed` and `delay` are in milliseconds. Setting bit 7 of `<pins>` (add `128`) turns off the previously lit LEDs when the step is applied.

Command	Description
`AT+CP=<count>`	Set pattern; device replies `>` then expects `<count>` step lines
`AT+CPD=<lines>`	Set pattern directly in one line (no `>` prompt)
`AT+CP?`	Get the current pattern (one `+CP:` line per step)
`AT+PP=<play>,<repeat>`	Play/stop pattern; `repeat`=255 means infinite
`AT+PP?`	Get play state: `+PP:<playing>,<repeat_count>`

Table 4: Custom pattern commands

```
# Define a 2-step pattern (red, then green), each 100 ms with 500 ms hold
> AT+CP=2
>
+CP:127,FF0000,100,500
+CP:127,00FF00,100,500

OK

# Same 2-step pattern set in a single line.
# Steps use the +CP: prefix and are separated by a literal \r\n.
> AT+CPD=+CP:127,FF0000,100,500\r\n+CP:127,00FF00,100,500
OK

> AT+PP=1,3               # play 3 times
OK
> AT+PP=1,255             # play forever
OK
> AT+PP=0,0               # stop
OK
```

4.3 Built-in Patterns

Firmware 3.0 ships 25 built-in LED animations, selected by ID.

Command	Description
`AT+GDPL`	List built-in patterns: `+DPL:<id>,<name>` per line
`AT+DP=<id>`	Activate a built-in pattern by ID (0–24)

Table 5: Built-in pattern commands

ID	Name	ID	Name
0	Purple Glow	13	Yellow Running
1	Police 1	14	Magenta Running
2	Police 2	15	Cyan Running
3	Red Flashes	16	White Running
4	Green Flashes	17	Red Pulses
5	Blue Flashes	18	Green Pulses
6	Yellow Flashes	19	Blue Pulses
7	Magenta Flashes	20	Yellow Pulses
8	Cyan Flashes	21	Magenta Pulses
9	White Flashes	22	Cyan Pulses
10	Red Running	23	White Pulses
11	Green Running	24	Rainbow
12	Blue Running		

Table 6: Built-in pattern IDs

```
> AT+GDPL
+DPL:0,Purple Glow
+DPL:1,Police 1
...
+DPL:24,Rainbow
OK

> AT+DP=1                 # activate "Police 1"
OK
> AT+PP=1,255             # play it forever
OK
```

5 Display Control

5.1 Display Brightness

```
> AT+DB?
+DB:75
OK
> AT+DB=50                # set display brightness to 50%
OK
```

5.2 Idle Backlight Dimming

The LCD backlight can dim automatically after a period of inactivity. Both settings persist across restarts. Dimming is off by default.

Command	Description
AT+DD=<0|1>	Enable/disable idle auto-dim
AT+DD?	Get state: +DD:<0|1>
AT+DDC=<timeout_s>,<dim_level>	Configure: timeout 1–120 s, dim level 0–20 (PWM duty)
AT+DDC?	Get config: +DDC:<timeout_s>,<dim_level>

Table 7: Idle dimming commands

```
> AT+DD=1                 # enable idle dimming
OK
> AT+DDC=30,5             # dim to level 5 after 30 s idle
OK
> AT+DDC?
+DDC:30,5
OK
```

5.3 Showing Images

Busy Tag displays PNG and GIF files stored in its file system.

Command	Description
AT+GPL	List picture files: +PL:<name>,<size> per line
AT+SP=<file>	Show the named image; emits +evn:SP,<file>
AT+SP?	Get the currently shown image: +SP:<file>
AT+DC=<RRGGBB>	Fill the screen with a solid color
AT+DC?	Get the current fill color: +DC:<RRGGBB>

Table 8: Image and fill commands

```
> AT+GPL
+PL:logo.png,2048
+PL:animation.gif,8192
OK

> AT+SP=animation.gif
OK
+evn:SP,animation.gif

> AT+DC=FF8000            # fill the screen with orange
OK
```

```
> AT+DC?
+DC:FF8000
OK
```

5.4 On-screen Status Messages

The device can print informational messages (transfer progress, Wi-Fi state, errors) directly on the LCD. Persisted across restarts.

```
> AT+SM?
+SM:1
OK
> AT+SM=0                  # hide on-screen status messages
OK
```

5.5 Status Toolbar

A configurable toolbar at the top of the display can show a Wi-Fi indicator, the internal temperature, and a clock. Each item can be enabled individually and placed on the left or right side. All toolbar settings persist.

