
Mustang 2 Changelog

All notable changes to the Mustang 2 firmware (Teensy 4.1, Mustang2Sketch/) and its Raspberry Pi companion display (PiDisplay/) are logged here, newest first. The Teensy's on-screen VERSION string (SDT.h) is the source of truth for what's actually running on the radio — if it doesn't match the version heading below, reflash before trusting anything else in these notes.

Firmware changes require reflashing the Teensy. Pi-only changes are noted as such and only need PiDisplay/ re-synced to the Pi and radio_ui.py restarted.

2026-10-04 — V1.0.34 released (website downloads updated) — current

- Mustang 2 **V1.0.34** built and posted on the website: Mustang2_V1.0.34_SerialMIDIAudio.hex (with digital mode), Mustang2_V1.0.34_DualSerial.hex, and Mustang2_V1.0.34_Source.zip (Mustang2Sketch + PiDisplay + changelogs). V1.0.34 includes everything from V1.0.26 to V1.0.33 - see the firmware changelog.
- Matching Pi side: PiDisplay as of this entry (phone remote V1.0.55). Flash the Teensy and copy PiDisplay to the Pi together.

2026-10-04 — Decoded CW text back on the Pi/remotes; Vol/AGC off the CW page (Pi/web only, no reflash)

- Fix:** after the CW settings page change the Pi only accepted the new, longer CW message (42 bytes). A radio still running the first V1.0.33 build sends the shorter 24-byte one, so the Pi silently dropped every frame and the decoded text vanished from the Pi and remotes (the T41 itself was unaffected). The Pi now accepts both: the text always works, and the CW page values show “-” until the radio has the newer build - now numbered **V1.0.34** so the two can be told apart.
- Vol: and AGC: removed** from the CW settings page on the Pi and both remotes; the top row is just the red CW title. Mobile remote is now V1.0.55.

2026-10-04 — CW settings page on the Pi and remotes (firmware V1.0.33)

- While **Decode is on**, the Settings pane on the Pi and both web remotes now switches to the T41's **CW settings page**, exactly when the T41's own pane does: Vol / **CW** / AGC across the top; Decoder (with the green/red lock box), AT Grace, Mode, Key Type, Dit(ms), Upd Age on the left; WPM, SNR, AvgD/D, Coeff on the right; then Temp(F) and Load. It switches back to the normal page when Decode is turned off.
- Values follow the T41's own rules (“-” until the decoder is running/locked).
- On the phone the page is packed into the small Settings box in smaller text and the version label is hidden while it shows.
- Firmware: the V1.0.33 CW_DECODE message now also carries the CW page values (42 bytes). Still V1.0.33 (not yet built). Mobile remote is now V1.0.54.

2026-10-04 — Decoded CW text under the waterfall (firmware V1.0.33)

- In CW receive with Decode on, the decoded text now shows on a line **below the waterfall on the Pi** (the T41's own State of Health spot), with “CW” in green when the decoder is locked (yellow while searching) and the decoded speed (e.g. “22 WPM”) on the right.
- The T41 only keeps its last 15 characters; the Pi keeps a longer running line (as much as fits, newest on the right).
- Web remotes (PC and phone):** the same line appears as a strip along the bottom of the waterfall, newest characters on the right, older ones scrolling off the left. A browser that connects mid-QSO gets the current line straight away.
- Hidden outside CW receive or with Decode off, like the T41. **Requires firmware V1.0.33.** Mobile remote is now V1.0.53.

2026-10-04 — Brighter 3D waterfall (Pi/web only, no reflash)

- The 3D waterfall's blue no longer looks muted: rows now fade to half brightness at the back instead of 18%, and the front half stays near full brightness; the lowest colours are a brighter blue; and every row keeps a thin blue base so the noise reads as a blue carpet instead of thin lines on black. Same on the Pi and both web remotes. Mobile remote is now V1.0.52.

2026-10-04 — Waterfall uses the T41's own colours, so the noise blue matches (firmware V1.0.32)

- Cause found:** the Pi and web remotes coloured the waterfall from the spectrum trace's height, but the T41 colours its waterfall from where the trace sits on its own screen (230 - y), after its auto-floor shift and trace smoothing and on a steeper scale. So the T41's noise landed in the bright-blue part of the colour table while the Pi's landed in the near-black part, however much the blues were brightened.
- Fix:** firmware V1.0.32 sends the T41's own colour index for every waterfall column (new PILINK_MSG_WATERFALL, 0x0D), and the Pi and both remotes colour the 2D waterfall with it. The trace and the 3D waterfall are unchanged. With older firmware the waterfall falls back to the old method.
- This morning's brighter blues are kept on top of this. Mobile remote is now V1.0.51. **Requires firmware V1.0.32.**

2026-10-04 — Powers up in DATA mode (firmware V1.0.31); waterfall blue much brighter (Pi/web)

- **DATA mode at power-up:** the radio now starts in DATA instead of SSB. Controlled by `#define STARTUP_MODE_DATA` in `Config.h` (comment it out to go back to SSB); only applies to USB-audio builds, the only ones with DATA. **Requires firmware V1.0.31.**
- **Waterfall blue much brighter** on the Pi and both web remotes: pure blue is now `#1f80ff` (was `#004cff` this morning, `#0000ff` originally) and the noise floor is a clearer blue; the 3D waterfall's two lowest colours brightened again to match. Cyan and up unchanged. Mobile remote is now V1.0.50.

2026-10-04 — Brighter waterfall blue; frequency readout hidden on calibration screens (Pi/web only, no reflash)

- **Waterfall blue is brighter** on the Pi and both web remotes: the blue part of the Teensy's colour table (quiet signals and noise floor) is lifted to a clear azure; cyan, green, yellow, red and magenta are unchanged. The 3D waterfall's two lowest colours are brightened to match.
- **Pi: the frequency readout no longer shows through the Transmit IQ calibration screen** (or any other full-screen page). With Freq Font set to Nixie/VFD/Dot Matrix, the digits were redrawn on top of the screen whenever the frequency changed; the full-screen page is now put back on top after each redraw.
- Web: the calibration overlay background is now solid black, so nothing behind it shows through. Mobile remote is now V1.0.49.

2026-10-04 — Calibration screens: Transmit IQ, Frequency, Power and S meter level on the Pi and web remotes (firmware V1.0.30)

- **Menu > Calibration** items S meter level, Frequency, Transmit IQ and Power are no longer greyed out on the Pi or either web remote.
- **S meter level** is an ordinary Menu value, so it opens the usual parameter box (wheel / arrow keys, Select to confirm) with the S-meter still visible behind it.
- **Transmit IQ, Frequency and Power** each get a full-screen copy of the T41 screen on the Pi (new `cal_screens.py`) and an overlay on both web remotes, with buttons for every control the T41's own instructions list:
 - Transmit IQ: Amp, Phase, Atten (loff on direct-coupled builds), Incr, Auto, Stop, Band, Exit, Save & Exit; IRR/dBc, the current-band pane, the all-bands table and the live spectrum with the wanted (blue) and suppressed (red) sideband windows.
 - Frequency: Factor, Incr, Modulation, Exit, Save & Exit; Factor, Incr., Error and Mod (green when SAM), plus the tuned frequency and a live spectrum.
 - Power: Record (Select), Atten, Power, Incr, Units W/dBm, PA 20/100W, Next step, Reset, Band, Exit, Save & Exit; Attenuation, measured/target power, the step list with ✓/► marks, the PA/Band/Freq/Transmit/Increment/Units pane and the Psat/k/Gain table.
- Arrow keys work on these screens (Pi and web): Transmit IQ Up/Down = Amp, Left/Right = Phase; Power Up/Down = Power, Left/Right = Atten; Frequency all arrows = Factor. The Pi's mouse wheel turns the Filter knob (Phase / Factor / Power).
- Transmitting during Transmit IQ and Power calibration still needs the radio's own PTT/key (Transmit IQ Auto keys it automatically, as on the T41).
- **Requires firmware V1.0.30** (new telemetry). Mobile remote is now V1.0.48.

2026-10-03 — Spectrum Fill starts Off at every boot (firmware V1.0.29 + Pi/web)

- **Radio (T41) Spectrum fill:** firmware V1.0.29 changes the power-up default of `spectrumFillTraceEnabled` from 1 to 0 (SDT.h). The setting isn't stored in EEPROM/SD, so the radio now always boots in outline mode; Menu > Display > Spectrum Fill > On turns it on until the next power cycle. **Requires reflashing V1.0.29.**
- **Pi / web remotes PI Spectrum Fill:** now starts Off on every start of `radio_ui.py` and is no longer restored from `pi_prefs.json`; Display > PI Spectrum > PI Spectrum Fill > On turns it on for that session. The Pi menu marker and both web remotes show Off at startup. Other remembered Pi settings (3D waterfall, speed, Freq Font, brightness) are still remembered.
- Mobile remote is now V1.0.47.

2026-10-03 — Layout: REC/SWR right, 3-row Pi toolbar, Vol: to settings, phone fits again (Pi/web only, no reflash)

Re-sync `PiDisplay/`, restart `radio_ui.py`, refresh browsers. Mobile V1.0.46.

