ChatGPT

# IRC Tramp VTX Control Protocol (Betaflight)

**Serial interface:** Tramp uses a half-duplex UART at 9600 baud by default. In Betaflight the VTX TX pin is driven push–pull (only a TX pin, no hardware CTS/RTS) with `vtx_halfduplex = ON` [1]. The UART format is 8 data bits, no parity, 1 stop bit (8N1) at 9600 bps [2]. (This matches ArduPilot's Tramp settings: AP_TRAMP_UART_BAUD = 9600, 16-byte frames [2].)

**Packet framing:** Every Tramp frame is fixed at 16 bytes. Byte 0 is always 0x0F (`TRAMP_SYNC_START`) and byte 15 is 0x00 (`TRAMP_SYNC_STOP`) [3]. The FC fills bytes 1…13 with the command code and any parameters, then computes an 8-bit checksum as the byte-sum of bytes 1–13 [4]. This checksum is written into byte 14, and byte 15 is left 0. In other words, a command packet looks like:

- **Byte 0:** 0x0F (start marker)
- **Byte 1:** command code (ASCII)
- **Bytes 2–13:** payload (parameters, zero-padded as needed)
- **Byte 14:** CRC (8-bit sum of bytes 1–13) [4]
- **Byte 15:** 0x00 (stop marker) [3]

The VTX replies use the same 16-byte format.

**Protocol flow:** Communication is request–response. Crucially, the FC must first send the *initialize* command (`TRAMP_COMMAND_CMD_RF`) to wake up the VTX and learn its limits [5]. Only after the VTX responds to that can the FC send other queries or set-commands. In Betaflight (and ArduPilot) the init command is ASCII "R" (0x52) and the VTX responds with "r" (0x72) [5] [6]. The response to init contains the VTX's frequency/power limits (see below). Once initialized, the FC can send *queries* that expect a reply and *commands* that require no immediate response [5] [7]. Set-commands (e.g. set power or frequency) do not require an acknowledgement [7]; the FC typically follows them with a read-query to verify the new setting.

**Commands and data:** Betaflight's Tramp implementation uses the following commands (all sent in byte 1 of the packet, with parameters in bytes 2–3 or more as needed):

- `R` **(0x52): Init/Ping.** No parameter. VTX responds (code "r"0x72) with device limits. The reply's bytes 2–3 = min frequency (MHz), bytes 4–5 = max frequency, bytes 6–7 = max power (mW) [8]. (These allow the FC to know the allowed range.)

- `F` **(0x46): Set frequency.** Parameter = 16-bit frequency in MHz (e.g. 5860). No response is returned. *Betaflight uses "CUSTOM" mode for Tramp*, i.e. it sends the raw frequency rather than a band/channel index [9] [10].

- `P` **(0x50): Set power.** Parameter = 16-bit power in mW (e.g. 400). No response is returned. Tramp uses a *fixed* set of power values; Betaflight requires using exactly {25, 100, 200, 400, 600} mW [11]. (These are configured via `vtxtable powervalues 25 100 200 400 600` [11].)

- `I` **(0x49): Set pit/race mode.** Parameter = 0 or 1 (typically 0 = enter Pit mode (low power), 1 = normal). No response is returned. The VTX will report the current pit/race state in status queries (below).

- `v` **(0x76): Get settings/status.** No parameter. VTX responds (code "v") with its current operating state. The reply payload contains: frequency (bytes 2–3), configured power (4–5), race/pit flag (byte 7), and actual output power (bytes 8–9) [12]. Betaflight uses this to read back and confirm the new settings.

- `s` **(0x73): Get temperature.** No parameter. VTX responds (code "s") with telemetry such as temperature. The reply's bytes 6–7 hold a signed temperature (centi°C) [13]. (This is rarely needed just for setting channel/power.)

The checksum is computed over bytes 1–13 (see above) and inserted into the packet [4]. All replies must also be 16 bytes (same format).

**LUTs and values:** Note that Tramp *does not use a channel/band table*. It sends raw frequencies (CUSTOM mode) because Tramp VTXs have no built-in band table [10]. (Betaflight still displays a band/channel via vtxtable, but it is only for user interface, not sent to the VTX.) By contrast, the *power* levels are fixed: Betaflight's recommended powervalues for IRC Tramp are 25, 100, 200, 400, 600 mW [11].

In summary, the FC exchanges 16-byte packets (first byte 0x0F, last byte 0x00) with the Tramp VTX. It always begins by sending "R" (0x52) to init, then uses "F", "P", "I" to set, and "v", "s" to query, following the above format and CRC. This matches the Betaflight implementation (and ArduPilot port) exactly [14] [11], so the same spec can be used in a custom flight controller.

**References:** Betaflight's VTX spec and code (via ArduPilot port) document the framing and commands [14] [11]. The Betaflight CLI help confirms half-duplex UART, and the AP_Tramp code (ported from BF) shows the 9600 baud, 16-byte frames and checksum [1] [4]. The Betaflight VTX table docs list the Tramp power levels [11].

---

[1] Command Line Interface (CLI) | Betaflight
https://betaflight.com/docs/development/Cli

[2] [4] [6] [8] [12] [13] LCOV - lcov.info - AP_VideoTX/AP_Tramp.cpp
https://firmware.ardupilot.org/coverage/AP_VideoTX/AP_Tramp.cpp.gcov.html

[3] [5] [7] [14] Управление параметрами FPV видеопередатчиков по протоколам IRC Tramp и TBS SmartAudio / Хабр
https://habr.com/ru/articles/731514/

[9] [10] [11] VTX | Betaflight
https://betaflight.com/docs/development/VTX