Command	Description
AT+TB=<0|1>	Show/hide the toolbar
AT+TB?	Get toolbar state: +TB:<0|1>
AT+TBI=<id>,<enabled>,<side>	Configure one item: id 0=Wi-Fi, 1=temperature, 2=clock; side 0=left, 1=right
AT+TBI?	Get item layout: +TBI:<id>,<enabled>,<side> per line
AT+TBR=<0|1>	Show/hide the numeric RSSI value next to the Wi-Fi icon
AT+TBR?	Get RSSI display state: +TBR:<0|1>
AT+BO=<0-100>	Set bar opacity (applies to the toolbar scrim and the status bar)
AT+BO?	Get bar opacity: +BO:<0-100>

Table 9: Toolbar commands

```
> AT+TB=1                  # show the toolbar
OK
> AT+TBI=2,1,1             # clock enabled, right side
OK
> AT+TBI?
+TBI:0,1,0
+TBI:1,0,0
+TBI:2,1,1
OK
> AT+BO=60                 # 60% bar opacity
OK
> AT+TBR=1                 # show RSSI next to the Wi-Fi icon
OK
```

5.6 Setup QR Code

The device can render a setup QR code on the display. The QR encodes a deep-link URL (see AT+SDLU) used by the mobile app to onboard the device.

```
> AT+QR=1                  # show the setup QR code
OK
> AT+QR?
+QR:1
```

```
OK
> AT+QR=0                  # hide it again
OK
```

6 Time & Power

6.1 Timezone

The clock (toolbar item 2) uses NTP time when Wi-Fi is connected, adjusted by the configured POSIX timezone string. Persisted.

```
> AT+TZ?
+TZ:CET-1CEST,M3.5.0,M10.5.0/3
OK
> AT+TZ=EST5EDT           # set US Eastern time
OK
```

6.2 Power Save

Power-save mode enables dynamic CPU frequency scaling, Wi-Fi modem sleep and adaptive transmit power. It reduces heat and power draw at a small cost in Wi-Fi latency. Persisted.

```
> AT+PS?
+PS:1
OK
> AT+PS=0                 # disable power save
OK
```

7 Wi-Fi

7.1 Wi-Fi Mode

Command	Description
AT+WM=<mode>	Set Wi-Fi mode: 0=OFF, 1=station, 2=access point, 3=STA+AP. Persisted.
AT+WM?	Get Wi-Fi mode: +WM:<mode>

Table 10: Wi-Fi mode commands

In AP and STA+AP modes the local HTTP server (Section 8.3) is started automatically (without authentication, since the AP network is private); in STA-only mode it is stopped.

```
> AT+WM=1                 # station mode
OK
> AT+WM?
+WM:1
OK
```

7.2 Station (Client) Connection

Command	Description
AT+STA=<ssid>,<password>	Connect to an access point
AT+STA?	Get status: +STA:<connected>,<ssid>,<ip>,<rssi>
AT+DSTA	Disconnect from the access point
AT+GIP	Get IP address: +IP:<ip> (0.0.0.0 when not connected)

Table 11: Station commands

```
> AT+STA=MyNetwork,MyPassword
OK
+evn:WIFI_CONNECTED,MyNetwork,192.168.1.42

> AT+STA?
+STA:1,MyNetwork,192.168.1.42,-58
OK

> AT+GIP
+IP:192.168.1.42
OK

> AT+DSTA
OK
```

If the connection fails, the device emits `+evn:WIFI_CONNECT_FAILED,<ssid>,<reason>` where `<reason>` is the ESP-IDF disconnect reason code.

7.3 Wi-Fi Scanning

Scanning is asynchronous: `AT+WSCAN=` starts a scan, `AT+WSCAN?` retrieves the results.