- **Pi screen layout:**
 - REC moved to the right edge of the screen, and the SWR/PWR readout moved right to sit just beside it.
 - Toolbar in three fixed rows. Row 1: Menu, Display, Audio, Mic, CW, Vol, Band, Mode, Demod. Row 2: Zoom, Filter, Noise, Decode, Home, Course, Fine. Row 3: Wheel:.
- **Vol: moved from the data line above the spectrum to the settings pane** (Pi: the free slot beside TX Atten; web/phone: settings grid). The data line's remaining 8 readouts are re-spaced. On the Pi the gaps are sized from typical values, so they look even in normal use.
- **After the Vol button, Up/Down arrows also change the volume** (web and Pi), not just Left/Right.
- **Phone: Filter slider usable again.** The Audio/Mic/CW buttons had made the page taller than the screen.

- Toolbar is now a 6-column grid (3 rows instead of 5), with slightly smaller labels.
- The data line fits on one row.
- The spectrum area shrinks properly: its canvas is absolutely positioned, so its pixel size no longer holds the row open. Its minimum height is 84px.
- The settings box is 100px with 12px text, and dropdowns in the right-hand columns open leftward.
- In testing at 390x664 (about the smallest iPhone Safari viewport), all four sliders are fully on screen.

2026-10-03 — Audio / Mic / CW dropdowns; Vol becomes a button (Pi/web only, no reflash)

Re-sync PiDisplay/, restart `radio_ui.py`, refresh browsers. Mobile V1.0.45.

- **New Audio, Mic and CW toolbar dropdowns** (Pi, web, phone), next to Display. They are the T41's own **Audio Options**, **Microphone** and **CW Options** menus, copied item for item and in the same order (e.g. Audio: AGC Off/Long/Slow/Medium/Fast, Adjust Equalizers, Toggle Aut notch, the four Noise Reduction choices; Mic: Mic gain, USB drive, USB RX level; CW: WPM ... CW Manual Mode). Picks send the same menu selection as before (and jump straight there with firmware V1.0.28+). Those three categories are no longer listed under Menu.
- **Vol is now a plain button.** One click sets the radio's Volume knob back to audio volume (from AGC/Mic/STn if needed) and points the mouse wheel and Left/Right arrows at it. Mic gain, sidetone and AGC are reached through the new Mic / CW / Audio dropdowns.
- Pi toolbar: rows are now split by width (the top row runs further right, where there's room), and "Course Inc"/"Fine Inc" are shortened to "Course"/"Fine" like the web, so the second row stays clear of the SWR readout. Phone: dropdowns in the right-hand toolbar columns open toward the left so they stay on screen.
- Pi: `_OWN_DROPDOWN_CATEGORIES / _OWN_DROPDOWN_LABELS, _make_plain_button()`. Web: `buildMenu(..., {only})` for each, `buildVolButton()`.

2026-10-03 — Web: arrow keys reach Filter; phone layout with Freq Font (web only)

Refresh browsers only (just web/remote.html and web/remote_mobile.html changed). Mobile V1.0.44.

- **Arrow keys now turn the Filter (and Volume) knob on the web remotes.** After you picked Lo Cut / Hi Cut from the Filter dropdown, the dropdown kept keyboard focus, so the arrows changed its own selection (flipping between Lo and Hi) instead of reaching the radio. Toolbar dropdowns now give focus back after a pick, and the arrow handler no longer skips a closed toolbar dropdown.
- **Phone: Freq Font styles no longer push the page down.** The styled readout canvas was 6px taller than the normal frequency text, which shoved the bottom (Filter) jog slider partly off screen. It now matches the text's measured height, so nothing below it moves.

2026-10-03 — Trackpad fix + arrow keys for Filter/Volume (Pi/web only, no reflash)

Re-sync PiDisplay/, restart `radio_ui.py`, refresh browsers.

- **Trackpad fix (web remotes):** filter Lo/Hi Cut (and every other wheel use) jumped around with a trackpad, because each of the trackpad's many tiny scroll events, including sideways ones, sent a full encoder step. Now:
 - vertical movement is added up and sent as one step per 40px of travel (a mouse-wheel notch is still exactly one step);
 - sideways swipes are ignored;
 - a change of direction restarts the count;
 - steps are at least 35 ms apart. The new code is `wheelStep()`.
- **Arrow keys on every version (Pi, web, phone with a keyboard):**
 - **Left/Right** turn whichever knob the wheel is set to (Filter Lo/Hi Cut or Volume, picked from those dropdowns), one step per press. With nothing picked they tune.
 - **Up/Down** always tune (Shift = fine tune).
 - In a Menu parameter box every arrow turns the Filter knob; on the equalizer screen Left/Right pick the band and Up/Down change its level.
 - Pi: new `display.bind_arrow_keys()` / `radio_ui._on_arrow_key()`. These are ignored while typing in the Network page's password box.
- Mobile page V1.0.43.

2026-10-03 — Nixie tubes redrawn (Pi/web only, no reflash)

Re-sync PiDisplay/freq_fonts.py and web/, restart `radio_ui.py`, refresh browsers.

- **Freq Font > Nixie redrawn** to look like real Nixie tubes, from the user's reference picture:
 - a tall glass tube with dark end caps and a warm halo;
 - a red-amber honeycomb mesh inside;
 - thin wire digits glowing yellow-orange, drawn as smooth strokes (wide and dim through thin and bright) rather than a font;
 - a full tube of its own for the decimal point, with a glowing dot at the bottom.
- Same shapes on the Pi (`freq_fonts.py`, Tk canvas) and the web (`drawNixieTube()`, canvas glow and gradients).
- Mobile page V1.0.42.

2026-10-03 — Smooth 3D waterfall, DSP: on data line, Freq Font styles (Pi/web only, no

reflash)

Re-sync PiDisplay/ (new file `freq_fonts.py`), restart `radio_ui.py`, refresh browsers.

- **3D waterfall smoothed (Pi):** rows are now drawn as one image with a true colour gradient (Pillow), so the horizontal colour steps are gone. It looks like the web version, and it's faster (about 13 ms per frame vs about 20 ms with bands). It falls back to colour bands if Pillow (python3-pil) isn't installed.
- **"DSP:" (RX DSP gain, dB) moved to the data line** above the spectrum, between AGC: and Scale:, and removed from the settings pane (Pi and web). On the Pi the readouts are now spaced with equal gaps based on their widest values, so nothing overlaps with 9 items. On the phone the line wraps to two rows.
- **New Display > Freq Font:** Original / Nixie (orange digits in glass tubes with a faint yellow glow) / VFD (green seven-segment, unlit segments faintly visible) / Dot Matrix (blue 5x7 dots). Shared across the Pi and web remotes and remembered across restarts.
 - Pi: new `freq_fonts.py`, `set_freq_font()`, `_set_pi_freq_font()`.
 - Web: `drawFreqStyler()/renderFreq()` on a `#freqCanvas`.
 - Messages: `pi_freq_font_set` / `pi_freq_font`.
- **Web remote: the S-meter's "-XXX.X dBm" reading was cut off on the right.** `marginRight` changed from 70 to 100, the same fix the phone got on 2026-09-27.
- Mobile page V1.0.41.

2026-10-02 — Menu highlights + Network (Wi-Fi) page (Pi/web only, no reflash)

Re-sync PiDisplay/ (two new files: `wifi.py`, `network_screen.py`), restart `radio_ui.py`, refresh browsers.

- **Menus highlight the current / last-selected choice** wherever there are two or more options:
 - **Pi:** radio-style entries with a green dot and bold green text. That covers every toolbar dropdown (Band, Mode, Demod, Zoom, Noise, Course Inc, Fine Inc, Decode, Vol, Filter, Home), every Menu/Display group, and the Pi-only flyouts.
 - **Web:** a green check on Menu/Display items (`.submenu-item.selected`). The toolbar `<select>`s already pre-highlighted the current choice.
 - Driven by the radio's real state where it's known: STATUS for Band/Mode/Demod and the T41's current menu item, SETTINGS for Zoom/Noise/Inc/Decode/Vol. So changes made on the T41's front panel show too. The Pi-only settings (fill, 2D/3D, speed, brightness) follow their shared state, and brightness is now broadcast as well (`pi_brightness`).
- **New Menu > Network page on the Pi screen** (`network_screen.py` + `wifi.py`, using NetworkManager's `nmcli`):
 - Shows the current SSID, 2.4/5 GHz band, signal and IP address.
 - Lists the visible networks: signal, band, LOCK/open, connected/saved.
 - Rescan, Connect, Forget and Close buttons.
 - A secured, unsaved network opens a password box with an on-screen keyboard (one-shot Shift, ?123 symbols, Show/Hide, Clear, backspace).
 - Scans and connects run on a worker thread, so the radio display never freezes. A failed new connection's half-made profile is removed.
 - Pi only: the web remotes don't get it, because switching networks from a browser on the current one would cut it off.
 - If Connect says "Not authorized", there's a one-time polkit rule in `PiDisplay/README.md` ("Network screen").
- Mobile page V1.0.40.

2026-10-02 — Toolbar dropdowns moved left (Pi/web only, no reflash)

- Pi: the two toolbar rows start 8 toolbar-font spaces further left (`toolbar_left_x = x0 + 340 - width of 8 spaces`, about 32px). A 2m frequency like "144.300.000" still clears it.
- Desktop web remote: `#freqBlock` min-width 300px to 270px, so the toolbar sits about 8 spaces further left. The phone page is unchanged (its toolbar is a full-width grid).

2026-10-02 — Display gets its own dropdown (Pi/web only, no reflash)

Re-sync PiDisplay/, restart `radio_ui.py`, refresh browsers.

- **Display has its own toolbar dropdown**, between Menu and Vol, and is no longer a category under Menu. Its items are the same grouped list (Spectrum floor, scale, smoothing, decay, trace smoothing, Spectrum shadow, Spectrum fill, Brightness, PI Spectrum), now one level shallower. Picks still send Display's real primary index, so nothing changes on the Teensy.
- Pi: `_OWN_DROPDOWN_CATEGORIES = ("Display",)`, plus a toolbar entry "Display" after "Menu". Web: `OWN_DROPDOWN_CATEGORIES`, and `buildMenu(container, {label, id, only})` builds both dropdowns. The `#menuWrap/#menuBtn/#menuPanel` styles became classes, so the two share styling, and opening one closes the other.
- Phone: everything in the Display dropdown opens inline (accordion). Mobile V1.0.39.

2026-10-02 — Display menu regrouped; 3D peaks blended (Pi/web only, no reflash)

Re-sync PiDisplay/, restart `radio_ui.py`, refresh browsers.

- **Menu > Display regrouped** into sub-menus. The T41's own menu numbers are unchanged; this only changes how the menu is shown:
 - Spectrum floor > Auto spectrum floor / Manual spectrum floor / Spectrum floor

- Spectrum scale, Spectrum smoothing, Spectrum decay, Spectrum trace smoothing
- Spectrum shadow > On / Off
- Spectrum fill > On / Off
- Brightness > 25-100%
- Pi Spectrum > Pi Spectrum Fill (On/Off) / Pi Waterfall (2D/3D) / Pi Waterfall Speed Defined once as `_MENU_LAYOUT` (display.py) / `MENU_LAYOUT` (web); every number in it is the item's real `DisplayOptions[]` index. Greyed-out items still work as before.
- **3D waterfall peaks blended:** the Pi now uses 12 colour bands (was 6), sampled from the same colour ramp as the web gradient. Each new row is 1-2-1 smoothed across neighbouring columns (valleys between nearby peaks fill in; peaks keep their height) and mixed 35% with the previous row (`_WF3D_TIME_BLEND`), so ridges look continuous. Row outlines are now dark navy instead of bright blue. The Pi takes about 20 ms per frame in testing.
- Phone page: deeper menu levels open inline (accordion) so they fit the screen. Mobile V1.0.38.

2026-10-02 — Pi remembers its display settings (Pi only, no reflash)

Re-sync `PiDisplay/radio_ui.py` and restart it.

- The Pi now **remembers its own display settings across restarts:** Pi Waterfall (2D/3D), Pi Waterfall Speed, Pi Spectrum Fill and Brightness. They're saved to `~/config/mustang2/pi_prefs.json` (written atomically) whenever one changes, from the Pi menu or any web remote, and re-applied when `radio_ui.py` starts. Web tabs pick them up on login. Delete the file to return to the defaults.
- `radio_ui.py`: `_load_pi_prefs()`, `_save_pi_prefs()`, `_apply_saved_pi_prefs()`; the four setters save after applying. Only `radio_ui.py` changed.

2026-10-02 — Waterfall speed setting (Pi/web only, no reflash)

Pi-only: re-sync `PiDisplay/`, restart `radio_ui.py`, refresh browsers.

- New **Menu > Display > "PI Waterfall Speed"** flyout: **Fast** (one waterfall row per spectrum frame, about 15 rows/s, as before), **Medium** (1 per 2 frames), **Slow** (1 per 4) and **Very slow** (1 per 8). It applies to both the 2D and 3D waterfalls. The frames in between are peak-held into the row, so a short signal is never skipped, and the spectrum trace above stays real-time.
- Shared state across the Pi and every web tab, like Pi Waterfall / Pi Spectrum Fill. It starts at Fast. Pi: `set_waterfall_speed()`, `_set_pi_waterfall_speed()`, web command `pi_waterfall_speed_set {divisor}`, broadcast `pi_waterfall_speed`.
- Mobile page V1.0.37.

2026-10-02 — 3D waterfall option (Pi/web only, no reflash)

Pi-only: re-sync `PiDisplay/` and restart `radio_ui.py`; refresh browsers. No Teensy reflash. The Teensy firmware stays at V1.0.28.

- New **Menu > Display > "PI Waterfall"** flyout with **2D / 3D**. 3D shows the newest spectrum row in front, with older rows stepping back (narrower and higher) and darkening until they fade out at the back. Blue foothills rise to yellow/orange/red peaks, and there's a dashed centre marker. Each row's noise floor is subtracted so signals stand up from a flat baseline.
- Shared state, like "Pi Spectrum Fill": picking it on the Pi or any web tab switches all of them, and a newly opened tab starts in the current mode. The default is 2D. The 2D waterfall keeps scrolling underneath, so switching back shows current history.
- Pi: `display.py _build_waterfall_3d()/_redraw_waterfall_3d()` (20 rows x 128 columns x 6 colour bands; about 12 ms per frame in testing), `radio_ui.py _set_pi_waterfall_3d()` + web command `pi_waterfall_3d_set`, `web_relay.py broadcast_pi_waterfall_3d()`. Web: `drawWaterfall3D()` (24 rows x 256 columns, gradient fill). Mobile page V1.0.36.

V1.0.28 (2026-10-01) — Menu picks jump straight to the item

Requires reflashing the Teensy together with re-syncing `PiDisplay/`.

- **New uplink `PILINK_MSG_MENU_SELECT (0x14)`**, payload `[primary, secondary]`: jumps straight to a Menu item via new `JumpToMenuItem()` (`MainBoard_DisplayMenus.cpp`). It dispatches the same `UISm` events the buttons do (`MENU`, `SELECT`, `SELECT`) after setting `primaryMenuIndex/ secondaryMenuIndex`, so variable items open their adjust screen and action items run exactly as if picked by hand. All three events happen before the next redraw, so no menu list is drawn on the way. It only acts from the Home screen and is refused while transmitting or for an out-of-range index.
- Fixes the Pi/web Menu dropdown visibly scrolling through every item in between (e.g. EEPROM to Display > Spectrum Floor). Same complaint the V1.0.16 Zoom fix answered.
- VERSION "Mustang 2 V1.0.28".
- **Pi (`radio_ui.py`, `pilink_client.py`):** `_select_menu_item()` now sends `HOME_SCREEN` then `send_menu_select()` when the Teensy reports V1.0.28 or later (parsed from the Settings VERSION string); on older firmware it falls back to the old tick-by-tick path. Covers every Menu item from the Pi screen, the web remote and the phone.

V1.0.27 (2026-10-01) — Equalizer "Exit" + Edit RX/TX buttons; "Load:" reading fixed

Requires reflashing the Teensy together with re-syncing `PiDisplay/`.

- **Audio Equalizer "Exit" (no save):** button 14 on the equalizer screen now restores both equalizers to the levels they had when the screen was entered and returns Home without saving (`RestoreEqualizerSnapshot()`, snapshot taken on the screen's first draw in

MainBoard_DisplayEqualizer.cpp; new case 14 in Loop.cpp's `UISm_StateId_EQUALIZER` button switch). Home still saves and exits. The T41's own Instructions pane lists the new button.

- **“Load:” now measures DSP work only:** `ReceiveProcessing()` (DSP.cpp) adds the block time to `elapsed_micros_sum` *before* `PlayBuffer()` instead of after. `AudioPlayQueue::playBuffer()` waits while the output queue is full, so Load: was counting that wait, and the wait depends on the queue alignment set at power-up. That's the likely reason Load: read 60-79% at power-on and fell to ~20% as soon as DATA mode (176.4 kbps) restarted the audio chain. Expect a steady, lower reading in every mode. No change to the audio itself.
- VERSION “Mustang 2 V1.0.27”.
- **Pi + web remotes:** equalizer controls now have “Edit RX” / “Edit TX” (direct select, lit green on the side being edited) in place of the single “Editing: (switch)” toggle, and “Exit” (button 14) next to “Save & Exit”. Mobile page V1.0.35.

V1.0.26 (2026-10-01) — Audio Equalizer screen ported to Pi/web; “TI:” renamed “CT:”

Requires reflashing the Teensy together with re-syncing PiDisplay/. New `PILINK_MSG_EQUALIZER` (0x09) telemetry message feeds a new Audio Equalizer screen (Menu > Audio Options > “Adjust Equalizers”) on the Pi's own display and both web remotes, at the user's request to “mimic the Audio Equalizers screen and function”. Mirrors the Teensy's own `DrawEqualizerAdjustment()`: Receive and Transmit panes of 14 bars (0-100), the edited pane outlined green with its selected cell green and the rest blue, the other pane grey, plus the “Incr.” step. Full remote control through the same primitives the physical panel uses - no new uplink message: button 15 (BEARING) toggles RX/TX editing, button 16 (SPARE) cycles the 1/10 step, the Volume encoder picks the cell, the Filter encoder changes its level, Home saves both equalizers and returns Home. Touch additions on the Pi and web: tap a bar to select it (paced Volume ticks, the short way round), tap the other pane to switch RX/TX, band centre frequencies labelled under the bars, a “1 kHz: 50”-style readout, and the mouse wheel/arrow keys adjust the selected cell while the screen is up.

Also (no reflash needed): the “New Line 2” readout strip's **“TI:”** label is now **“CT:”** (Course Tune increment) on the Pi and both web remotes, at the user's request. Mobile page bumped to Mustang Mobile V1.0.34.

Full detail in `CHANGELOG_firmware.md`, `CHANGELOG_pidisplay.md`, `CHANGELOG_remote.md` and `CHANGELOG_remote_mobile.md`.