Command	Description
`AT+WSCAN=[ssid],[auth]`	Start a scan. Both filters are optional: `ssid` is a case-insensitive substring match, `auth` filters by security type (e.g. `OPEN`, `WPA2`, `WPA3`).
`AT+WSCAN?`	Get results: `+WSCAN:<count>` then one `+WSCAN:<ssid>,<channel>,<rssi>,<auth>` line per network. `+WSCAN:BUSY` while a scan is running.

Table 12: Wi-Fi scan commands

```
> AT+WSCAN=                # scan everything
OK

> AT+WSCAN?
+WSCAN:2
+WSCAN:MyNetwork,6,-58,WPA2
+WSCAN:CoffeeShop,11,-73,OPEN
OK
```

Auth strings: `OPEN`, `WEP`, `WPA`, `WPA2`, `WPA/WPA2`, `WPA3`, `WPA2/WPA3`, `WPA-ENT`, `WPA2-ENT`, `WPA3-ENT`, `WAPI`, `OWE`.

7.4 Access Point

The `<auth>` parameter is a numeric ESP-IDF auth mode (`0`=open, `2`=WPA-PSK, `3`=WPA2-PSK, `4`=WPA/WPA2-PSK, `6`=WPA3-PSK).

```
> AT+AP=BusyTag-Setup,setup1234
OK

> AT+AP?
+AP:BusyTag-Setup,1
OK

> AT+APST?
+APST:1
+APST:A4:C1:38:12:34:56,192.168.4.2,-40
OK
```

Command	Description
AT+AP=<ssid>,<password>[,<channel>][,<max_conn>][,<auth>]	Configure and start the access point. Defaults: channel 6, max connections 4, auth WPA2-PSK. Switches the device into AP mode if needed.
AT+AP?	Get AP status: +AP:<ssid>,<connected_stations> or +AP:DISABLED
AT+APST?	List connected stations: +APST:<count> then +APST:<mac>,<ip>,<rssi> per station
AT+APKICK=<mac>	Disconnect a station from the AP

Table 13: Access point commands

```
> AT+APKICK=A4:C1:38:12:34:56
OK
```

8 Cloud Connectivity

When connected as a Wi-Fi station, the device maintains a WebSocket connection to the Busy Tag cloud. The cloud can send the same AT commands remotely; the device authenticates with an Ed25519 key pair generated on first boot.

8.1 WebSocket Status & URLs

Command	Description
AT+WS?	Get WebSocket status: +WS:<state>,<url>
AT+WS=<url>	Set the WebSocket URL (wss://...); takes effect on next restart
AT+WSREG?	Get the registration URL: +WSREG:<url>
AT+WSREG=<url>	Set the registration URL (https://...); used on next registration
AT+SDLU?	Get the setup deep-link URL base: +SDLU:<url>
AT+SDLU=<url>	Set the deep-link URL base used by the setup QR code
AT+GCAK	Get the device's Ed25519 public key: +CAK:<hex>

Table 14: Cloud commands

WebSocket states: 0=disconnected, 1=connecting, 2=TLS handshake, 3=authenticating, 4=subscribed (fully online), 5=error.

```
> AT+WS?
+WS:4,wss://iot.busytag.com/ws
OK

> AT+GCAK
+CAK:3B6A27BCCEB6A42D62A3A8D02A6F0D73653215771DE243A63AC048A18B59DA29
OK
```

8.2 Network File Transfer

Command	Description
AT+DL=<url>,<filename>[,<size>]	Download a file over HTTP(S) into storage. A .bin filename triggers an OTA firmware update. The optional <size> (bytes) enables percentage progress when the server sends no Content-Length.
AT+UL=<url>,<filename>	Upload a stored file to the cloud via HTTPS POST. The device streams /data/<filename> and sends its SHA-256 for server-side verification.
AT+DPR=<0|1>	Enable/disable transfer progress events
AT+DPR?	Get progress reporting state: +DPR:<0|1>

Table 15: Network transfer commands

```
> AT+DPR=1
OK

> AT+DL=http://example.com/pic.png,pic.png
+evn:DL,pic.png,51200,204800,25,180.5,8
+evn:DL,pic.png,204800,204800,100,195.2,0
+DL:OK,pic.png
OK

# Firmware OTA: download a .bin, update runs after the transfer
```

```
> AT+DL=http://example.com/fw.bin,fw.bin
+evn:FWDL,STARTING
...
+evn:FWDL,DOWNLOADED
+evn:FU,42.00%
...
+DL:OK,fw.bin,UPDATE_PENDING
OK

> AT+UL=https://iot.busytag.com/upload,logo.png
+evn:UL,logo.png,102400,102400,100,150.0,0
+UL:OK,logo.png
OK
```

Progress events have the form `+evn:DL,<file>,<current>,<total>,<percent>,<kB/s>,<eta_s>` (`UL` for uploads) and are throttled to every 5 % or 500 ms.