V1.0.25 (2026-10-01) — I2C BIT test ported to Pi/web; several menu items greyed out

Requires reflashing the Teensy. New `PILINK_MSG_BIT_RESULTS` telemetry message feeds a new I2C BIT (Built-In Test) results screen (Menu > Diagnostics > “I2C BIT test”) ported to the Pi's own display and both web remotes, at the user's request (“the I2C BIT Test needs to be ported over the the PI and both remotes”). Mirrors the Teensy's own `DrawBIT()` - PASS/FAIL per board (front panel, BPF, RF, RF SI5351, K9HZ LPF), plus a “Home” button (the only control that screen's own `Loop.cpp` switch handles). Same “no new uplink message needed” pattern as the Receive IQ screen - entering/exiting routes through the existing generic menu-select/ virtual-button messages.

Also, at the same request, several menu items that have no Pi/web screen or action behind them yet are now **greyed out and non-selectable** on the Pi and both web remotes (front panel unaffected, still fully selectable): Calibration > S meter level/Frequency/Transmit IQ/Power (only Receive IQ stays enabled), Diagnostics > Restore receive EQ/Buffer print, and EEPROM > Print Data to Serial. Full detail in `CHANGELOG_firmware.md` (Teensy side) and `CHANGELOG_pidisplay.md` (Pi/web side).

Pi-only follow-up (no reflash): the new I2C BIT test screen's title/ rows were rendering too close to the top on the real Pi touchscreen, hidden under the two-row toolbar - moved down with more clearance. Both web remotes were unaffected. Full detail in `CHANGELOG_pidisplay.md`.

Pi-only follow-up #2 (no reflash): moving the content down wasn't enough - the toolbar is still always on top of canvas-drawn content regardless of position, so it kept jamming into the BIT screen. The toolbar (Menu/Vol/Band/.../Home dropdown row) is now hidden outright while the I2C BIT test screen is showing, since that screen already has its own Home button. The Receive IQ calibration screen's toolbar is now hidden the same way for consistency. Full detail in `CHANGELOG_pidisplay.md`.

Pi-only follow-up #3 (no reflash): with the toolbar hidden, the title and Home button were correct, but all 5 board-status rows were piled on top of each other at the top-left instead of stacking down the screen - a y/x unpacking bug in `_build_bit_pane()`'s row-position helper (grabbed the X component instead of the Y component for every row, so every row landed at y=0). Fixed. Full detail in `CHANGELOG_pidisplay.md`.

V1.0.24 (2026-09-30) — Receive IQ calibration screen: layout fix + Stop/Exit

Requires reflashing the Teensy. Follow-up to V1.0.23 below, after the user tried it: Pi-only layout fix (Delta readout moved to the left edge and down a line; spectrum/Adjust/Table panes moved up so the GEN row in the all-bands table no longer gets cut off at the bottom), plus two new buttons, “Stop” (aborts a runaway-looking Auto sweep early - a full sweep of all bands is bounded but can take a couple of minutes, which looked like a hang on GEN, the last band) and “Exit” (leaves without saving, distinct from the existing “Save & Exit”). Both new buttons are repurposed button ids scoped only to this screen's own `Loop.cpp` switch. Scoped to the Pi's own screen only, not the web remotes. Full detail in `CHANGELOG_firmware.md` (Teensy side) and `CHANGELOG_pidisplay.md` (Pi side).

V1.0.23 (2026-09-30) — Receive IQ calibration screen added

Requires reflashing the Teensy. New `PILINK_MSG_RXIQ_CAL` telemetry message feeds a new Receive IQ calibration screen (Menu > Calibration > Receive IQ) added to the Pi's own display and both web remotes, at the user's request (“Need to add the Calibration Screen from T41 to the PI” → “Yes the PI needs tpo control calibrataion the same as the T41 would”). Full remote control (Amp/Phase adjust, step-size toggle, auto-tune, band cycling, Save & Exit) routes through the existing generic virtual-button/ virtual-encoder messages — no new

uplink protocol needed. Web remotes deliberately skip the live two-sideband spectrum mini-plot the T41/Pi screen shows. Frequency/Transmit IQ/Power calibration screens are deferred to future passes. Full detail in `CHANGELOG_firmware.md` (Teensy side) and `CHANGELOG_pidisplay.md` (Pi/web side).

Unreleased (no reflash needed)

Rebuilt 2026-09-30 after this section (and the V1.0.16 heading below) went missing from this file outside of any edit made here — the Pi-side changelogs (`CHANGELOG_pidisplay.md`, `CHANGELOG_remote.md`, `CHANGELOG_remote_mobile.md`) stayed intact throughout and are the source for everything below; see those for full detail.

Fixed

- **Receive IQ calibration screen (Pi only): frequency-ruler bleed-through, cramped instructions, GEN row crowding its own pane border** - follow-up to V1.0.24's layout fix above, after the user tried it again. The Home spectrum pane's frequency-tick ruler was redrawing itself on top of the calibration screen's Adjust/Table panes (freshly-recreated canvas items always render above an already-raised backdrop); now suppressed while that screen is active. Instructions font 11pt → 14pt. Adjust/Table panes grown 230px → 255px (added to the bottom, kept equal) - the real cause of "GEN cutoff" was the Table box's own border, not the screen edge. Pi-only, `display.py`. Full detail in `CHANGELOG_pidisplay.md`.

Added

- **Both web remotes gained the SWR/PWR readout**, which had never been built on either page before — the data was already flowing over the WebSocket. No reflash needed. Full detail in `CHANGELOG_pidisplay.md`.
- **Independent "PI Spectrum Fill" toggle** (Menu > Display > "PI Spectrum Fill", On/Off) for the Pi's own screen and both web remotes, separate from the T41's own Menu > Display > "Spectrum fill on/off" setting — supersedes V1.0.22's mirroring below, at the user's follow-up request to tune each independently ("T41 On/Off and PI/Remotes On/Off"). No reflash needed. Full detail in `CHANGELOG_pidisplay.md`.

Fixed

- **Both web remotes dropped the "(USB)" demod suffix** — `MODULATION_NAMES` had `0: null` instead of `0: "USB"` on both pages, present since each was first built. Fixed. Full detail in `CHANGELOG_pidisplay.md`.
- **Choppy audio on both web remotes** — the real cause was a mis-clocked playback rate (`onAudioPcm()` labeled each chunk with the browser's `AudioContext` sample rate instead of the PCM's true, fixed 44.1kHz capture rate), which drifted the playback schedule out of sync with real chunk arrival and forced periodic audible resyncs. An initial fix attempt (raising the jitter buffer, `AUDIO_LEAD_S`, 0.15s → 0.4s) made it worse, which is what pointed at the drift bug instead of a jitter one — reverted once the real bug was found and fixed. **Web remotes only.**
- **Still cutting out after the above** — the sample-rate fix didn't resolve it ("No change that I can hear... cutting out very rapidly"), which pointed further upstream, past the browser entirely, to the Pi's own audio capture. `AudioCapture` (`audio_relay.py`) now opens its `PortAudio` stream with `latency="high"` (a bigger, more xrun-resistant capture buffer — the Tkinter GUI sharing this process's GIL is the suspected culprit) and logs `status` flags so a next occurrence confirms or rules this out. **Requires restarting `radio_ui.py`** — not a browser refresh.
- **No xruns, still cutting out** — confirmed capture-side is clean, pointing at delivery instead (the asyncio broadcast thread sharing this process's GIL with the Tkinter GUI, or something further downstream). Added timing diagnostics to `web_relay.py`: a gap counter on the callback-arrival side and a queue-delay counter on the send side, both logged past 80ms. **Requires restarting `radio_ui.py`.**

Changed

- **Pi's Scale: readout nudged 2 character-spaces left**, at the user's request. **Pi's own screen only.**
- **Spectrum trace box shrunk and New Line 2 strip moved above it** — at the user's request ("reduce the spectrum's pane height, there is a lot of wasted space... move the readouts that are currently in the spectrum display to above the top line"). Trace box height cut (Pi: 150/345 of the pane → 90/323; web: 40% of remaining canvas height → 25%), freeing more room for the waterfall. The Vol:/AGC:/Scale:/Floor:/Zoom:/TL:/FT:/ Noise: strip is no longer drawn overlapping the trace box — it's now a separate row above the spectrum pane/canvas on all three surfaces (Pi's own `_build_newline2_pane()`; both web pages' new `#newline2Row` DOM strip, `updateNewLine2()`, replacing the old canvas-drawn `drawNewLine2()`).
- **Pi's New Line 2 strip switched to plain even column spacing**, at the user's explicit request — replaces the 3 content-weighted passes with simple 1/8 increments. **Pi's own screen only** — `remote.html/remote_mobile.html` keep the content-weighted fractions.
- **Mobile page's New Line 2 strip font reduced 14px → 10px**, at the user's request. **Phone page only** — `remote.html` unchanged.
- **Menu > Display > Brightness combined into one flyout** — the 4 flat "Brightness 25%"/"50%"/"75%"/"100%" entries became a single "Brightness" entry with its own nested percentage flyout, at the user's follow-up request. Still entirely Pi-side, no Teensy involved.
- **"Scale:" moved right, centered between AGC: and Floor:** on the New Line 2 strip, with more room budgeted for its full "Scale: XX.XdB" width — at the user's follow-up request, after the overlap fix below left it crowded against AGC: with unused space before Floor:.