8.3 Local HTTP Server

The device runs a local HTTP server that exposes the AT command set over REST, used by the mobile apps when connected to the device's AP.

Command	Description
`AT+LS=<enable>,<port>,<auth_token>`	Configure the server. Port 1024–65535. A non-empty token enables authentication.
`AT+LS?`	Get status: `+LS:<enabled>,<port>,<auth_required>,<RUNNING|STOPPED`
`AT+LT?`	Get the current auth token: `+LT:<token>`
`AT+LT`	Regenerate the auth token: `+LT:<new_token>`

Table 16: Local server commands

```
> AT+LS?
+LS:1,8080,1,RUNNING
OK

> AT+LT?
+LT:a1b2c3d4e5f6
OK

> AT+LT                    # rotate the token
+LT:f6e5d4c3b2a1
OK
```

9 File Management

9.1 Storage Information

```
> AT+GFSS                 # free storage, bytes
+FSS:1048576
OK
> AT+GTSS                 # total storage, bytes
+TSS:3145728
OK
```

9.2 File Operations

Command	Description
AT+GFL	List all files: +FL:<name>,<type>,<size> per line
AT+UF=<file>,<size>	Upload a file over USB; device replies > then expects <size> bytes
AT+GF=<file>	Download a file from the device: +GF:<file>,<size> then the raw bytes
AT+DF=<file>	Delete a file
AT+AFSS	Re-scan the file storage

Table 17: File operation commands

```
> AT+GFL
+FL:config.json,json,256
+FL:def.png,png,1024
OK

# Upload a 12-byte file
> AT+UF=test.txt,12
>
Hello World!
OK

# Read a file back
> AT+GF=config.json
+GF:config.json,256
{ ...file content... }

> AT+DF=old_file.txt
OK
```

File transfer timeout

After the > prompt the device waits up to 10 seconds for the full payload. If the byte count is not met in time the transfer aborts with a timeout error.

10 System & Reset Commands

WARNING

AT+FD erases all stored files. AT+FR additionally erases every persisted setting – Wi-Fi credentials, cloud registration, display and LED configuration – and restarts the device in its factory state. Use with caution.

Command	Description
AT+GLEC	Get last error code: +LEC:<code>
AT+GLRR0	Get last reset reason for core 0
AT+GLRR1	Get last reset reason for core 1
AT+RST	Restart the device
AT+FD	Format storage
AT+FR	Factory reset: format storage, erase all settings (NVS), restart

Table 18: System and reset commands

10.1 Reset Reason Codes

AT+GLRR0 and AT+GLRR1 return the last reset reason for CPU core 0 and core 1 as a numeric code (+LRR0:<code> / +LRR1:<code>).

Code	Meaning
0	No reset reason recorded
1	Power-on reset (Vbat)
3	Software reset of the digital core
5	Deep-sleep wake reset
7	Timer Group 0 watchdog reset (core)
8	Timer Group 1 watchdog reset (core)
9	RTC watchdog reset (core)
10	Intrusion-detection reset
11	Timer Group 0 watchdog reset (CPU)
12	Software reset of the CPU
13	RTC watchdog reset (CPU)
15	Brownout reset (unstable VDD)
16	RTC watchdog reset (core + RTC)
17	Timer Group 1 watchdog reset (CPU)
18	Super watchdog reset
19	Clock-glitch reset
20	eFuse CRC reset
21	USB UART reset
22	USB JTAG reset
23	Power-glitch reset

Table 19: Reset reason codes (AT+GLRR0 / AT+GLRR1)

11 Asynchronous Events

The device emits unsolicited +evn: lines on the serial port. They can arrive at any time, including between a command and its OK.