Added

- **Menu > Display > Brightness** — a Pi-only control (`display.py`'s `T41_MENU`, `remote.html/remote_mobile.html`'s `T41_MENU`) for the Raspberry Pi Touch Display 2's own screen backlight, with no Teensy firmware counterpart at all — intercepted in `radio_ui.py`'s `_select_menu_item()` before any real menu navigation is sent. Four presets (25/50/75/100%), at the user's request.

Fixed

- **“Scale:”/“Floor:” overlap on the New Line 2 strip** — Scale:’s widest value (“12.5dB”) was overwriting part of Floor:’s label. Column fractions recomputed by actual character-count budget instead of even spacing, across `display.py/remote.html/remote_mobile.html`.

Changed

- **Pi-screen layout pass** (`display.py`): the Pi’s LAN IP now shows in front of the clock (top-right corner) so the web-remote address is visible at a glance; SWR/PWR text moved next to REC/XMT (was left- anchored with a wide visible gap; now right-anchored, flush against it) — Pi’s own screen only; Audio Filter pane’s bottom tick labels enlarged 8pt → 11pt on the Pi and the desktop web remote (`remote.html`), phone page (`remote_mobile.html`) left as-is at the user’s confirmation it’s fine there. Full detail in `CHANGELOG_pidisplay.md`.
- **Spectrum/waterfall color pass** (Pi’s own screen + both web remotes), at the user’s request that the T41’s own colors are “much brighter” than the Pi’s: spectrum trace/fill green brightened from a cyan-tinted `#00ff66` to pure `#00ff00` (matching the Teensy’s actual `SPECTRUM_FILL_COLOR/RA8875_GREEN`); the audio-filter passband highlight lightened from a dim navy (`#000080`) to `#3399ff`; “REC” on the Pi’s own screen changed from white to green (`#00cc44`, matching what the web remotes already did); and the green→yellow→red fill transition — reported as showing up “as a very distinct line” — now fades with a smoothstep ease over a slightly widened span instead of a hard-edged linear ramp. Full detail in `CHANGELOG_pidisplay.md`.

V1.0.22 (2026-09-30) — current

Added

- **Pi’s own spectrum pane and both web remotes now mirror Menu > Display > “Spectrum fill on/off”** instead of always drawing filled — a cosmetic follow-up to V1.0.21’s fix, at the user’s request, sent over a new PiLink field (`spectrumFillTraceEnabled`). Unrelated to the audio fix itself. **Requires reflashing the Teensy** and re-syncing/restarting `PiDisplay/`. Full detail in `CHANGELOG_firmware.md/` `CHANGELOG_pidisplay.md`.

Version

- `VERSION` (`SDT.h`) bumped to “Mustang 2 V1.0.22”.

V1.0.21 (2026-09-30)

Fixed

- **Choppy audio — root cause found, fixed, and confirmed on the radio.** The filled/color-graded spectrum trace could take long enough to draw on a strong/broadband signal that `loop()` didn’t get back to servicing the audio DSP chain in time, overflowing the continuously-running audio input — on the Teensy’s own local speaker output as much as the Pi/web remotes. User-confirmed on V1.0.21’s new toggle: fill ON = choppy everywhere with “Load:” a low 0-24%; fill OFF = clean everywhere with “Load:” a higher 60-67%. That’s not backwards — dropped blocks are never counted into “Load:” at all (only fully-processed ones are), so the choppy state’s low reading reflects blocks being silently dropped, not the DSP chain being idle; 60-67% is just this receiver’s true, normal DSP cost once nothing is being dropped. Full detail in `CHANGELOG_firmware.md`.

Added

- **Menu > Display > “Spectrum fill on”/“Spectrum fill off”** — the fix above is now a live setting (`ED.spectrumFillTraceEnabled`) instead of a compile-time flag, so the filled look and this failure mode can be traded off from the radio itself if it ever comes back on a wide enough signal, no reflash needed. Defaults to on. **Requires reflashing the Teensy.** Full detail in `CHANGELOG_firmware.md`.

Version

- `VERSION` (`SDT.h`) bumped to “Mustang 2 V1.0.21”.

V1.0.20 (2026-09-30) — TEST BUILD, diagnostic only

Changed

- **SPECTRUM_FILL_TRACE (MainBoard_DisplayHome.cpp) set to 0 (outline-only trace) as a diagnostic** — the choppy-audio report persisted after V1.0.19’s gain-cache fix, and DSP Gain / Noise Reduction / Auto-Notch were all ruled out by testing (no change either way), with “Load:” reading a steady 0-25% the whole time. But “Load:” only covers time inside `PerformSignalProcessing()`, not `DrawDisplay()`, which `loop()` calls right after it every pass on the same core. The filled/gradient spectrum-trace fill already scales with signal strength (per its own code comment); if a slow sweep delays `loop()` from getting back to the audio processing in time, the continuously-running audio input can overflow regardless of measured DSP load — on the Teensy’s own local output as much as anywhere downstream. This build isolates that variable to test the theory; it is **not a fix**, and the trace fill is meant to go back to 1 once the result is known either way. **Requires reflashing the Teensy.** Full detail in `CHANGELOG_firmware.md`.

Version

- `VERSION` (`SDT.h`) bumped to "Mustang 2 V1.0.20".

V1.0.19 (2026-09-30)

Changed

- `ApplyRFGain()` (`DSP.cpp`) **caches its linear gain instead of recomputing** `pow(10, dB/20)` **on every audio block** — found while chasing a “sound cutting out very rapidly” report traced all the way back to the Teensy’s own local audio (confirmed independent of USB, the Pi, and the web remotes — both a raw Pi-side recording and the Teensy’s own speaker were choppy). This read has been “live every audio block” since V1.0.6, so it isn’t a new cost — if choppiness persists after this reflash, the rest of the DSP chain is next to check. **Requires reflashing the Teensy.** Full detail in `CHANGELOG_firmware.md`.

Version

- `VERSION` (`SDT.h`) bumped to "Mustang 2 V1.0.19".

V1.0.18 (2026-09-30)

Added

- **“Floor:” reading added to the New Line 2 strip**, right after Scale:, at the user’s follow-up request (confirmed Menu > RF Options > RX DSP Gain fixed the weak-signal spectrum complaint once reflashed). New `PiLinkSettingsMsg` fields `spectrumFloor/spectrumFloorAuto` (54 → 57 bytes) carry `ED.spectrumNoiseFloor[currentBand]/ ED.spectrumFloorAuto[currentBand]`; the Pi/web remotes show “Auto” while Menu > Display > “Auto spectrum floor” is on, otherwise the signed manual value (a raw pixel shift, not dB). **Requires reflashing the Teensy together with re-syncing PiDisplay/** — same stale-pane caveat as V1.0.17’s `spectrumScale`.

Version

- `VERSION` (`SDT.h`) bumped to "Mustang 2 V1.0.18".

See `CHANGELOG_firmware.md`, `CHANGELOG_pidisplay.md`, `CHANGELOG_remote.md`, and `CHANGELOG_remote_mobile.md` for the full per-file writeups.

V1.0.17 (2026-09-30) — current

Added

- **RX DSP Gain menu entry enabled** (`RFSet[]`, `MainBoard_DisplayMenus.cpp`) — Menu > RF Options > RX DSP Gain, -5.0 to 20.0 dB in 0.5 dB steps, bound directly to `ED.rfGainAllBands_dB`. At the user’s report “I don’t see a way to change RF gain on any menu options” — the variable was already live in the RX chain (`ApplyRFGain()`, `DSP.cpp`) and already shown read-only on the Settings pane as “DSP Gain:”, it just had no menu entry to change it from (commented out as `"[_RX_DSP_Gain]"`). No post-update callback needed — the DSP chain reads the value live, unlike RX Attenuation’s separate hardware write.
- `PiLinkSettingsMsg.spectrumScale` (new field, struct grew 53 → 54 bytes) — the current Menu > Display > “Spectrum scale” index, feeding a new “Scale:” reading in the Pi/web remotes’ New Line 2 strip, right after AGC:. Added while diagnosing a weak-signal-barely-visible spectrum report. **Requires reflashing the Teensy together with re-syncing PiDisplay/** — `pilink_client.py`’s SETTINGS frame-size check will fail on an un-reflashed Teensy, and the *entire* Settings pane/New Line 2 strip goes stale (not just this field) until both sides match. Full detail in `CHANGELOG_pidisplay.md/CHANGELOG_remote.md/ CHANGELOG_remote_mobile.md`.

V1.0.16 (2026-09-27)

Added

- **PILINK_MSG_VIRTUAL_BUTTON_REPEAT (0x13)** — a new uplink message (`[button_id, count]`) that queues N back-to-back presses of the same button on the Teensy in one shot, instead of the Pi pacing them 300ms apart. Fixes the user-reported “why does it have to go through all the levels” complaint for Zoom/Band/Mode/Demod/Noise/Course Inc/Fine Inc/Vol/ BW selection — cycling multiple steps now happens near-instantly instead of visibly stepping through each intermediate level. Doesn’t add a reverse direction to any button (Zoom etc. still have no “down” counterpart in `Loop.cpp`), it just removes the pacing between repeats of the *same* button id. **Requires a Teensy reflash together with re-syncing PiDisplay/** — `radio_ui.py` no longer falls back to the old one-press-at-a-time path. Full detail in `CHANGELOG_firmware.md/ CHANGELOG_pidisplay.md`.

V1.0.15 (2026-09-26)

Fixed

- **Menu-index telemetry (see V1.0.14’s “Remote Menu selection... landed back on Home” entry for the full writeup) - version bump only, so the Teensy’s own screen and the Pi’s Settings pane have a visible marker that this reflash actually took.** `PiLinkStatusMsg` gained `menuPrimaryIndex/menuSecondaryIndex`; `radio_ui.py` now reads the Menu’s real position from

telemetry on every STATUS frame instead of guessing/shadowing it, closing out the drift bug for good. **Requires reflashing the Teensy** - confirm you're on this version by checking the version string, not just the button that used to be broken.

V1.0.14 (2026-09-25)

Added

- **LAN-only browser remote control (mouse + USB rotary encoder), no reflash of the touchscreen behavior needed beyond the new tuning encoders below.** A new `PiDisplay/web_relay.py` runs a small WebSocket server on the Pi (`config.WEB_RELAY_PORT`, default 8765) that re-broadcasts the exact same telemetry `radio_ui.py` already parses from PiLink (STATUS/SETTINGS/ SMETER/SPECTRUM) to any connected browser, and forwards browser button/ encoder commands to the SAME `_select_*/()/send_virtual_button()/ send_virtual_encoder()` code path a touch on the Pi's own screen uses - never a separately-derived copy of the press-counting logic. Open `http://<pi-hostname>:8765/` from any browser on the same network (served directly by `web_relay.py` - no separate web server needed) for a page mirroring the Pi's VFO/toolbar/band-mode/TX-RX/S-meter/spectrum+waterfall+ frequency-scale panes and the Menu parameter-adjustment overlay, mouse-only (no touch handling). Gated by a passkey (`config.WEB_PASSKEY`, default "123" - change it). **LAN-only by design:** plain `ws://`, no TLS - do not forward this port through your router without adding both.
- **PILINK_ENCODER_TUNE / PILINK_ENCODER_FINETUNE** (firmware): the main VFO tuning knob and the fine-tuning knob now have virtual-encoder ids, wired to the same `iCENTERTUNE_INCREASE/DECREASE/iFINETUNE_INCREASE/DECREASE` events their own physical ISRs post - previously PiLink only exposed Filter/Volume remotely, so nothing (Pi touchscreen or browser) could move the VFO at all. The web remote page drives these directly from a mouse wheel over the frequency/spectrum area (Shift = fine) or from a USB rotary encoder configured to emit arrow/page keys, regardless of mouse position. **Requires reflashing the Teensy.**
- **Web remote: Audio Spectrum pane, Settings pane, and a real cascading Menu** (Pi-only, no reflash - just re-sync `PiDisplay/` and restart `radio_ui.py/reload` the page). The remote page now also mirrors the Audio Spectrum pane (live magenta audio-FFT bars plus the white passband- edge/yellow CW-filter markers and 1kHz ruler) and the Settings pane (Vol/ AGC/Tune Inc/FT Inc/RX+TX Atten/AutoNotch/Noise/DSP Gain/Antenna/Zoom/ Rate/Temp/Load/version) - all from telemetry that was already flowing to the browser but not yet drawn. The Menu control was also rebuilt: it was a flat grouped `<select>` (read, fairly enough, as a "table of contents" rather than a menu), now replaced with a nested/cascading popup - click "Menu", hover or click a category, its options fly out to the side - much closer to the Pi's own Tkinter `Menu` button + cascade widget. Still not covered: SWR and State of Health.
- **Web remote: BW (Filter) dropdown, wheel-target fix, and a wheel deselect control** (Pi-only, no reflash - re-sync `PiDisplay/` and restart `radio_ui.py/reload` the page). A new "BW" dropdown sits behind Zoom in the toolbar, listing "Hi Cut"/"Lo Cut" and driving `FILTER_BUTTON` (`Config.h` id 19, which toggles `changeFilterHiCut` in `Loop.cpp`) the same way the existing Vol dropdown drives `VOLUME_BUTTON` - no firmware change needed, since `PILINK_MSG_VIRTUAL_BUTTON` already forwards any button id unchanged. Fixed a bug where the mouse wheel was hardcoded to always emulate Volume outside the spectrum area/parameter overlay, with no awareness of BW at all; picking from either the Vol or BW dropdown now "claims" the wheel for that knob (`_wheel_target_encoder` in `radio_ui.py`, `state.wheelTarget` in `remote.html`), and a claimed wheel target now takes priority even while hovering the spectrum/tune area. Added an explicit "Wheel: Tune only / Volume / Filter" indicator on both the Pi's own screen (a clickable label next to the other toolbar controls) and the web page (a clickable status pill) - clicking it releases ("deselects") the wheel back to tuning-only, so turning the wheel afterwards no longer accidentally changes Volume or Filter. Also fixed the BW dropdown's Hi Cut/Lo Cut label order, which had been reversed relative to the real hardware (a Teensy-side `changeFilterHiCut` shadow-state drift issue across restarts, not a code bug - resolved by power-cycling the Teensy so both sides start from a known default).
- **Live RX audio on the web remote, plus a Mute button** (Pi-only, no reflash - re-sync `PiDisplay/` and restart `radio_ui.py/reload` the page; DOES need the Teensy's own native USB port wired to a Pi USB-A port, and the sketch built with Arduino IDE's "Tools > USB Type > Serial"
 - MIDI + Audio" - the same USB-audio path already documented for FT8/ WSJT-X. New `PiDisplay/audio_relay.py` captures 44.1kHz mono PCM from that USB audio device (via `sounddevice/PortAudio` - new dependency, see `requirements.txt`) and hands it to `web_relay.py`, which streams it to every logged-in browser as raw binary WebSocket frames (uncompressed - LAN-only, ~86KB/s, trivial bandwidth, no codec to install on either end). `remote.html` plays it back by scheduling each chunk as its own `AudioBuffer` back-to-back (no `ScriptProcessorNode/AudioWorklet` needed). A new "Mute" button sits with the other toolbar dropdowns and turns red while muted; it's entirely local to that browser tab - it does not stop the Pi from sending audio, and doesn't affect the Pi's own speaker or other viewers. If the USB-audio cable isn't connected yet, or `sounddevice/PortAudio` isn't installed, `audio_relay.py` logs one line and disables itself - everything else in the app is unaffected. Config: `config.AUDIO_ENABLED/AUDIO_DEVICE_NAME_HINT/AUDIO_SAMPLE_RATE/ AUDIO_CHANNELS/AUDIO_CHUNK_MS`.
- **Blue filter-bandwidth highlight on the Spectrum pane, on both the Pi's screen and the web remote** (Pi-only, no reflash - re-sync `PiDisplay/` and restart `radio_ui.py/reload` the page). Ports `StampTuningBar()`'s filter-bandwidth highlight box (`MainBoard_DisplayHome.cpp`, `FILTER_WIN = 0x10`, a dim navy blue) onto this pane's own trace box - a solid rect spanning the current receive passband, centered on the same fixed screen-center tuning marker the cyan dashed line already showed. Reuses the SAME `passband_low_hz/passband_high_hz` STATUS fields already flowing for the Audio Spectrum pane's white markers - no new PiLink field needed, and both panes stay in lockstep since they read the same telemetry. Drawn/layered so the live trace paints over it (behind the graded fill bands on the Pi's Tkinter canvas; drawn first, before `drawTrace()`, on the web page's immediate-mode canvas) - it only shows through in the empty space above the curve, the same visual effect as the real hardware. `display.py's set_spectrum_filter_bar()` / `remote.html's drawFilterBar()`.
- **Desktop icons to start `radio_ui.py` and shut down the Pi** (Pi-only, no reflash - new files under `PiDisplay/desktop/`, one-time setup needed, see `desktop/README.md`). "Start Radio UI" opens a terminal and runs the same `DISPLAY=:0 python3 radio_ui.py` command used by hand, refusing to start a second copy if one's already running (it holds the PiLink UART port exclusively). "Shutdown Pi" asks for confirmation (a `zenity` dialog if installed, else a terminal prompt) before running `sudo shutdown -h now` - needs a one-time sudoers rule scoped to ONLY the `shutdown` command for user `bob`, not blanket passwordless `sudo` (see the README for the exact `visudo` line). Both are real `.desktop` launcher files meant to be copied onto the Pi's actual Desktop, not run from inside `~/PiDisplay` itself.