Event	Meaning
+evn:SP,<file>	An image was shown on the display
+evn:WIFI_CONNECTED,<ssid>,<ip>	Station connected and got an IP address
+evn:WIFI_CONNECT_FAILED,<ssid>,<reason>	Station connection failed (ESP-IDF reason code)
+evn:DL,<file>,<cur>,<total>,<pct>,<kBps>,<eta>	Download progress (when AT+DPR=1)
+evn:UL,<file>,<cur>,<total>,<pct>,<kBps>,<eta>	Upload progress (when AT+DPR=1)

Event	Meaning
`+evn:DL_START,<file>`	Cloud-initiated download started
`+evn:DL_COMPLETE,<file>,<bytes>`	Cloud-initiated download finished
`+evn:DL_ERROR,<file>`	Cloud-initiated download failed
`+evn:UL_START,<file>`	Upload to cloud started
`+evn:UL_COMPLETE,<file>,<bytes>`	Upload to cloud finished
`+evn:UL_ERROR,<file>`	Upload to cloud failed
`+evn:FWDL,STARTING`	Firmware download starting
`+evn:FWDL,DOWNLOADED`	Firmware image downloaded, update will run
`+evn:FWDL,FAILED`	Firmware download or update failed
`+evn:FU,<percent>%`	Firmware update flash progress

Table 20: Asynchronous events

Diagnostic log lines are prefixed `+log:` and can be ignored by host software.

12 Error Codes

On failure the device responds with `ERROR:<code>`. The last error is also retrievable with `AT+GLEC`.

Code	Meaning
-1	None
0	Unknown error
1	Unknown command
2	Invalid argument
3	File not found
4	Invalid size
5	Storage not mounted
6	File list error
7	Storage info error
8	Memory allocation error
9	Wi-Fi scan error
10	Token error
11	Sensor error
12	File delete error
13	File reception timeout

Table 21: Error codes

Changed from firmware 2.0

Codes 9 (Wi-Fi scan error) and 10 (token error) are new in 3.0; the sensor, file-delete and timeout codes moved from 11/12/13 in 2.0 accordingly.

13 Complete Command Reference

Command	Type	Description
AT	Action	Test connection
AT+GDN	Get	Device name
AT+GMN	Get	Manufacturer name
AT+GID	Get	Device ID
AT+GFV	Get	Firmware version
AT+GHV	Get	Hardware version
AT+GESPV	Get	ESP-IDF version
AT+GESPCT	Get	Compile date/time
AT+GTEMP	Get	Internal temperature
AT+GCHI	Get	Chip info
AT+GUSBS	Get	USB source and line state
AT+GBLV	Get	Bootloader version
AT+GBLCD	Get	Bootloader compile date
AT+GBLIDF	Get	Bootloader ESP-IDF version
AT+GLEC	Get	Last error code
AT+GLRR0	Get	Last reset reason, core 0
AT+GLRR1	Get	Last reset reason, core 1
AT+SC?/=	Get/Set	Solid LED color
AT+CP?/=	Get/Set	Custom pattern
AT+CPD=	Set	Custom pattern (direct)
AT+PP?/=	Get/Set	Play pattern
AT+GDPL	Get	Built-in pattern list
AT+DP=	Set	Activate built-in pattern by ID
AT+DB?/=	Get/Set	Display brightness
AT+DD?/=	Get/Set	Idle backlight dimming
AT+DDC?/=	Get/Set	Idle dimming config
AT+SP?/=	Get/Set	Show picture
AT+DC?/=	Get/Set	Display solid color fill
AT+SM?/=	Get/Set	On-screen status messages
AT+TB?/=	Get/Set	Status toolbar
AT+TBI?/=	Get/Set	Toolbar item layout
AT+TBR?/=	Get/Set	Toolbar Wi-Fi RSSI display
AT+BO?/=	Get/Set	Bar opacity
AT+QR?/=	Get/Set	Setup QR code
AT+GPL	Get	Picture list
AT+TZ?/=	Get/Set	Timezone (POSIX TZ)
AT+PS?/=	Get/Set	Power save
AT+WM?/=	Get/Set	Wi-Fi mode
AT+STA?/=	Get/Set	Station status / connect
AT+DSTA	Action	Disconnect station
AT+GIP	Get	IP address
AT+WSCAN?/=	Get/Set	Wi-Fi scan
AT+AP?/=	Get/Set	Access point
AT+APST?	Get	AP connected stations