Fixed

- **A live-hardware regression, introduced and reverted the same day: do NOT trust PiLink.h/PiLink.cpp as checked in on the Mac for the exact PiLinkSettingsMsg wire size.** First diagnosed (wrongly) as `pilink_client.py`'s `SETTINGS_FMT` having two phantom trailing fields (`audioVolume`, `secondScreenActive`) not present in the Mac's `PiLink.h` - that struct is only 17 fields/51 bytes there, and `PiLink.cpp`'s `SendSettings()` sends exactly `sizeof(msg)`, so the reasoning from source alone was sound. Removing those two fields from `SETTINGS_FMT` was pushed as a fix - but a live diagnostic against the real Teensy (a one-time size-mismatch print added to `pilink_client.py`'s `dispatch()`) showed the ACTUAL hardware sends 53 bytes, matching the ORIGINAL (pre-"fix") 19-field format exactly. Reverted `SETTINGS_FMT` back to "`<BhBIihBhBhbBhB24sBBB`" (53 bytes) - the real, currently-flashed firmware genuinely does send `audioVolume/secondScreenActive`; it's the Mac's checked-in `PiLink.h/PiLink.cpp` that are the stale copy, presumably from the same class of silent-sync bug that hit `web/remote.html` and this same file three separate times this session, just from an earlier one. **Open item, not yet fixed:** those firmware source files on the Mac still need reconciling against whatever's actually flashed (need to find/recover the real `PiLink.h/.cpp` that produces a 53-byte `PiLinkSettingsMsg`, or hand-add the two fields back with a real guess at their types/meaning and confirm against a reflash) - deferred until that's actually needed (e.g. before any future firmware-level change touching this struct), since the Pi-side fix above doesn't require it. The temporary one-time diagnostic print in `pilink_client.py`'s `dispatch()` (both the mismatch and the "frame size OK" cases) is left in for now in case this surfaces again. **Resolved (checked 2026-10-01, V1.0.25):** `PiLink.h/PiLink.cpp` now match `pilink_client.py` field for field (`SETTINGS` is 58 bytes after the later V1.0.17-V1.0.22 additions, including `audioVolume/secondScreenActive`); no source change needed.
- **Remote Menu selection (Pi and web both) landed back on Home instead of sticking on the picked setting's adjustment screen** - confirmed 2026-09-26 with "Menu → Display → Spectrum floor" (and RF Options → SSB Power, Display → Spectrum scale): the same items worked correctly from the front panel. Root cause: `radio_ui.py` tracks the Teensy's `primaryMenuIndex/secondaryMenuIndex` with its own shadow copy (`_menu_primary_shadow/_menu_secondary_shadow`), since `PiLink` has no telemetry field for them - a limitation flagged in the code's own comment since it was written, "worth confirming on real hardware." It drifts out of sync the moment anything else moves the real menu position (the front panel, or `radio_ui.py` itself being restarted while the Teensy keeps running - both routine this session). A desynced shadow sends the next remote click's Filter-encoder ticks to the wrong category/option; landing on a `functionOption` by accident looks exactly like this bug, since the Teensy correctly (by design) flashes and auto-returns Home for those. Fixed by resyncing the shadow from ground truth on every `PARAMETER` frame (`_handle_parameter()` in `radio_ui.py`): the Teensy's own `SendParameter()` (`PiLink.cpp`) already sends the real option's label text whenever `uiSM` is actually in `UPDATE` state, however it got there, so a label → (`primary_idx, secondary_idx`) lookup (`_MENU_LABEL_TO_INDEX`, built from `_T41_MENU`) corrects the shadow every time - no new `PiLink` message needed. Doesn't cover drift from browsing `MAIN_MENU/SECONDARY_MENU` or landing on a `functionOption` without passing through a `variableOption` first (no telemetry for those either), a narrower gap than the one this fixes. **Confirmed 2026-09-26, after a re-sync and `radio_ui.py` restart, that this shadow-resync fix alone did NOT resolve the reported symptom** - the shadow-desync bug above is real and worth fixing on its own merits, but it isn't the (or isn't the whole) explanation. Re-reading the Teensy firmware's button path (`FrontPanel.cpp`, `PiLink.cpp`, `Loop.cpp`) turned up a second, more likely root cause: `FrontPanel.cpp`'s `ButtonPressed` is a **single-slot mailbox** (`static int32_t ButtonPressed = -1;`), not a queue, yet every virtual button send also queues a *separate*, generic `iBUTTON_PRESSED` marker into a real FIFO (`interruptFifo`, `Loop.cpp`). The button ID and "a button was pressed" notice travel by two different mechanisms that aren't kept atomically in step. If the Teensy's main loop - busy with real-time SSB audio/DSP plus ~15fps spectrum FFT - falls behind the gap between two button sends, a second `SetButton()` call can overwrite the mailbox before the main loop has drained the first press's FIFO marker, so both FIFO entries end up processed against the SAME (latest, wrong) button ID. This corrupts whichever bracketing press (`HOME_SCREEN/MAIN_MENU_UP/ MENU_OPTION_SELECT`) got clobbered, landing back on Home - while Filter-encoder ticks are unaffected, since their value rides directly in the FIFO entry rather than through the shared mailbox, matching exactly what was seen: category/option ticks flicker through correctly, but the bracketing Select never sticks. Raised `PRESS_SPACING_MS` in `radio_ui.py` from 120 to 300 to give the Teensy's main loop more headroom - **confirmed 2026-09-26 this made no difference either** (ruled out as the real cause: if a load-dependent race were the explanation, the single-button-repeat controls - Zoom/Band/Mode/Demod/Noise/Vol/BW, which use the exact same send path - would misbehave too, and they never have; only Menu's multi-different-button sequence ever failed). **Actual root cause, confirmed 2026-09-26:** the shadow-desync bug described above was real, and worse than "narrow" - it was the whole story. `_menu_primary_shadow/_menu_secondary_shadow` started this session assumed at (0, 0) and, because the shadow only self-corrects *after* a selection successfully reaches a `variableOption`'s screen, a badly-drifted shadow can never repair itself - every attempt keeps computing tick counts from the wrong starting point, landing on the wrong item, never landing on a `variableOption`, so the fix never gets a chance to run. A live front-panel check (press Menu, then Select) showed the Teensy was actually sitting on Diagnostics > Buffer print (indices 7, 2), nowhere near the assumed (0, 0). Hand-recalibrating the shadow's starting values to (7, 2) in `radio_ui.py` **immediately fixed remote Menu selection on both the Pi and the web remote**, confirmed against real hardware with Menu → Display → Spectrum floor. The hand recalibration above was a one-time snapshot, not a structural fix - the shadow could drift the same way again (front panel use, or a `radio_ui.py` restart while the Teensy keeps running). `PRESS_SPACING_MS` is left at 300 (harmless, just extra headroom) even though it wasn't the actual fix.
- **Permanent fix for the above, same day: `primaryMenuIndex/secondaryMenuIndex` (`MainBoard_DisplayMenus.cpp`) added to `PiLink` telemetry, so `radio_ui.py` never has to guess/shadow the Menu's position again. Requires reflashing the Teensy.** `PiLinkStatusMsg` (`PiLink.h`) gained two `uint8_t` fields, `menuPrimaryIndex/menuSecondaryIndex`, populated every `SendStatus()` call (`PiLink.cpp`) - these are plain persistent globals on the Teensy (not reset on entering the Menu, not scoped to any particular `uiSM` state), so sending them continuously in the already-frequent `STATUS` frame (not gated to `UISm_StateId_UPDATE` the way `PILINK_MSG_PARAMETER` is) means the Pi's copy is never more than one `STATUS` frame stale, in ANY screen state. `pilink_client.py`'s `STATUS_FMT` grew two trailing Bs to match; `radio_ui.py`'s `_handle_status()` now overwrites `_menu_primary_shadow/_menu_secondary_shadow` from real telemetry on every `STATUS` frame instead of only advancing them from its own sent clicks. The `PARAMETER`-label resync added earlier this session (`_handle_parameter()/_MENU_LABEL_TO_INDEX`) is left in place as a redundant second check - harmless, and would still catch it if `_T41_MENU` (`display.py`) and the firmware's `primaryMenu[]` table ever drift apart. This closes out the whole class of bug: there is no longer any guessed state for a remote Menu click to compute from.

V1.0.13 (2026-09-25)

Added

- **Frequency-scale ruler on the Pi's spectrum pane.** The spectrum/waterfall pane was missing the tick marks, frequency labels, and red out-of-band coloring the Teensy's own screen draws between the trace box and the waterfall (`DrawFrequencyBarValue()`, `MainBoard_DisplayHome.cpp`). `PILINK_MSG_STATUS` now carries `lowerFreqHz/upperFreqHz` (the visible spectrum span's edges - `GetLowerFreq_Hz()/GetUpperFreq_Hz()`), NOT the same as the already-sent `frequencyHz`, which differs by `ED.fineTuneFreq_Hz`) and `bandLowHz/bandHighHz` (the current band's legal limits, `bands[ED.currentBand].fBandLow_Hz/.fBandHigh_Hz`). The Pi reserves a 20px label strip (matching the Teensy's own fixed gap) and draws one tick per "nice" round-number step (targeting ~8 across the visible span, same algorithm as the Teensy), colored red when the tick falls outside the band's legal limits, green near the tuned center frequency, white otherwise, with tick marks always yellow - exactly mirroring `DrawFrequencyBarValue()`'s own coloring rule. **Requires reflashing the Teensy** (new `PiLinkStatusMsg` fields).

V1.0.12 (2026-09-25)

Added

- **Select on the Menu parameter-adjustment overlay** (Pi display only, no reflash needed): the red-boxed "Label: value" box (e.g. Display > Spectrum Floor) now has a tap-able "Select >" affordance that sends the real front panel's SELECT button (`MENU_OPTION_SELECT`) to confirm the value and return to Home - mirroring the physical action, since the wheel could change the value but nothing previously sent the confirming press short of digging into the unrelated Home dropdown.
- **Pi mirrors the Teensy's secondary ("Toggle Layout") home screen.** `ToggleHomeScreen()`/the SPARE button always correctly toggled the Teensy's own physical screen between its 11-pane primary and 5-pane secondary layouts, but the Pi had no equivalent and never visibly changed. `PILINK_MSG_SETTINGS` now carries `secondScreenActive` (plus `audioVolume`, always `ED.audioVolume` regardless of which function the volume knob currently multiplexes to), and the Pi switches its own display to match automatically - hiding State of Health/SWR/Audio Spectrum/Settings (no equivalent on that screen), showing a new "New Line 2" status strip (`Vol/AGC/Zoom/TI/FT/Noise`, mirroring `DrawNewLine2Pane()`), and moving the S-meter/TX-RX panes up into the freed space - all driven by telemetry, so it follows a screen switch made at the front panel too, not just one requested from the Pi's own Home dropdown. (Simplification: the Pi's spectrum/waterfall pane keeps its normal size in both modes, unlike the Teensy's own secondary screen, which additionally widens it.)