Command	Type	Description
AT+APKICK=	Action	Kick station from AP
AT+WS?/=	Get/Set	Cloud WebSocket status / URL
AT+WSREG?/=	Get/Set	Registration URL
AT+SDLU?/=	Get/Set	Setup deep-link URL base
AT+GCAK	Get	Cloud auth public key (Ed25519)
AT+DL=	Action	Download file from URL (OTA for .bin)
AT+UL=	Action	Upload stored file to cloud
AT+DPR?/=	Get/Set	Transfer progress reporting
AT+LS?/=	Get/Set	Local HTTP server
AT+LT?	Get	Local server auth token
AT+LT	Action	Regenerate auth token
AT+GFL	Get	File list
AT+GFSS	Get	Free storage size
AT+GTSS	Get	Total storage size
AT+UF=	Action	Upload file (USB)
AT+GF=	Action	Get/download file (USB)
AT+DF=	Action	Delete file
AT+AFSS	Action	Re-scan storage
AT+RST	Action	Restart
AT+FD	Action	Format disk
AT+FR	Action	Factory reset

Table 22: Complete AT command reference

14 Programming Example

All examples open the USB CDC serial port at `460800` baud, send each command terminated with `\r\n`, and read the `+.../OK` reply line. Host software must be prepared for unsolicited `+evn:` and `+log:` lines between replies.

Note

The serial port name varies by operating system: on Windows it is a COM port (e.g. `COM3`), on macOS a `/dev/tty.usbmodem*` device, and on Linux a `/dev/ttyACM*` device. Substitute the value that matches your host.

14.1 Python

```python
import serial

ser = serial.Serial('COM3', 460800, timeout=1)

def cmd(c):
    ser.write((c + "\r\n").encode())
    while True:
        line = ser.readline().decode().strip()
        # Skip asynchronous event/log lines
        if line.startswith("+evn:") or line.startswith("+log:"):
            continue
        return line

print(cmd("AT"))                        # OK
print(cmd("AT+GFV"))                    # +FV:3.0
cmd("AT+SC=127,0000FF")                 # all LEDs blue
cmd("AT+STA=MyNetwork,MyPassword")      # join Wi-Fi
cmd("AT+DL=http://example.com/pic.png,pic.png")
cmd("AT+SP=pic.png")                    # show the downloaded image

ser.close()
```

14.2 C#

```csharp
using System;
using System.IO.Ports;

var port = new SerialPort("COM3", 460800) { NewLine = "\r\n", ReadTimeout =
    1000 };
port.Open();

string Cmd(string c)
{
    port.WriteLine(c);
    while (true)
    {
        var line = port.ReadLine().Trim();
        if (line.StartsWith("+evn:") || line.StartsWith("+log:"))
            continue;   // skip asynchronous lines
        return line;
    }
}

Console.WriteLine(Cmd("AT"));          // OK
Console.WriteLine(Cmd("AT+GFV"));      // +FV:3.0
```

```
Cmd("AT+SC=127,0000FF");              // all LEDs blue
Cmd("AT+STA=MyNetwork,MyPassword");   // join Wi-Fi
Cmd("AT+SP=logo.png");                // show an image

port.Close();
```

14.3 Swift

Uses the `ORSSerialPort` library (macOS). Replies are delivered to the port's delegate; only the writes are shown here.

```
import ORSSerial

let port = ORSSerialPort(path: "/dev/tty.usbmodem1101")!
port.baudRate = 460800
port.open()

func cmd(_ c: String) {
    port.send((c + "\r\n").data(using: .utf8)!)
}

cmd("AT")                                 // OK
cmd("AT+GFV")                             // +FV:3.0
cmd("AT+SC=127,0000FF")                   // all LEDs blue
cmd("AT+STA=MyNetwork,MyPassword")        // join Wi-Fi
cmd("AT+SP=logo.png")                     // show an image

port.close()
```

Thank you for choosing Busy Tag!
SIA BUSY TAG