V1.0.11 (2026-09-25)

Fixed

- **Menu > Display changes ("Spectrum Scale", "Spectrum floor") weren't reaching the Pi's spectrum/waterfall pane.** `PiLink.cpp`'s `SendSpectrum()` was quantizing every bin against a fixed -140..-20dBm window, completely independent of `ED.spectrumScale[]/ED.spectrumNoiseFloor[]` - so adjusting either setting rescaled/shifted the Teensy's own onscreen trace but never touched what got sent to the Pi. `SendSpectrum()` now reuses the Teensy's own `pixelnew()` (the exact per-bin calculation `ShowSpectrum()` draws the real trace from - already reads spectrum scale/smoothing live) plus the noise-floor shift, normalized against the same on-screen box height (`SPECTRUM_HEIGHT`, exposed via a new `GetSpectrumHeightPx()` accessor) - so both settings now affect the Pi's display the same way they affect the Teensy's own.

V1.0.10 (2026-09-25)

Added

- **Vol dropdown:** cycles `Vol -> AGC -> Mic -> STn` via the real `VOLUME_BUTTON` cycle (`Loop.cpp`), same forward-cycle-with-wraparound pattern as Band/Mode/Demod. The mouse wheel now adjusts whichever of those four is currently selected (turning the real Volume rotary encoder), everywhere outside the Menu parameter-adjustment overlay - which required extending `PILINK_MSG_VIRTUAL_ENCODER` from a 1-byte to a 2-byte payload (`[encoder id, direction]`, new `PILINK_ENCODER_FILTER/ PILINK_ENCODER_VOLUME` ids) so one message can drive either the Filter or the Volume encoder's interrupt.

Changed (Pi display only, no reflash needed)

- Toolbar reordered and rebalanced into two fixed rows of 6 and 5, by count rather than by available width: **Menu, Vol, Band, Mode, Demod, Course Inc / Fine Inc, Zoom, Noise, Decode, Home.**

V1.0.9 (2026-09-25)

Added

- Menu parameter adjustment overlay: when a Menu "variableOption" item is selected (e.g. Display → Spectrum Scale), the Pi now shows the same red-boxed "Label: value" readout the Teensy's own `DrawParameter()` draws (in the same spot — the name-badge pane, which normally shows the clock/callsign), fed by a new `PILINK_MSG_PARAMETER` message sent at 10Hz while `uiSM.state_id == UISM_StateId_UPDATE`. The mouse wheel now adjusts the value while that box is showing — each wheel tick sends one virtual Filter-encoder click (new `PILINK_MSG_VIRTUAL_ENCODER` `direction`, same message added in V1.0.8), exactly like turning the real encoder; the wheel is a no-op everywhere else. Box hides again once you leave that screen.

V1.0.8 (2026-09-25)

Added

- Direct **Band dropdown** (160M...4M), replacing the BAND –/BAND+ buttons — built from the real `bands[]/BAND_UP/BAND_DN` cycle order (`Loop.cpp`).
- Eight new toolbar dropdowns, each populated straight from the firmware source rather than guessed: **Mode** (SSB/CW/DATA, `TOGGLE_MODE`’s real cycle), **Demod** (USB/LSB/AM/SAM, `DEMODULATION`), **Noise** (Off/Kim/Spectral/LMS, `NOISE_REDUCTION`), **Course Inc/Fine Inc** (the real `incrementValues[]/selectFT[]` tables), **Decode** (on/off, now backed by a new `decoderOn` Settings-telemetry field so it sets state correctly instead of blindly toggling), **Home** (Home Screen / Toggle Layout — `HOME_SCREEN / SPARE`’s `ToggleHomeScreen()`), and **Menu** — a real two-level picker mirroring the T41’s own menu tree, ported verbatim from `MainBoard_DisplayMenus.cpp`’s `primaryMenu[]`. Selecting a specific menu item needs Filter-encoder turns, not button presses, so this added a new uplink message, `PILINK_MSG_VIRTUAL_ENCODER`, injecting the same `iFILTER_INCREASE/iFILTER_DECREASE` interrupt the physical encoder’s ISR posts.

Changed (Pi display only, no reflash needed)

- “PI Temp:” label now reads “PI Temp(F):”.
- All toolbar dropdown buttons (Zoom, Band, Menu, Mode, Demod, Noise, Course Inc, Fine Inc, Decode, Home) sized to fit their own label/content, wrapping to a second row if they don’t all fit on one line; later made 10% larger for easier touch targets.
- Menu now opens as a single button with real cascading flyout submenus (one category → its own options, like a normal desktop app’s nested menus) instead of a second dropdown button appearing to its right.

V1.0.7 (2026-09-25)

Fixed

- TX/RX status pane showed “XMT” while the radio was actually receiving.** `SendStatus()` was reading the raw PTT hardware pin (`digitalRead(PTT)`), which can read spuriously high with no relation to whether the radio is actually transmitting. Now derived from `modeSM.state_id` — the exact same source the Teensy’s own on-screen REC/XMT indicator (`DrawTXRXStatusPane()`) uses — so the Pi can no longer disagree with the Teensy’s own display.

Changed (Pi display only, no reflash needed)

- TX/RX pane relabeled “REC”/“XMT” (was “RX”/“TX”) to match the Teensy’s own on-screen wording exactly.
- Removed the pane border around the SWR/PWR readout and the S-meter, matching the real Teensy, which borders neither.
- Settings pane: Vol/AGC top row enlarged to 20pt bold; all other fields and the version string enlarged to 14pt; labels now white, live values now green (previously a single uniform gray).
- Spectrum trace is now filled under the curve in the same green as the trace line itself, fading through yellow at 1/3 of the pane’s height and on to red by 1/2 height — banded in 6px strips the same way the Teensy’s own `FillSpectrumColumnGraded()/SpectrumAmplitudeColor()` does, instead of an unfilled line.
- Waterfall now uses the Teensy’s own 117-entry color table (`gradient[], MainBoard_DisplayHome.cpp`) ported value-for-value, instead of a simplified green-only ramp, so waterfall colors match the real hardware’s black→blue→cyan→green→yellow→red→pink heat map.

V1.0.6 (2026-09-25)

Added

- Settings pane full parity with the T41’s own Settings page 1: the Vol/AGC/Mic/STn slot, AGC mode, Tune Increment, Fine-Tune Increment, RX Attenuation, Auto-Notch, TX Attenuation, Noise Reduction, DSP Gain, Antenna Selection, Spectrum Zoom, Sample Rate, CPU Temp, CPU Load, and the running firmware version string — all fed by a new `PILINK_MSG_SETTINGS` wire message, sent at 2Hz.
- Moved the BAND –/BAND+ buttons from the Settings pane to the right of the frequency readout on the VFO pane.
- Fullscreen kiosk mode now reveals the Pi OS’s own top panel when the mouse is moved to the top edge of the screen, without leaving fullscreen — Escape still exits fullscreen entirely.

Fixed

- Audio-filter/audio-spectrum pane’s bottom margin was overlapping the top of the Settings pane directly below it; the pane now reserves the same 30-of-150px bottom margin fraction the Teensy’s own pane does.

V1.0.5 (2026-09-25)

Added

- Live audio-FFT bars in the audio-spectrum pane (new `PILINK_MSG_AUDIO_SPECTRUM` message) — a genuinely live magenta bar display mirroring `ShowSpectrum()`’s `audioYPixel[]` plot. This had been missing entirely; an earlier pass had wrongly concluded this pane was static-only (frequency ruler + passband/CW-filter markers), missing the separate always-running live-bar block.

V1.0.4 (2026-09-25)

Fixed

- Spectrum/waterfall showed outlines but no data even after the V1.0.3 fix: the Teensy's `Serial1` TX buffer was far smaller than a spectrum frame (518 bytes), so `SendFrame()`'s non-blocking guard silently dropped every spectrum frame, forever. Fixed with an explicit 1024-byte `addMemoryForWrite()` buffer in `InitializePiLink()`.
- Waterfall was scrolling bottom-to-top instead of top-to-bottom (Pi-only fix, no reflash needed) — newest data now enters at the top and scrolls down, matching the Teensy's own `BTE_move()` direction exactly.

V1.0.3 (2026-09-25)

Fixed

- Spectrum/waterfall data was pinned to full-scale (a flat line, solid- colored waterfall): `SendSpectrum()` was sending the raw `psdnew[]` FFT magnitude directly into `QuantizeDbm()`, which expects real dBm. Fixed by converting via `dBm = psdnew*10 + RECEIVE_POWER_OFFSET`, the same conversion the Teensy's own trace (`pixelnew()`) uses.

V1.0.2 and earlier

Baseline PiLink protocol (STATUS/SMETER/SPECTRUM messages, virtual-button uplink), the S-meter pane rebuilt to match the real T41's S-meter gauge (scale, tick labels, two-tone track), the SWR/PWR pane reverted to plain text after a bar-gauge version didn't match the real hardware, and an earlier fullscreen fix — all predate this changelog's start and aren't individually dated here.